SECURING CLOUDS USING CRYPTOGRAPHY AND TRAFFIC CLASSIFICATION

A thesis submitted by

Aqeel Sahi Khader Al-nassar

For the award of

Doctor of Philosophy

2018

QUOTES

“The mantra of any good security engineer is: “Security is a not a product, but a process.” It's more than designing strong cryptography into a system; it's designing the entire system such that all security measures, including cryptography, work together.”

Bruce Schneier

"Cryptography shifts the balance of power from those with a monopoly on violence to those who comprehend mathematics and security design.”

Jacob Appelbaum

ABSTRACT

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. Over the last decade, cloud computing has gained popularity and wide acceptance, especially within the health sector where it offers several advantages such as low costs, flexible processes, and access from anywhere.

Although cloud computing is widely used in the health sector, numerous issues remain unresolved. Several studies have attempted to review the state of the art in eHealth cloud privacy and security however, some of these studies are outdated or do not cover certain vital features of cloud security and privacy such as access control, revocation and data recovery plans. This study targets some of these problems and proposes protocols, algorithms and approaches to enhance the security and privacy of cloud computing with particular reference to eHealth clouds.

Chapter 2 presents an overview and evaluation of the state of the art in eHealth security and privacy. Chapter 3 introduces different research methods and describes the research design methodology and processes used to carry out the research objectives. Of particular importance are authenticated key exchange and block cipher modes. In Chapter 4, a three-party password-based authenticated key exchange (TPAKE) protocol is presented and its security analysed. The proposed TPAKE protocol shares no plaintext data; all data shared between the parties are either hashed or encrypted. Using the random oracle model (ROM), the security of the proposed TPAKE protocol is formally proven based on the computational Diffie-Hellman (CDH) assumption. Furthermore, the analysis included in this chapter shows that the proposed protocol can ensure perfect forward secrecy and resist many kinds of common attacks such as man-in-the-middle attacks, online and offline dictionary

attacks, replay attacks and known key attacks. Chapter 5 proposes a parallel block cipher (PBC) mode in which blocks of cipher are processed in parallel. The results of speed performance tests for this PBC mode in various settings are presented and compared with the standard CBC mode. Compared to the CBC mode, the PBC mode is shown to give execution time savings of 60%. Furthermore, in addition to encryption based on AES 128, the hash value of the data file can be utilised to provide an integrity check. As a result, the PBC mode has a better speed performance while retaining the confidentiality and security provided by the CBC mode.

Chapter 6 applies TPAKE and PBC to eHealth clouds. Related work on security, privacy preservation and disaster recovery are reviewed. Next, two approaches focusing on security preservation and privacy preservation, and a disaster recovery plan are proposed. The security preservation approach is a robust means of ensuring the security and integrity of electronic health records and is based on the PBC mode, while the privacy preservation approach is an efficient authentication method which protects the privacy of personal health records and is based on the TPAKE protocol. A discussion about how these integrated approaches and the disaster recovery plan can ensure the reliability and security of cloud projects follows.

Distributed denial of service (DDoS) attacks are the second most common cybercrime attacks after information theft. The timely detection and prevention of such attacks in cloud projects are therefore vital, especially for eHealth clouds. Chapter 7 presents a new classification system for detecting and preventing DDoS TCP flood attacks (CS_DDoS) for public clouds, particularly in an eHealth cloud environment. The proposed CS_DDoS system offers a solution for securing stored records by classifying incoming packets and making a decision based on these classification results. During the detection phase, CS_DDOS identifies and determines whether a packet is normal or from an attacker. During the prevention phase, packets classified as malicious are denied access to the cloud service, and the source IP is blacklisted. The performance of the CS_DDoS system is compared using four different classifiers: a least-squares support vector machine (LS-SVM), naïve Bayes, K-nearest-neighbour, and multilayer perceptron. The results show that CS_DDoS yields the best performance when the LS-SVM classifier is used. This combination can detect DDoS TCP flood attacks with an accuracy of approximately 97% and a Kappa coefficient of 0.89 when under attack from a single source, and 94% accuracy and a Kappa coefficient of 0.9 when under attack from multiple attackers. These results are then

discussed in terms of the accuracy and time complexity, and are validated using a k-fold cross-validation model.

Finally, a method to mitigate DoS attacks in the cloud and reduce excessive energy consumption through managing and limiting certain flows of packets is proposed. Instead of a system shutdown, the proposed method ensures the availability of service. The proposed method manages the incoming packets more effectively by dropping packets from the most frequent requesting sources. This method can process 98.4% of the accepted packets during an attack.

Practicality and effectiveness are essential requirements of methods for preserving the privacy and security of data in clouds. The proposed methods successfully secure cloud projects and ensure the availability of services in an efficient way.

CERTIFICATION OF THESIS

This Thesis is entirely the work of Aqeel Sahi Khader Al-nassar except where otherwise acknowledged. The work is original and has not previously been submitted for any other award, except where acknowledged.

Principal Supervisor: Dr. David Lai

Associate Supervisor: Professor Yan Li

Student and supervisors' signatures of endorsement are held at the University.

ACKNOWLEDGMENTS

To my inspiration, my late father Sahi Khader: I owe it all to you. Many thanks!

Second and foremost, I would like to express my deepest gratitude to my supervisors, Dr David Lai and Professor Yan Li (University of Southern Queensland) for their exceptional guidance, caring, encouragement and patience, and for providing me with an excellent atmosphere for doing research. I would also like to thank Dr Barbara Harmes and Sandra Cochrane for their help and support.

I would also like to thank my mother, sister, and brother. They have always supported me and encouraged me with their best wishes.

I also thank my wonderful children, Zainab, Mayas, and Fadl, for always making me smile.

Last but by no means least, thanks to my wife Saba, for her unconditional love, care, and support. She was always there, cheering me up, and has stood by me through the good times and bad.

I would like to acknowledge the support by Australian Commonwealth Government through the Research Training Program (RTP) Fees Offset scheme during my research.

LIST OF RELATED PUBLICATIONS

1. Sahi, A, Lai, D, Li, Y & Diykh, M 2017, 'An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment', *IEEE Access*, vol. 5, pp. 6036-48. **(Q1 journal, Impact Factor: 3.244, published)**
2. Sahi, A, Lai, D & Li, Y, 'A review of the state of the art in privacy and security in the eHealth cloud', *Computer Science Review*, **(Q1 journal, Impact Factor: 7.63, Submitted)**
3. Sahi, A, Lai, D & Li, Y 2016, 'Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan', *Computers in biology and medicine*, vol. 78, pp. 1-8. **(Q2 journal, Impact Factor: 2.115, published)**
4. Sahi, A, Lai, D & Li, Y, 'Three-Party Password-Based Authenticated Key Exchange Protocol Based on the Computational Diffie-Hellman Assumption', *International Journal of Communication Networks and Distributed Systems*, **(Q3 journal, Impact Factor: 0.75, published)**
5. Sahi, A & Lai, D 2015, 'Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol', in *22nd International Conference on Telecommunications (ICT),: proceedings of the* IEEE, pp. 204-8. **(IEEE conference in Sydney, published)**
6. Sahi, A, Lai, D & Li, Y, 'An efficient hash-based parallel block cipher mode of operation', published in *the 4th International Conference on Computer and Communication Systems (ICCCS2018), Japan.* **(IEEE conference in Japan, published)**
7. Sahi, A, Lai, D & Li, Y 2015, 'Parallel Encryption Mode for Probabilistic Scheme to Secure Data in the Cloud', in *10th International Conference on Information Technology and Applications (ICITA)* Sydney. **(IEEE conference in Sydney, published)**
8. Sahi, A, Lai, D & Li, Y 2017, 'An Energy Efficient TCP DoS Attacks Mitigation Method in Cloud Computing', in *The First MoHESR and HCED Iraqi Scholars Conference in Australasia 2017*, Swinburne University of Technology, pp. 289-94. **(A conference at Swinburne University of Technology, published)**

A copy of these papers can be found in Appendix D

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

CHAPTER 1

INTRODUCTION

The unintentional and unexpected exposure of data in the eHealth sector is increasing at a disturbing rate. According to the Identity Theft Resource Centre (ITRC, 2015), in 2015, health organisations in the United States were attacked 276 times, exposing 177,866,236 health records and in 2016, they were attacked 376 times, compromising 15,942,053 health records,. It is clear that the health industry remains a prime target for cybercriminals (Roberts, 2017). This motivates the researcher to study and propose a suite of solutions to handle the security and privacy of eHealth clouds using cryptography and other techniques, thus helping health industry stakeholders gain maximum benefits from eHealth cloud projects.

1.1 Introduction

Cloud computing is a model for the use of computer resources and other modern technological functionality in the information technology world, providing services such as storage and applications (Mell & Grance, 2011). Users can access and use cloud computing services without the need to acquire knowledge, expertise or even administrative rights for the infrastructure that supports these services. There are three main types of services offered by the cloud: software as a service, platform as a service and infrastructure as a service (Kushida et al., 2015; Sugumaran et al., 2014). In addition, four deployment models have been identified for cloud architecture solutions; namely private, community, public and hybrid clouds (Hsu & Cheng, 2015; Zissis & Lekkas, 2012).

The technologies used in cloud computing serve to ensure simple and straightforward on-demand network access to a shared group of computing resources,

which can be prepared quickly and put together with minimal effort (Meenakshi, 2012). This has become an important direction for technology research, and many scientists and researchers claim that cloud computing has changed both computing processes and IT markets. Powered by cloud computing, users can adopt comprehensive sets of tools to access various applications, storage and platforms through the Internet (Vuyyuru et al., 2012). Since clouds share distributed resources through the Internet and intranets in environments that may not be secure, security is an important factor.

Some have expressed concern over storing all of their data and files in the cloud (Chen & Zhao, 2012; Gonzales et al., 2012; Ma, 2012). If a successful attack can be launched on an exposed service, an attacker may be able to obtain all the users' information. Furthermore, if an attacker were able to use or re-sell the stolen information, this would constitute a privacy and security problem. To overcome these issues, cloud service providers often use secure and reliable encryption techniques to secure their data. The provision of data privacy and security in cloud computing projects is therefore desirable.

Cryptography is one of the most widely used and practical options for cloud computing developers seeking to ensure a high level of security (Li et al., 2013). Cryptography is the art of changing plain information into a cipher or hidden information, where only the intended user(s) or machines can gain access to the original information (Kapoor et al., 2011; Lek & Rajapakse, 2012). It is divided into two main categories based on the keys used: symmetric and asymmetric cryptosystems (Elminaam et al., 2010). Symmetric cryptography systems require the sender and the receiver (for example, Alice and Bob) to share an identical key for the encryption and decryption processes. This shared symmetric key must be changeable, for better protection, and must be strong enough to stop a third party (Eve) from getting the original information (Forouzan & Mukhopadlhyay, 2011). Asymmetric cryptography systems require Alice and Bob to each have a set of two different keys. The asymmetric cryptography systems process is slower than the symmetric one (Kumar et al., 2013).

The safety of any symmetric cryptosystem relies heavily on the key exchange algorithm used, such as the Diffie-Hellman (DH) protocol. A key exchange algorithm is a technique for exchanging the keys in a secure way between the users (Forouzan & Mukhopadlhyay, 2011; Kumar et al., 2013). There are many algorithms for distributing the keys between Alice and Bob after Alice has physically chosen a key,

such as sending the key to Bob by mail, by phone or in person. Alice may also use a previous key shared with Bob to encrypt the new key and send it to Bob. Unfortunately, these techniques are currently insecure in wide network systems. In addition, a very strong encryption algorithm such as the Advanced Encryption Standard (AES) algorithm may be used, but if Alice and Bob do not secure the keys, their systems can easily be compromised.

Furthermore, cryptanalysis shows weaknesses in block ciphers, such as parallelisation for cipher block chaining (CBC) mode (Beeputh et al., 2010). In view of these issues, we need a faster block cipher mode, such as a method using a probabilistic encryption scheme.

Hence, the development of a secure cloud system is vital to ensure the security, privacy, and availability of data in the cloud. This thesis focusses on key exchange protocols, block cipher modes of operation, security and privacy preserving methods, disaster recovery plans, and the detection and prevention of flood attacks. It reviews current research and existing solutions, and proposes protocols, algorithms and approaches that will benefit cloud users in many sectors, such as health, education, and government. The outcomes of this study will help to improve the security, privacy and availability of clouds such as eHealth clouds.

1.2 Research Problems

Cloud are used by millions of people around the world. Cloud services give users the opportunity to store data in the cloud for easy access, anytime and anywhere (Carroll et al., 2011). However, the use of a cloud poses many security and privacy problems. Both security and privacy are vital for data distribution in the cloud (Chen & Zhao, 2012). In the cloud environment, a user's data are controlled by the service providers rather than by users themselves. There is, therefore, a potential for data leaks, either intentional or accidental, which is unacceptable (Gonzales, et al., 2012; Ma, 2012). Furthermore, data in the cloud are stored in geographically diverse locations. Thus, confidentiality, authentication, and communication between parties become important concerns (Hussain & Ashraf, 2014). When users use cloud services, they may not know precisely where their information is held (Gampala et al., 2012). In this situation, it is better for cloud service providers to offer high levels of encryption to secure the confidentiality of data wherever they are stored, and an appropriate key exchange method should be adopted.

Confidentiality is not enough to guarantee cloud computing security. Users want to ensure that their data cannot be modified or compromised by a third party. Cloud service suppliers should also apply methods to guarantee data integrity (Sugumaran et al., 2014). In addition to integrity issues, the availability of services must be ensured. Flood attacks caused by distributed denial of service attacks (the second most common cybercrime attacks after information theft) must be detected and prevented, and excessive energy consumption reduced. Hence, the following research questions were identified:

1. Has a researcher in this area reviewed the major and relevant literature on cloud security and privacy?
2. How can the security of key exchange between parties be hardened, and different types of attacks prevented?
3. How can the performance of the adopted encryption algorithm be improved on top of security assurance?
4. How can security and privacy be preserved in eHealth clouds? How can a client be enabled to connect to the system at any time, even during a disaster?
5. How can DDoS TCP flood attacks be detected and prevented? How can the energy consumption caused by DoS attacks be reduced?

1.3 Research Hypotheses

Having identified the research questions of this study, it is time to be more specific. Five tentative predictions (hypotheses) for the study are proposed:

H 1. Investigation and review of the relevant literature on eHealth cloud security and privacy would facilitate the researcher's decisions on research directions

H 2. Developing a key exchange protocol based on the Computational Diffie-Hellman assumption should secure key distribution between parties and protect systems against multiple attacks

H 3. A new parallel hash-based block cipher mode of operation would be able to improve the encryption process on top of security assurance

H 4. Integration of potential key exchange protocol and block cipher mode would preserve privacy and security on eHealth clouds. A disaster recovery plan with heartbeat signals will result in 24/7 services availability

H 5. Classification techniques can detect DDoS attacks in eHealth clouds, and filtering techniques can reduce excessive energy consumption caused by DoS attacks.

1.4 Contributions of the Thesis

Data security and privacy are sensitive, critical, and play a vital role in the information technology world (Ritchey et al., 2013). Cryptography can be used effectively to support and enhance data security and users' privacy in cloud computing (Jaber & Zalkipli, 2013). This study is focused on security, privacy, prevention and mitigation of attacks on clouds such as eHealth clouds, using encryption and other techniques. The expected outcomes of this research are as follows:

1. For a fuller understanding of the research directions in security and privacy in eHealth clouds, it summarises and analyses the state of the art publications in eHealth cloud security and privacy. An extensive review is conducted, and over 100 studies from several peer-reviewed databases such as IEEE Xplore are investigated. The selected studies are reviewed and summarised in terms of their benefits and risks. Details can be found in Chapter 2. The content of this chapter has been submitted to the journal *Computer Science Review*
2. A TPAKE protocol based on the CDH is presented. It securely distributes keys among users and protects systems against man-in-the-middle (MITM) attacks, amongst others. Details of this are given in Chapter 4. Part of the content of this chapter has been accepted by the *International Journal of Communication Networks and Distributed Systems*, and the remainder was published in the *International Conference on Telecommunications (ICT), 2015 22nd: Proceedings of the IEEE*
3. A PBC mode of operation is introduced in which blocks of cipher are processed in parallel, ensuring both high performance and security. The details are presented in Chapter 5. Part of the content of this chapter was published in *The 4th International Conference on Computer and Communication Systems (ICCCS2018), Japan*, and the remainder was published in the *10th International Conference on Information Technology and Applications (ICITA), Sydney*
4. The PBC mode and TPAKE protocol are used to introduce both security preservation and privacy preservation approaches, thus ensuring both the privacy and the security of eHealth clouds. In addition, a disaster recovery plan that ensures

connectivity for users during a disaster is presented. The details are given in Chapter 6. The content of this chapter has been published in the journal *Computers in Biology and Medicine*

5. Finally, a classification-based security system that helps to detect and mitigate DDoS TCP flood attacks is proposed. A method to mitigate DoS attacks in the cloud by reducing excessive energy consumption via limitation of the number of packets is also presented. Rather than a system shutdown, the proposed method ensures continued availability of service. The details of these approaches are presented in Chapter 7. Part of the content of this chapter has been published in the journal *IEEE Access*, and the remainder was presented in the *First MoHESR and HCED Iraqi Scholars Conference in Australasia*, Melbourne.

1.5 Connections Between Chapters

This research focuses on the hardening of security for data in eHealth cloud projects. This study uses a TPAKE protocol to prevent multiple attacks in DH key exchange (Chapter 4), and introduces a new parallel encryption mode entitled PBC (Chapter 5). TPAKE and PBC are then brought together to preserve the security and privacy of eHealth clouds. In addition, a disaster recovery plan that ensures uninterrupted connectivity for users during a disaster is presented (Chapter 6). A system that detects and prevents DDoS attacks, and a method for reducing the excessive energy consumption caused by DoS attacks are introduced (Chapter 7). A diagrammatic representation of the relationships between chapters is given in Figure 1.1.

1.6 Structure of the Thesis

This thesis consists of eight chapters, each of which provides important information related to the research project. The rest of the thesis is structured as follows:

Chapter 2 provides an overview of cloud computing, in terms of its security and privacy and some necessary background knowledge. It briefly introduces contextual information and knowledge related to the current research on eHealth cloud security, privacy and associated issues. The chapter then summarises and analyses state-of-the-art security and privacy issues related to the cloud. An extensive review is conducted in this chapter, and over 100 studies from several peer-reviewed databases such as

IEEE Xplore were examined. The selected studies are reviewed and summarised in terms of their benefits and risks.

Chapter 3 describes some research design methodologies and the methodology adopted in this research project. The Vaishnavi and Kuechler research design methodology, and its five phases (shown in Figure 1.1) are described to provide an overall picture of the process used in later chapters.

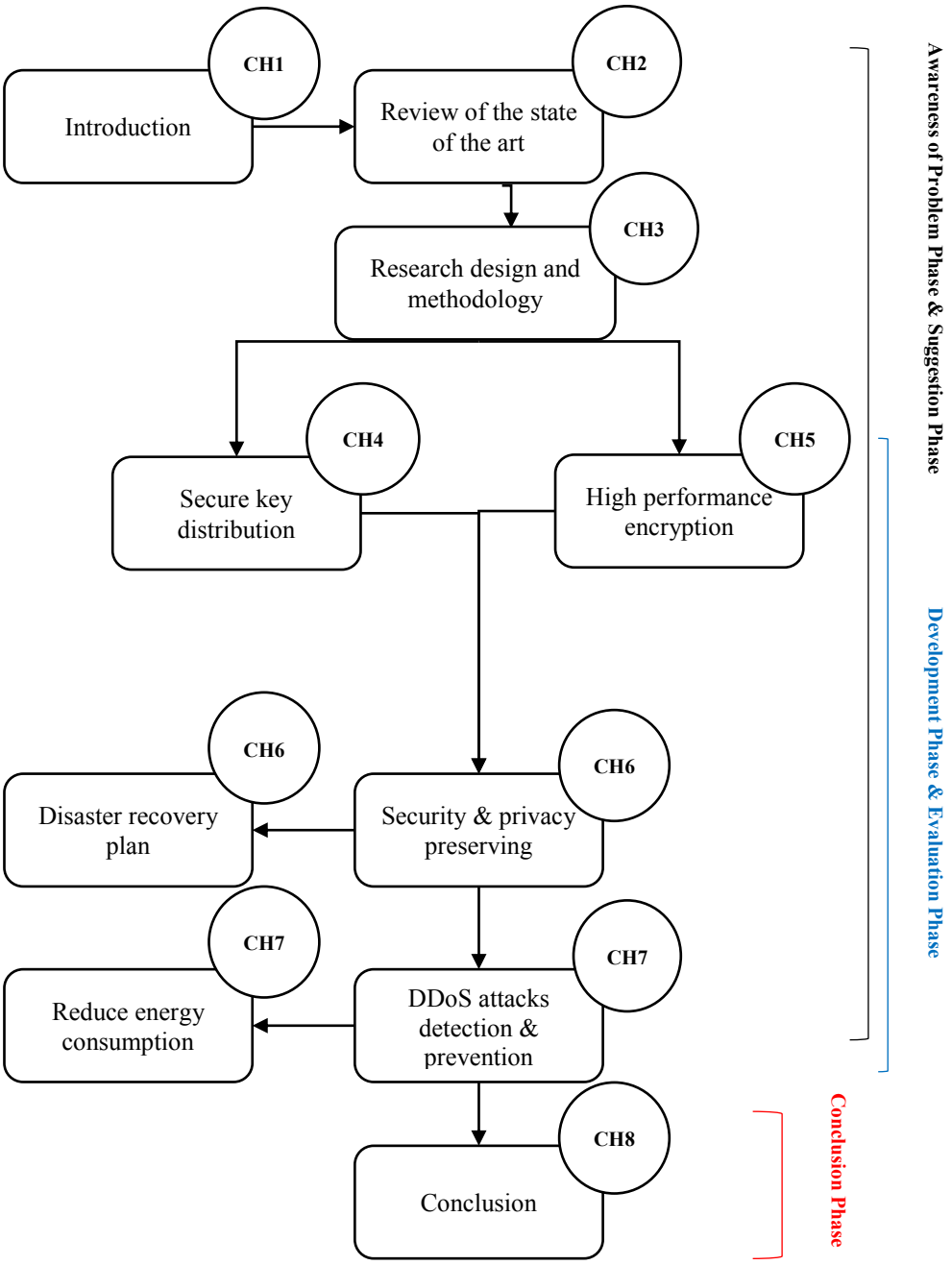


Figure 1.1 Connections between chapters

Chapter 4 identifies a gap in the security of key exchange between parties, and presents a new TPAKE protocol that securely distributes keys between users and protects systems against MITM attacks, among others.

To enhance the speed performance of encryption, the PBC mode is introduced in **Chapter 5**. In PBC, blocks of cipher can be processed in parallel to ensure both high performance and security.

In **Chapter 6,** with the security and privacy of eHealth clouds in mind, the PBC mode and the TPAKE protocol are integrated to form a security-preserving approach and a privacy-preserving approach. In addition, this chapter presents a disaster recovery plan which ensures uninterrupted connectivity for users during disasters.

Chapter 7 presents a classification-based security system that helps to detect and prevent DDoS TCP flood attacks. This chapter also introduces a method for mitigating DoS attacks in the cloud and reducing excessive energy consumption by limiting the number of packets. Instead of a system shutdown, the proposed method ensures the continued availability of service in the case of a DoS attack. The method was evaluated with more than 14,900 attacking packets per second and the server can still provide services (no missing packets).

Chapter 8 presents a summary and the findings of this study. Possible directions for future work are also discussed in this chapter.

The **Appendices** provide programming code for some of the proposed methods, code for the techniques described in Chapters 4, 5, and 7, and a copy of related publications.

2

CHAPTER 2

LITERATURE REVIEW

To fully understand the research directions of security and privacy in eHealth clouds, this chapter summarises and analyses the security and privacy issues and the state-of-the-art solutions proposed for eHealth clouds. A review is conducted in this chapter, and over 100 studies from several peer-reviewed databases, such as IEEE Xplore were investigated. The selected studies have been reviewed and summarised in terms of their benefits and risks.

2.1 Introduction

The official definition of cloud computing, according to the National Institute of Standards and Technology is as follows: "cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (Mell & Grance, 2009). Over the past decade, cloud computing has gained popularity within the health sector as it offers several advantages such as low costs and flexible processes (Li et al., 2013). Cloud-based health services allow physicians, patients, and owners of health data to control and share their data easily. However, eHealth cloud computing poses a range of challenges, such as data security and privacy for clients and cloud service providers (CSPs) (Yu et al., 2017). Security and privacy issues threaten an open network and semi-trusted servers which may lose, leak or disclose data (Tang et al., 2016). This can breach users' privacy.

A great deal of research has explored the security and privacy issues in eHealth clouds, and many solutions have been suggested to resolve the identified issues. To

obtain a clear picture of the security and privacy problems that can affect eHealth clouds, this literature review summarises and analyses the current state of the art in eHealth security and privacy. The aim of this study is to deliver a clear and complete picture of eHealth security and privacy issues, and to discuss recent research that targets these issues and the proposes solutions. As shown in Figure 2.1, the study is divided into five main categories: security and privacy, security controls, effective encryption, data security requirements, and disaster recovery plans.

Although cloud computing is widely used in the health sector, numerous issues remain unresolved (Jing et al., 2013; Sahi et al., 2015). Several studies have attempted to review the state of the art in security and privacy in eHealth clouds (Abbas & Khan, 2014; Fernandez-Aleman et al., 2013; Thilakanathan et al., 2014 (B); Gonzalez-Martinez et al., 2015; Sajid & Abbas, 2016; Yuksel et al., 2017) however, some of these studies are now rather outdated, and others do not cover certain vital features of cloud security and privacy such as access control, revocation and data recovery plans.

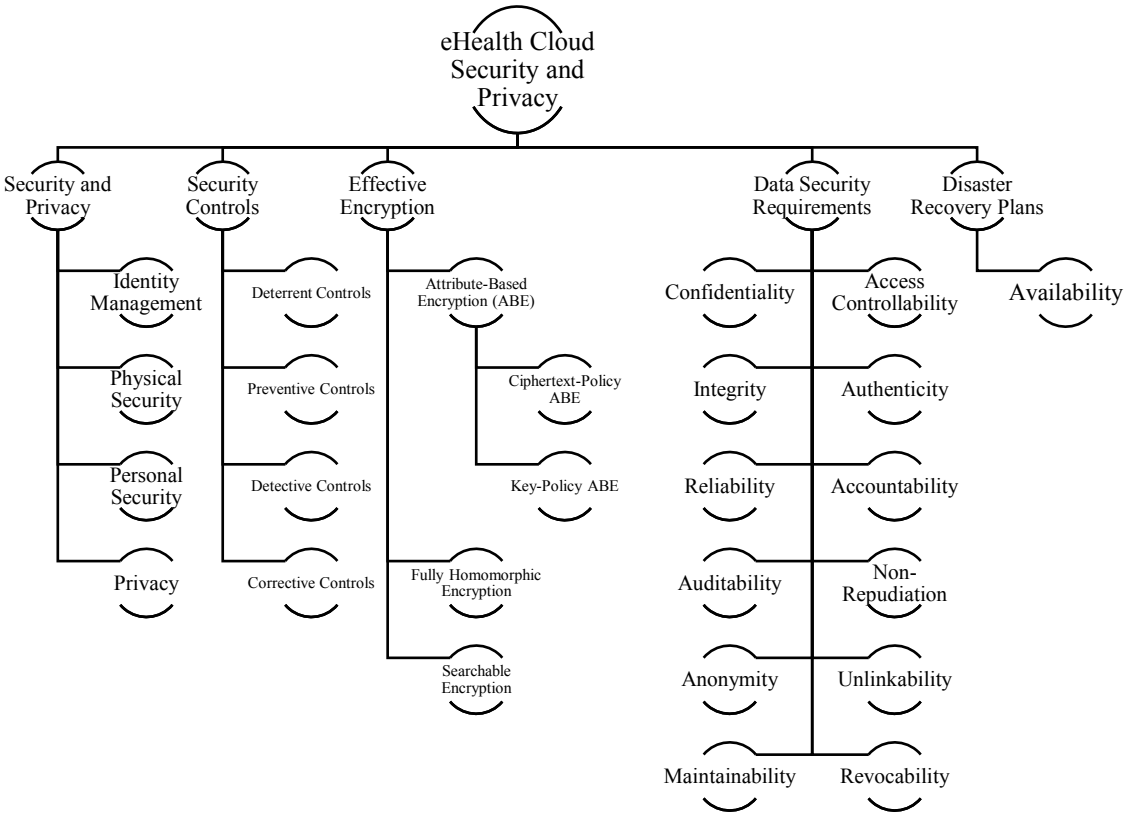


Figure 2.1 Security and privacy taxonomy for eHealth clouds

In addition, some of the existing review papers focus on either the privacy of the cloud or the security of the cloud, but not both. This chapter classifies and evaluates these review papers. The review covers most of the recent studies in this area, and can be considered a good literature base for researchers in the area of eHealth clouds.

The remainder of this chapter is organised as follows, using a structure similar to that of Figure 2.1. Section 2 describes the proposed schemes with regard to the security and privacy of eHealth clouds; Section 3 describes the proposed schemes with regard to security controls; Section 4 describes the effective encryption of eHealth clouds; Section 5 discusses the data security requirements of eHealth clouds; Section 6 describes disaster recovery plans; and finally, Section 7 concludes the chapter.

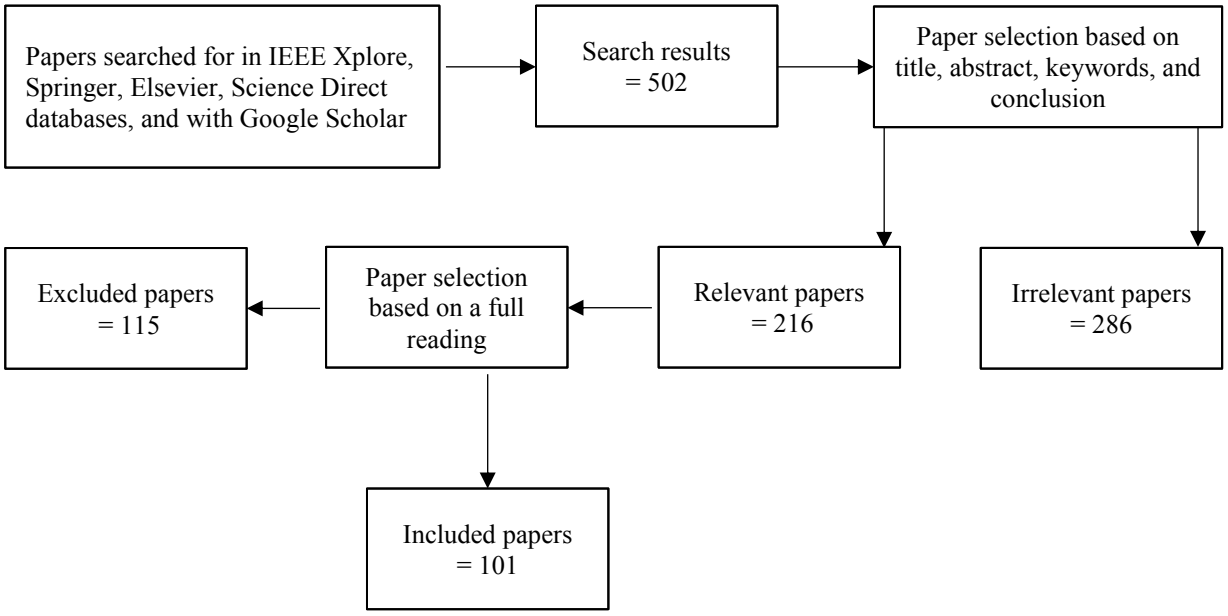


Figure 2.2 The inclusion/exclusion process

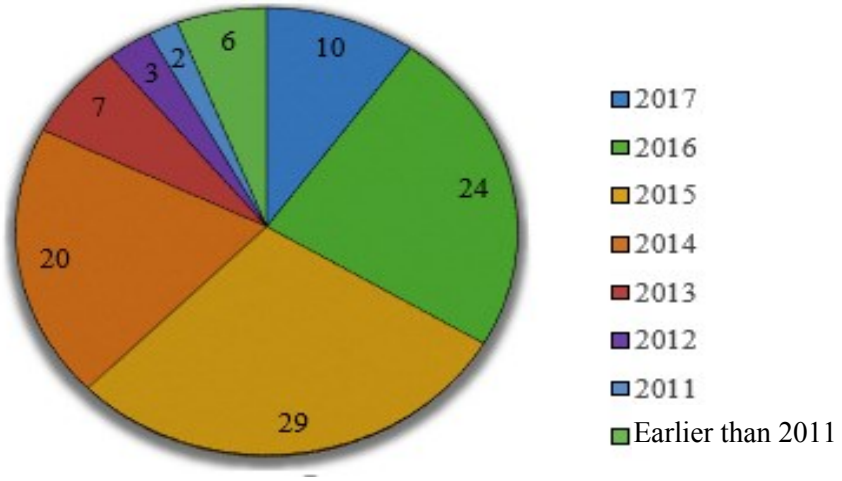


Figure 2.3 Distribution of the selected articles by year

2.1.1 Paper Selection

Papers were selected according to their titles, abstracts, keywords, and conclusions to include the most relevant papers and to exclude irrelevant ones from the study. Figure 2.2 shows the inclusion and exclusion processes, and Figure 2.3 shows the distribution of the selected articles over the years.

The contributions of this study are as follows: Collection and evaluation of papers on the current state of the art in eHealth security and privacy schemes. The collected papers are classified into five categories, as shown in Figure 2.1.

In the next section, the literature of security and privacy in the cloud will be reviewed.

2.2 Security and Privacy in the Cloud

Cloud computing is a model commonly used to save money and effort in many sectors, and particularly in the health sector. However, despite the benefits of eHealth clouds, there are still many unresolved issues regarding security and privacy which require a great deal of research to resolve.

2.2.1 Identity Management

An Identity Management System (IMS) is a comprehensive organisational system used to identify entities in a cloud project. Access to information and resources in the project is managed by linking client privileges and constraints with a proven identity. The main aim of an IMS is to determine what clients can do within a cloud project and under what conditions. In addition, an IMS is utilised to improve the security and privacy of a cloud system, and to reduce the running costs and effort.

According to Bichsel et al., there is a high security risk of identity theft and impersonation of the user by the Cloud Service Providers (CSP) or an intruder who has gained access to the CSP resources (Bichsel et al., 2013; Bichsel et al., 2014). To manage access to data and resources, CSP use either their own IMS (such as CloudID (Haghighat et al., 2015)), or incorporate the client's IMS into their infrastructure. Take, for example, the use of a biometric-based IMS to preserve the privacy of the cloud project's information (Haghighat et al., 2015). A biometric-based IMS is used to connect the private data of the clients to their biometrics, which are saved as ciphertexts. To ensure that the CSP or any possible attackers cannot obtain any type

of access to this private information, the proposed biometric-based IMS is implemented in an encrypted domain using a searchable cryptographic system.

Recently, Wang et al. proposed a computationally efficient secure eHealth cloud system using the Identity Based Encryption (IBE) method (Wang et al., 2017). In this system, there are four parties with different roles: the cloud, the health community, physicians, and patients. The system works as follows. First, the system sets up public and private keys for all parties according to their published identities (e.g. email addresses). These identities are considered to be public keys and are used to generate private keys using an IBE algorithm. Next, the Electronic Health Records (EHRs) are encrypted by the parties using a block cipher algorithm such as AES, and the keys are encrypted using the IBE and sent to the cloud. Following this, the parties can receive the encrypted EHRs from the cloud and decrypt them using their identity keys.

According to a survey conducted by Khalil et al., (2014), more than 66% of users' identities are stored in unsafe places. Khalil et al. (2014) therefore proposed a new IMS system called the Consolidated Identity Management (CIDM) system, which they claim is resistant to certain attacks, such as server compromise attacks, mobile device compromise attacks, and traffic interception attacks (Khalil et al., 2014). The CIDM mechanism splits permission identifications and spreads them between the parties at the IMS to prevent traffic interception attacks. To mitigate mobile device compromise attacks, a challenge-response approach is adopted. Finally, the security of the communication channels between the CIDM and the CSP is addressed to reduce the possibility of any effective compromise of that channel (Khalil et al., 2014). However, further investigation is required to resolve the problem of insufficient dynamic federated identities and privacy in most of the current IMS systems (Sanchez et al., 2012). This is an architectural problem and must be considered at the design level. Also, a survey of 67 random mobile users is hardly a justification for the claim, as 66% means only 44 people.

Haufe et al. proposed a new framework named the Information Security Management System (ISMS) (Haufe et al., 2014) consisting of many vital security procedures for eHealth clouds. The proposed security management framework was implemented based on the ISO 27000 family of standards. The ISMS was able to identify the most frequent cloud computing threats and the information that these attacks aimed to collect from the cloud system (Haufe et al., 2014). One drawback is

that the ISMS needs specific details from processes, such as input, output, and interfaces, to facilitate communication and interaction between processes.

In another study, the concept of the Identity Management as a Service (IDaaS) was discussed (Nunez & Agudo, 2014). In this work, the authors proposed a new IMS called BlindIdM which preserved the privacy of data and delivered them as IDaaS. Specifically, the authors described how a system based on the Security Assertion Markup Language (SAML) was employed with proxy encryption to enhance the security of the cloud project with respect to the CSP (Nunez & Agudo, 2014). To improve the proposed system, extending the IDaaS from a single domain to a cross-domain approach has been suggested, as in the System for Cross-domain Identity Management (SCIM) (Grizzle et al., 2015; SCIM, 2017).

Xiong et al. proposed a scheme named Privacy pReserving Identity and Access Management (PRIAM) (Xiong et al., 2014), which has five components: registration, token withdrawal, tenant pre-authorisation, access control, and token spending. PRIAM is described as being able to fulfil all the requirements of cloud security. The proposed scheme uses a hash function, signature, and mutual authentication to ensure the privacy of clients. To deliver secure access control for clients and CSP, it utilises a service-level agreement. Finally, Burrows Abadi Needham (BAN) logic is used to confirm the correctness of the scheme (Burrows et al., 1989).

2.2.2 Physical Security

Physical security is the concept of securing and controlling hardware, software, networks and data from physical actions and events that could cause serious loss or damage to workstations. In other words, the aim of physical security is to prevent intruders from accessing a cloud's physical facilities. Cloud hardware, such as servers, switches etc., is also physically secured by the CSP from any unusual activities such as attacks, threats, and floods (Regola & Chawla, 2013), and is provided with the necessary power supplies to reduce any potential interruptions.

Mxoli et al. showed that in order to protect Personal Health Records (PHRs) from any physical intrusion, the system hardware must have a physical security border (Mxoli et al., 2014). For example, physical access control, offices and rooms must be secured, and resistance against disasters and other environmental situations must be available. All of these security borders must be in place to ensure that the cloud and network equipment is not readily accessible to the public. The equipment and

applications used by the CSP, which may contain PHRs, must not be moved out of the site or repositioned without the administrator's authorisation (Mxoli et al., 2014).

The IT equipment building, or the site where data centres and other cloud hardware are located, must be properly secured. Rodrigues et al. highlighted that these buildings must be secured by security staff members, video surveillance systems and Intrusion Detection Systems (IDS); in addition, only authorised people should be allowed to enter the building using authenticated access controls (Rodrigues et al., 2013).

Carlson stated that CSPs should adopt Federal Information Security Management Act (FISMA) standards to ensure the physical security of their records. Since the physical entrances to the physical machines and storage devices are a possible route for data compromise, FISMA must be implemented at the client sites as well as at the server sites (Carlson, 2014).

2.2.3 Privacy

Credentials such as digitally signed IDs or other information used to authenticate or identify a client have not been designed to respect the clients' privacy (Rannenberg et al., 2015). CSP uses encryption and other techniques to preserve the privacy of clients' critical information, such as credit card numbers, and only authorised clients have the right to access this kind of information.

Abbas and Khan reviewed the state of the art in eHealth cloud privacy in 2014 (Abbas & Khan, 2014). This part of the review aims to cover, not only the issues regarding privacy, but also other security concerns such as storage security, access controls, and disaster recovery plans. This section will, therefore, first review some of the proposed approaches with regard to eHealth privacy. More details on privacy preservation approaches can be found in Abbas and Khan article (Abbas & Khan, 2014).

A three-factor authentication protocol based on Elliptic Curve Cryptography (ECC) was proposed by Yeh et al. in 2013 (Yeh et al., 2013). The protocol has certain disadvantages, such as a vague procedure, impractical IDs, and no shared key (Wu et al., 2015). In addition, the protocol cannot prevent spoofing attacks (Wu et al., 2015). Another authentication protocol based on a fingerprint was proposed by Khan and Kumari (Khan & Kumari, 2013). However, this protocol cannot mitigate impersonation or desynchronisation attacks (Wu et al., 2015). To overcome the weaknesses of these protocols, Wu et al. proposed a new biometrics-based three-factor

authentication protocol that can overcome all of these drawbacks as well as ensuring the privacy of clients (Wu et al., 2015). This protocol uses the ECC and mobile devices, and adopts a fuzzy extractor to deal with inadequate biometric inputs. The protocol proposed by Wu et al. was formally proven using random oracles and Elliptic Curve Gap Diffie–Hellman (ECGDH) problem assumption to demonstrate the low probability of attack success. However, this protocol is vulnerable to other attacks, such as impersonation and offline password guessing attacks, if the mobile device falls into the wrong hands. In addition, the user revocation procedure is not included in the protocol (Jiang et al., 2016 (A)). Therefore, another three-factor authentication protocol that is able to resist these attacks and offers more security features was proposed by Jiang et al. in 2016 (Jiang et al., 2016 (A)).

Yang et al. presented a privacy preservation approach for health records in eHealth clouds (Yang et al., 2015). This approach is based on the classification of health record attributes. It collects these attributes vertically from the health dataset to ensure that these are collected from all areas of the dataset with different privacy aspects. Their approach consists of four steps: (1) vertical data collection, (2) data merging, (3) integrity checks, and (4) plain and cipher text searches. Cryptography and statistical analysis are combined to create multiple approaches which can strike a balance between the use of health records and privacy preservation (Yang et al., 2015). This approach, however, does not consider the situation where several users are using the service at the same time.

Many online services and applications need some authentication of users to start trust relations, either for only one endpoint of communication or for both. A password-based authentication mechanism is widely used for this (Camenisch et al., 2012). Scheme proposed by Sahi et al. aims to preserve the privacy of the PHRs (Sahi et al., 2016). This scheme adopts a three-party password-based authenticated key exchange protocol (3PAKE) based on the computational Diffie–Hellman assumption proposed by Khader and Lai (Khader & Lai, 2015). This scheme uses a different generator and primitive root in each session to ensure that only the specific client has complete access to his/her PHR, and clients are revoked at the end of the session. This ensures that old session keys cannot be used to access a client's PHRs. A disaster recovery plan and a break-glass technique are also addressed in this scheme.

Based on the HireSome-I method, an improved history record-based service optimisation method (HireSome-II) was proposed by Dou et al. in 2015 (Dou et al.,

2015). HireSome-II was proposed in order to ensure the privacy of big data such as health records in cloud computing. The cloud rejects requests which reveal transaction information for privacy reasons, and the proposed method can efficiently support the cloud service structure to complete transactions securely (Dou et al., 2015).

Another framework to ensure the privacy of patient data was proposed by Page et al. (Page et al., 2015). This framework combines monitoring and analytic methods to deliver secure and authenticated health records. This framework was based on fully homomorphic encryption (FHE). However, FHE is known as computationally-intensive therefore, to measure the practicality of the proposed framework, the authors developed a proof of concept and a prototype system (Page et al., 2015).

In the next section, the literature of cloud security controls will be reviewed.

2.3 Cloud Security Controls

Security approaches are effective in cloud environments when an excellent protection mechanism is adopted. This mechanism must identify the potential problems that may arise during the management process. These problems will be addressed and considered by the security controls, thus preserving the security of the system from its own weaknesses and reducing the number of attacks (Sajid & Abbas, 2016). There are many cloud security controls, which can be categorised as follows.

2.3.1 Deterrent Controls

Deterrent controls aim to reduce the number of attacks on a cloud project. A “No Trespassing” sign can alert security personnel to watch out for intruders as well as highlighting the consequences of intrusion. Deterrent controls serve to warn attackers that there will be penalties and punishments if they proceed with attacks (Nedelcu et al., 2015; Rajamani et al., 2016).

2.3.2 Preventive Controls

Preventive controls aim to secure cloud projects by preventing or decreasing vulnerabilities. For example, an effective authentication protocol can ensure the security of the cloud’s clients and prevent any unauthorised access to that cloud. Preventive controls can therefore help the cloud system to confidently identify their clients (Nedelcu et al., 2015; Rajamani et al., 2016).

2.3.3 Detective Controls

Detective controls aim to detect and respond appropriately to attacks which could threaten the cloud system. During an attack, the detective control will notify the preventive control or the corrective control to report the problem. An intrusion detection system (IDS) is typically used as a detective control (Li et al., 2013; Yu et al., 2017).

2.3.4 Corrective Controls

Corrective controls aim to reduce the damage of an attack. These controls are usually initiated during or after attacks. Restoring a cloud system from a backup to ensure the availability of services is an example of a corrective control (Nedelcu et al., 2015; Rajamani et al., 2016).

Generally, access controls are linked to security policies delivered to clients while accessing the service. A company typically has its own security controls which allow staff members access to a set of data rather than giving them full data access. This control limits the access of a staff member to a particular group of data. These kinds of security controls must be put in place in cloud projects to avoid unauthorised access. The Software as a Service (SaaS) model must be sufficiently elastic to combine the set of controls offered by the company (Subashini & Kavitha, 2011).

Recently, much research has been done on cloud security controls. Some of these works are discussed in the following paragraphs.

Many stakeholders might access PHRs without authorisation. Access control is therefore a major problem for the privacy of data when health records are stored and shared in the cloud. Thus, a dynamic access control is necessary to ensure the privacy of the stored health records. Son et al. proposed a new dynamic access control scheme for securing the privacy of the PHRs in cloud projects (Son et al., 2017). Their scheme can detect unauthorised access dynamically by altering the context information, meaning that even if the subject has the same role, access authorisation will not be defined in the same way, according to the conditions and the context information. The proposed scheme was tested using a real-life health system.

Tong et al. proposed another access control architecture designed to ensure the privacy of data (Tong et al., 2014). To overcome the misuse of health records, the proposed architecture had several features including key exchange, storage data privacy, emergency retrieval, and auditability. A pseudorandom number generator was used as a key exchange to ensure unlinkability, and a redundancy-based secure

indexing feature was proposed to preserve the privacy of the data by hiding the search and access patterns. Finally, to mitigate any potential misbehaviour, an attribute-based encryption was integrated with threshold signing to be used in emergency and normal situations as an access control with auditability.

Based on two-stage keyed access control and a zero-knowledge protocol, Kahani et al. proposed another security control method (Kahani et al., 2016). Their method aimed to facilitate access control and authentication in electronic health cloud systems. When a user requests access to a health record, a limited amount of access will be extracted based on the user's rights. To connect two parties in the system securely, a two-stage key management is used. This two-stage key management is a combination of public key encryption and Derived Unique Key Per Transaction (DUKPT)

Fernando et al. proposed a new approach which aimed to reduce leaks of patient information using unlinkability (Fernando et al., 2016). This approach provides the health data owner with the ability to make decisions in terms of access control. To fulfil the policies of the service provider, the proposed approach utilises a personal information management protocol which improves the privacy of the patients.

This approach depends on a scenario in which patient EHRs are stored on a Health Information Exchange (HIE) cloud service. The approach demonstrates the communication techniques between EHR consumers, EHR owners, EHR creators, and the HIE service. The authors claimed that the privacy of the EHR was ensured by the unlinkability of consumers' sessions with the HIE service. In addition, the HIE service cannot reach the consumer classes even when they have access policies. The proposed approach works as follows: a patient consults a doctor and the doctor prescribes a medical test. The patient goes to a laboratory with the doctor's instructions, and the laboratory carries out the test. The results of the test are sent by the laboratory to the HIE. Finally, the patient provides access to the doctor and the HIE (Fernando et al., 2016).

In 2015, Wand et al. proposed a new scheme called Constant-Size Ciphertext Policy Comparative Attribute-Based Encryption (CCP-CABE). This method inserts similar characteristics from all attributes into a key and combines the restrictions of these attributes into a single chunk of a ciphertext. This procedure is carried out during the encryption process to apply elastic access control rules with a variety of relationships. The authors showed that the CCP-CABE scheme was efficient, as it produced keys and ciphertexts of the same size each time for any number of attributes, as well as

reducing the cost of the computation to a trivial size. To ensure access privacy, the authors extended CCP-CABE to numerous attribute domains (Wang et al., 2015 (C)).

Younis et al. proposed a new model named Access Control for Cloud Computing (AC3) (Younis et al., 2014). This model utilises the role and task principles, as shown in Figure 2.4, and uses clients' jobs as a categorising factor. Based on the clients' jobs' roles, security domains are created to restrict each client to a particular security domain. Each role within the AC3 is given a group of related and required tasks for performing those roles. For access to data and resources, security classification is done for each task, and an authentic permission is required to complete the task. The authors employed a risk engine to interact with unpredictable client behaviours. However, an authentication protocol that can deal with massive storage complexity and high performance is required.

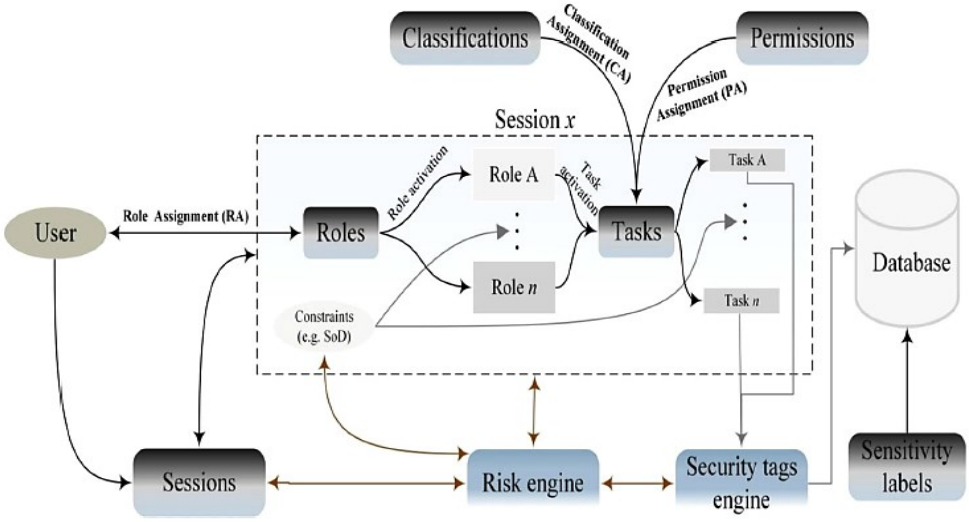


Figure 2.4 Access Control for Cloud Computing (AC3), adapted from (Younis et al., 2014)

In 2014, Yang and Jia proposed a new multi-authority access control scheme (Yang & Jia, 2014). In this scheme, the authors presented a Ciphertext-Policy Attribute-based Encryption (CP-ABE) scheme. This is an extension to a single-authority scheme proposed by Lewko and Waters in (Lewko & Waters, 2012). Yang and Jia adopted Chase's multi-authority scheme (Chase, 2007) in which all generated secret keys for the same client are combined. CP-ABE also uses a revocable scheme and can mitigate collusion attacks. More specifically, the functionality of a single authority is divided into a certificate authority and multiple attribute authorities. The proposed multi-authority access control scheme is shown in Figure 2.5.

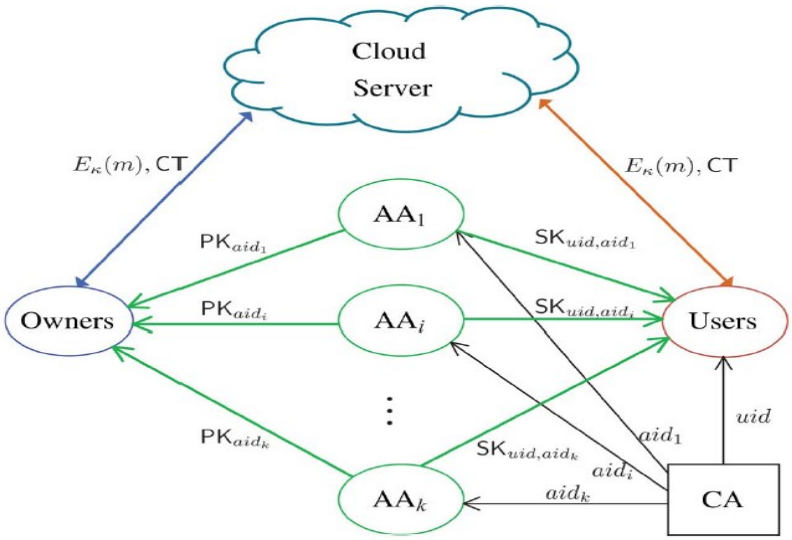


Figure 2.5 Multi-authority access control scheme, adapted from (Yang & Jia, 2014)

Li et al. adopted Semantic-Based Access Control (SBAC) techniques to propose a new architecture called IntercroSsed Secure Big Multimedia Model (2SBM) for securing access between different cloud systems (Li et al., 2016 (B)). In addition to the architecture shown in Figure 2.6, the 2SBM architecture can be summarised in three steps:

- To relate attributes to each other, the proposed architecture formats the data by linking the attributes in a matrix
- Based on these relationships, the architecture creates interrelations between attributes in the matrix
- The architecture builds a tree of attributes and sorts the attributes according to their frequency, thus improving the efficiency of access control.

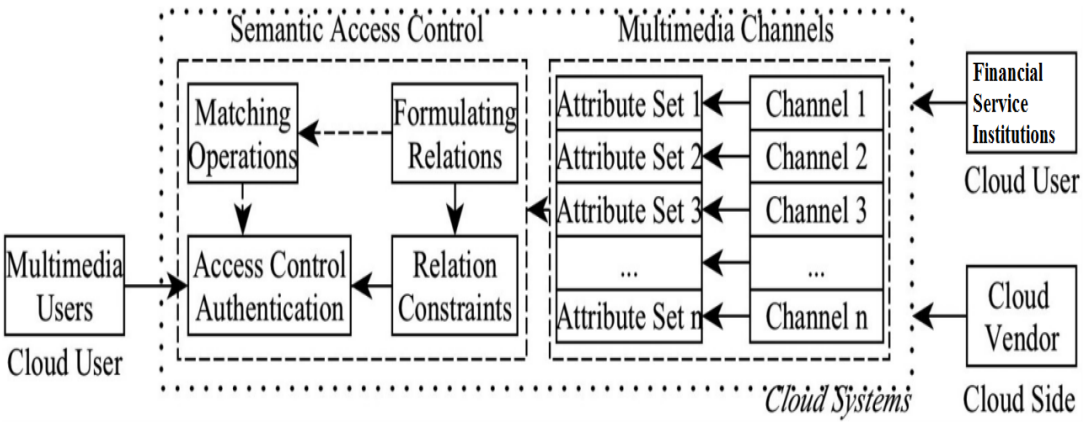


Figure 2.6 Architecture of 2SBM, adapted from (Li et al., 2016 (B))

Choi et al proposed the ontology-based Access Control Model (Onto-ACM) in 2014 (Choi et al., 2014). Onto-ACM is a model of analysis which recognises and presents the differences between providers and clients. Based on this analysis, the ontology cognitive and context-aware technologies, the proposed model can decide whether to allow data access or not. This model can be seen as a detailed access control, which can be used to establish cloud feature boundaries.

Yu et al. proposed a new scheme which claimed to achieve secure, scalable, and fine-grained access policies for cloud projects (Yu et al., 2010). The proposed scheme uses attribute-based encryption (ABE), proxy re-encryption (PRE), and lazy re-encryption. Specifically, it allows the data owner to pass the operations of computation to the servers without revealing the original data. In this scheme, the data owner is therefore responsible for the accessibility of the data, which is particularly suitable for cloud projects.

Ruj et al. proposed another form of access control in 2014 (Ruj et al., 2014). In this method, there are three types of clients: creator, reader, and writer. For example, Alice is the client and a trusted party gives her a token. The trusted party could be any government office controlling health records. When submitting a claim, Alice presents her identification (e.g., a health card), and the trusted party provides her with the token. In this scheme, there are two key distribution centres (KDCs) which are responsible for distributing the keys to the clients. Based on the information in the token and the keys from one or two of the KDCs, a creator makes a decision on the claim, ensuring the identity of Alice and authenticating and encrypting the messages under this claim. The signed ciphertext is then sent to the cloud. The cloud system authenticates the signature of the ciphertext and keeps it on the cloud servers. When the reader asks to read a message, the cloud system will send the ciphertext. Without the appropriate keys, the user would not be able to retrieve the plaintext; however, the access control manager has full access to all client information and can decrypt the ciphertexts.

In the next section, encryption used for some of the proposed security and privacy protocols will be reviewed.

2.4 Effective Encryption

Several advanced encryption algorithms have been used in cloud computing to protect the security and privacy of eHealth data. Encryption schemes such as public key encryption (PKE) and symmetric key encryption (SKE) are frequently used to protect

data in eHealth cloud projects (Abbas & Khan, 2014). Other encryption schemes are also used to ensure the security and privacy of eHealth records, including attribute-based encryption (ABE), fully homomorphic encryption (FHE), and searchable encryption (SE).

2.4.1 Attribute-Based Encryption Algorithms

The first ABE algorithms were presented by Sahai and Waters in 2005 (Sahai & Waters, 2005), and by Goyal et al. in 2006 (Goyal et al., 2006).

ABE is a type of PKE where the ciphertext and shared key of a client are reliant on attributes (Pletea et al., 2015). In ABE systems, retrieving a plaintext from ciphertext is applicable for clients who have a group of key attributes that match ciphertext attributes. One of the most important features of the ABE system is that it is collusion resistant; an attacker who has many keys can only access the system when at least one key has an endorsed access. Recently, numerous researchers have proposed ABE algorithms; some of these are discussed in the following paragraphs.

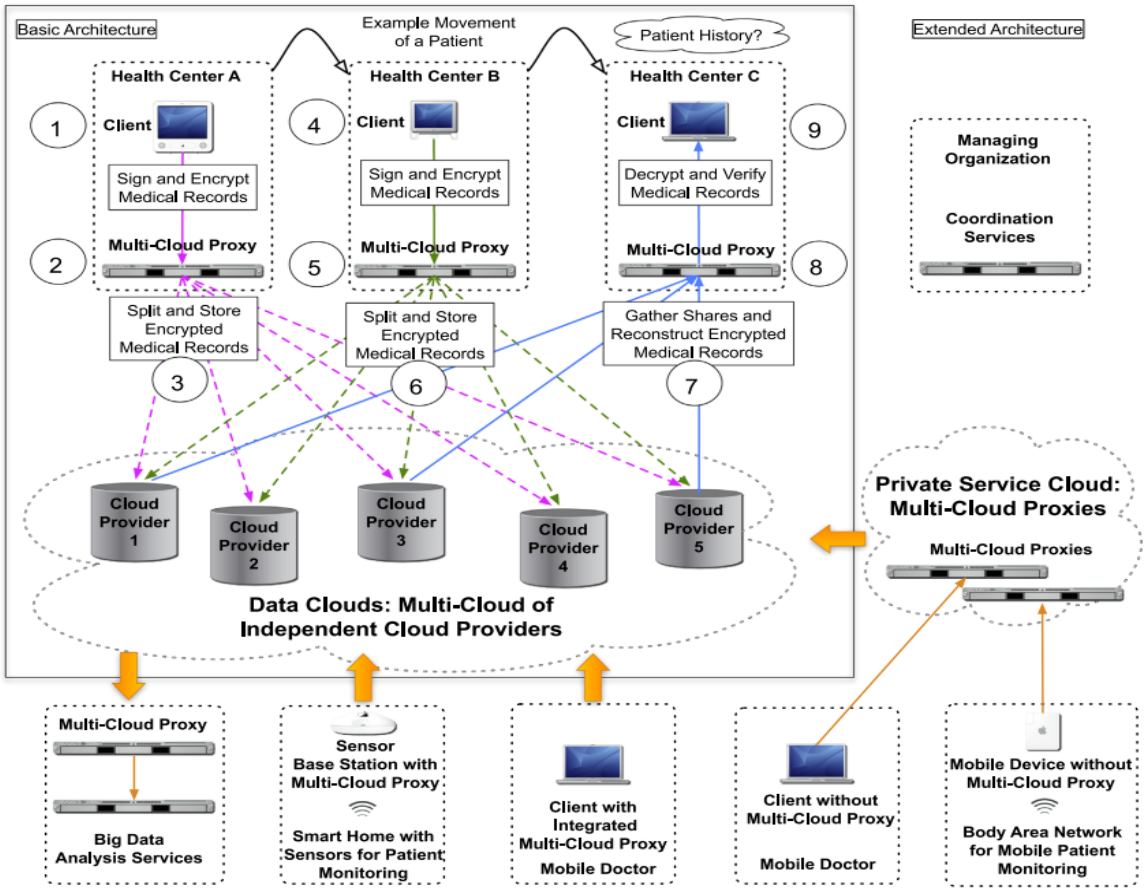


Figure 2.7 Scheme proposed by Fabian et al. (Fabian et al., 2015)

Fabian et al. proposed a new ABE-based scheme for secure data sharing in eHealth clouds (Fabian et al., 2015). The proposed scheme aims to preserve the security and privacy of patients' records in partly trustworthy cloud servers. It uses the ABE algorithm to manage users' accessibility to health records and shared keys, and to distribute information and health records among several clouds. Figure 2.7 shows a patient visiting three different health centres (HCs A, B and C). His/her health record is updated at each of the three centres. When the patient visits HC C, the doctors at HC C can request the full health record for that patient from HCs A and B through the multi-cloud proxy. However, the key management process needs to be reconsidered and solved. In addition, the key authority of the ABE algorithm has to be distributed, and security responsibilities must be separated.

Li et al. proposed a new ABE-based framework for secure sharing of PHRs in eHealth clouds (Li et al., 2013). The authors assumed that the cloud servers were semi-trusted; they also argued that the PHR records had to be encrypted to ensure the privacy of the patients. They used the ABE algorithm to encrypt PHRs, and patients can delegate others from public domains to access their PHR records. This work involved verifying key management complexity reduction and privacy enhancement. The proposed framework involves multiple data owners, clients, attribute authorities (AAs), and SDs. The framework can use one of two ABE algorithms: the revocable key policy ABE system proposed by Yu et al. was used for each public and personal domain (PSD) (Yu et al., 2010), and the authors proposed their own revocable MA-ABE system to be used for each personal and public domain (PUD).

Outsourced ABE (OABE) approaches can significantly decrease the computational cost of encryption by moving the large computation to a CSP. However, large encrypted files which are saved on the cloud are likely to affect the query processing in a negative way. Li et al. therefore proposed a keyword search function (KSF-OABE) approach, which aims to solve this problem (Li et al., 2017). KSF-OABE offers key issuing, decryption, and keyword search functions. It retrieves part of the ciphertext according to a particular keyword. In this approach, operations that consume a large amount of time will be moved to the CSP, while users needing less time will process consuming operations. Thus, the processing time can be reduced on both the CSP and user sides. However, the proposed KSF-OABE approach does not offer verifiability features. The proposed approach was tested only for a Replayable Chosen-Ciphertext Attack (RCCA) and was not tested for a chosen-ciphertext attack (CCA). CCA-secure

approaches are RCCA-secure, although RCCA-secure approaches are not CCA-secure. Therefore, testing under both CCA and RCCA conditions is suggested.

A PHR system based on the ABE algorithm was presented by Xhafa et al. for secure sharing and storing of PHRs in the cloud (Xhafa et al., 2015 (B)). This system permits users to share their PHRs and personal information selectively with health service providers. The proposed system is practical as it provides searchability, revocation, and local decryption.

Based on their operations, ABEs can be classified as ciphertext-policy or key-policy ABEs.

2.4.1.1 Ciphertext-Policy ABEs

In the ciphertext-policy (CP-ABE) approaches, the encryptor manages the access operation. The public key process is more complex due to the complexity of the access operation and hardens the system. Most CP-ABE research concentrates on the access control design (Su et al., 2011).

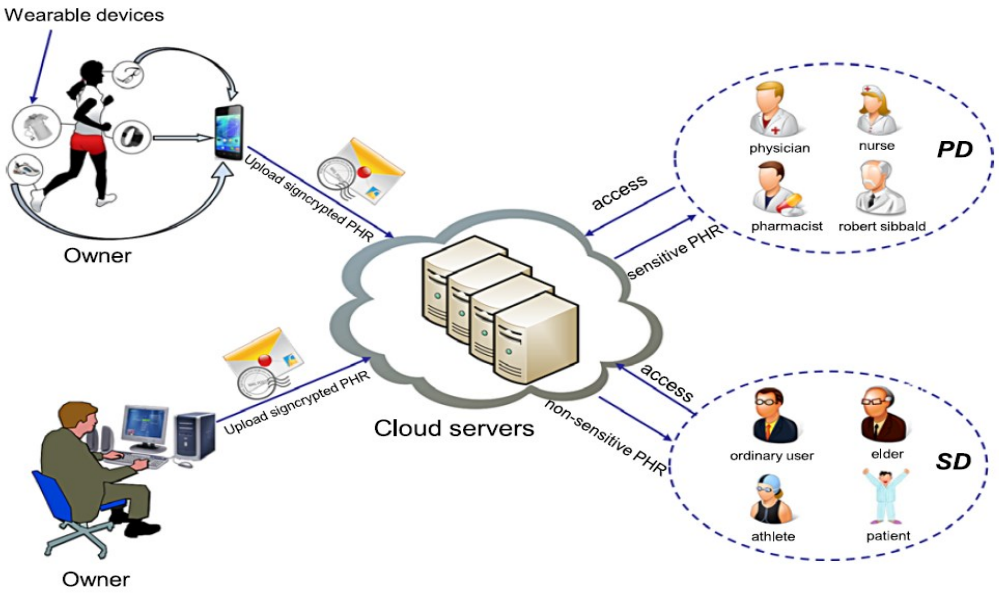


Figure 2.8 Approach proposed by Liu et al. (Liu et al., 2015 (C))

Liu et al. proposed a new approach based on CP-ABE with a signature (SignCryption), called CP-ABSC. It delivers PHR authentication, encryption, and access control (Liu et al., 2015 (C)). The proposed approach allows a patient to sign the PHR record using a secret key and a group of personal attributes, as shown in

Figure 2.8. CP-ABSC has two features: access control and signature encryption (SignCryption). The authors claim that a combination of these two features can deliver the authenticity, unforgeability, confidentiality and collusion prevention required by a PHR system. A revocation process, however, is not considered. In addition, according to Rao, this approach cannot provide verifiability for a public ciphertext property which is necessary to resist any invalid ciphertext decryption in order to decrease the redundant load on the decryptor (Rao, 2017).

As a result, in 2017, Rao proposed another CP-ABSC approach for PHR cloud projects which claims to be verifiable for a public ciphertext (Rao, 2017). This approach satisfies the important security properties of the attribute-based signature (ABS) and ABE. Furthermore, it uses communication links to a lesser extent than other approaches. This CP-ABSC has two assumptions: Existential Unforgeability in Selective Signing Predicate and adaptive Chosen Message Attack (EUF-SSP-CMA) and the resistance of the Computational Diffie–Hellman Exponent (CDHE) problem, and Decisional Bilinear Diffie–Hellman Exponent (DBDHE) problem" (Rao, 2017).

These assumptions can prevent the "indistinguishability of ciphertext in selective encryption predicate and adaptive chosen ciphertext" attack (IND-sEP-CCA2).

Wang et al. (Wang et al., 2015 (A)) introduced another cloud-based PHR (CB-PHR) system. CB-PHR permits the owners of PHRs to safely store their records in a partly trustworthy CSP, and to share them with several clients of their choice. PHR clients are divided into public and personal domains to decrease the complexity of key management. In this approach, health records are encrypted by the owner of the PHRs using CP-ABE for presentation to the public domain, whereas health records are encrypted using a nameless multi-receiver identity-based encryption algorithm for the personal domain. Therefore, only accredited clients whose identification can meet the CP specifications can decrypt health records (Wang et al., 2015 (A)). The authors suggest services such as statistics and retrieval of data. It should be mentioned that the CB-PHR has a high computational cost, as it encrypts the same record twice.

Motivated by cloud security requirements, Xu et al. modified the CP-ABE scheme to propose a new Verifiable Delegation CP-ABE (VDCPABE) (Xu et al., 2016). This cloud computing scheme is based on verifiable technology and multilinear maps. Hybrid encryption is used to encrypt data by its owner. For each ciphertext block, a verifiable message authentication code (MAC) is generated privately, and the full

ciphertext is then uploaded to the cloud. When the data owner is not online, the client who requests the data can ask the cloud server directly (Xu et al., 2016).

Health records are usually represented using a multilayer hierarchical structure. However, according to Wang et al., this hierarchical characteristic of health records has not been investigated thoroughly in terms of CP-ABE (Wang et al., 2016 (B)). They, therefore, proposed a data hierarchy ABE approach for such cloud projects. A single access control method is used rather than levelled access control methods, and the hierarchical data are encrypted using this single access control method. The proposed scheme was shown to reduce storage and time costs, since the parts of the ciphertext related to attributes are distributed by the records (Wang et al., 2016 (B)).

A PHR privacy preserving approach based on a multi-authority CP-ABE which offers revocation features and ensures fine-grained access was proposed by Qian et al. (Qian et al., 2015). The authors show that this approach can be implemented in a partly trustworthy server and encrypted PHRs with multiple owners stored on that server. The proposed approach is able to work in public cloud PHR systems (Li et al., 2013). Once PHRs encryption is complete, to achieve a fine-grained access, the patient can combine ciphertext with multilayer access attributes. A key exchange scheme is used to preserve the privacy of the PHRs. This key exchange scheme ensures that if cracked, authorities will expose zero information regarding the client's global identifier (GId). As a result, the tracing of a GId by an attacker yields no information about the client's attributes. The revocation of lazy client and on-demand services are features provided by this approach to decrease the computational overhead (Qian et al., 2015).

Another approach based on CP-ABE was proposed by Guo et al. to secure EHRs in health cloud environments (Guo et al., 2016). This approach uses a CP-ABE algorithm to encrypt tables published by healthcare providers, such as EHRs. The patient's identification number is used as a primary key to store these records in a database. This permits multiple clients with multiple constraints to search multiple database columns. The authors highlight that their work differs from others in terms of securing outsourcing records, as the search management of columns in the database is emphasised (Guo et al., 2016).

Xhafa et al. presented a multi-authority CP-ABE approach with a patient accountability feature to secure PHR sharing in a health cloud project (Xhafa et al., 2015 (A)). In the proposed work, patient privacy is secured by the access control policy

being hidden. The reduction of authority and PHRs trust assumptions are ensured through the accountability feature.

2.4.1.2 Key-Policy ABE (KP-ABE)

In the Key-Policy Attribute-Based Encryption (KP-ABE) schemes, ciphertext has a group of attributes, and the access regulations are controlled by the client's private key. Ciphertext can be decrypted only when these groups of attributes match the structure of access to the client's private key (Attapadung et al., 2012).

Based on the Decisional Bilinear Diffie-Hellman (DBDH) assumption, a privacy-preserving KP-ABE (PP KP-ABE) approach was proposed for secure data sharing in a cloud system (Rahulamathavn et al., 2016). This approach permits clients to retrieve data from the cloud and then decrypt it, without exposing any attribute information to a third party. The issue of collusion attacks is resolved in this research, as PP KP-ABE is collusion resistant. The authors of PP KP-ABE utilise a key management scheme to strengthen the connection between client and the secret key, thus multiple clients cannot use their secret keys to produce a secret key for an unapproved client (Rahulamathavn et al., 2016).

Another KP-ABE-based scheme named access policy re-definable ABE (APR-ABE) was proposed by Qin et al. for securing EHRs in cloud environments (Qin et al., 2015). In APR-ABE, attribute vectors are used to implement access control. This access control is linked to clients' secret keys; higher level clients can easily redefine their access control to be commensurate with their roles, and can then provide lower level clients with a secret key that has more limitations.

2.4.2 Fully Homomorphic Encryption (FHE)

FHE is a type of encryption that has a special feature permitting operations to be conducted on a ciphertext as well as on plaintext (Maral et al., 2016). This is a vital feature, especially in modern ICT systems, as it enables the possibility of chaining several services together without leaking information. There are several schemes which secure health records using FHE, and some of these are discussed in the following paragraphs.

A FHE-based scheme was proposed to secure computations for the Genome-Wide Association Study (GWAS) (Lu et al., 2015). The proposed scheme aims to preserve the privacy of patients' genomic data. It adapts FHE to encrypt genotype and

phenotype data for all patients to implement meaningful operations on a ciphertext. However, the authors do not consider the computational complexity of the FHE in their proposed scheme, which is a major issue for the proposed FHE scheme (Kumarage et al., 2016).

Another approach based on FHE was proposed to preserve the privacy of health data in a public cloud (Kocabas & Soyata, 2015; Page et al., 2014). A detailed analysis was provided based on heart rate (average), heart rate (max/min), and the automated detection of irregular heartbeats. The authors provided a set of experimental results over 24 hours using an electrocardiogram (ECG) signal dataset and a homomorphic encryption library (HElib). The results show that the proposed approach can be adapted for a health cloud system to secure data from these sources (Kocabas & Soyata, 2015; Page et al., 2014). However, the proposed scheme does not solve the problem of computational complexity in FHE. The implementation of this approach in a real-time parallel system also needs to be considered to reduce the procedure time.

Zhao et al. proposed a different FHE-based system to solve the issue of lack of data safety in a health cloud (Zhao et al., 2014). The authors claim that the proposed method is suitable for both retrieving and processing ciphertext for secure storage of health data on cloud servers and transmission of data between the cloud and the clients. This method was able to offer search date for a third party. However, in the same way as the previous methods, this method also suffers from high computation requirements.

2.4.3 Searchable Encryption (SE)

SE is a cryptographic scheme that provides safe searching in a ciphertext. For enhanced effectiveness, SE typically constructs keyword indexes to verify client requests. SE schemes can be based on either a public key or secret key. Many proposals have been offered to deliver secure search over encrypted text, and some of these are described below.

Yang and Ma proposed a time-dependent SE approach with a designated tester and timing enabled proxy re-encryption function (Re-dtPECK) (Yang & Ma, 2016). This approach allows patients to give limited access privileges to others, which helps to control search procedures over the health records within a particular timeframe. People who are given access privileges by patients can search and decrypt health records within this limited timeframe. In addition, Re-dtPECK offers a linked word search, and can prevent guessing attacks (Yang & Ma, 2016). However, the revocation feature

is not considered in this approach as the patient holds the same key most of the time, meaning that Re-dtPECK needs to consider redistributing secret keys among authorised clients.

A scheme named secure channel-free searchable encryption (SCF-PEKS) has been proposed to offer a secure search over encrypted EHR (Wu et al., 2016 (B)). This version of SCF-PEKS was shown to be able to reduce storage and computational costs when compared to the previous SCF-PEKS. Moreover, it can resist keyword guessing attacks. However, despite reductions in storage and computation costs, ranked and fuzzy keyword searches were not provided, and integrity checks were missing.

Another proposed scheme uses a Bloom filter tree index to permit accredited users to retrieve data from ciphertext in a cloud (Song et al., 2017). In addition to the proposed scheme, the authors introduced a ranking method based on keyword membership, to retrieve only vital keywords. The authors argued that their work was the first to be able to retrieve full encrypted text from a large cloud storage database. However, a collusion attack could possibly threaten the proposed scheme.

Liu et al. proposed a novel EHR cloud project which aimed to safely share and store EHR records in a cloud environment (Liu et al., 2016). The proposed approach is based on binary trees for saving EHR ciphertext, and the ABE algorithm is adopted for efficient encryption of the shared keys. The authors claimed that the proposed project was designed to secure EHRs, and these were encrypted using a symmetric algorithm. With fewer cryptographic operations, a searchable encryption scheme might improve the system further. However, integrity checks were not offered by the proposed system.

Since the security of data sharing is an important factor for any cloud-based system, especially health cloud systems, Liang and Susilo defined a new notation searchable attribute-based proxy re-encryption (ABPRE) scheme to address this issue (Liang & Susilo, 2015). However, the authors did not state how they might reduce the search token size, and how a key holder could create tokens. A modified scheme was recommended to address these issues.

In addition, Li et al. introduced two fine-grained multi-keyword search (FMS) schemes, FMS_I and FMS_II (Li et al., 2016 (A)). FMS_I was designed to provide an accurate search by considering common keyword factors and related scores. FMS_II was built to offer a secure complex search, which might contain several keywords connected with logical operations such as “AND” and “OR” operations. Finally, to

enhance the efficiency of the proposed schemes, FMS classified support (FMSCS) sub-dictionaries were proposed. However, the proposed method cannot deal with a multi-user cloud.

Finally, a multi-keyword SE method was proposed to safely search over encrypted text on a cloud (Xia et al., 2016). This method was able to offer dynamic operations such as insert and delete operations. The authors designed their own tree-based index, as well as a "greedy depth-first search" method to enhance the ranked search using multiple keywords. They chose the KNN algorithm to encrypt the query and the index. In addition, this algorithm was chosen to compute the score of the connections between the query and the index. Shade terms were inserted into the index to prevent statistical attacks. However, a revocation feature is not offered by the proposed approach, as the patient holds the same key most of the time, as in Re-dtPECK discussed above.

In the next section, the data security requirements will be reviewed.

2.5 Data Security Requirements

Several security issues are related to cloud systems such as EHR cloud-based systems. These issues include not only common concerns such as DDoS attacks (Sahi et al., 2017 (A)), but also specific issues in the cloud such as side channel attacks, etc. (Tang et al., 2016). Thus, setting security requirements for any cloud systems is essential and must be included in the review. From an eHealth cloud perspective, the security requirements (R) of cloud system include the following.

2.5.1 Confidentiality (R1)

The confidentiality of data in a health cloud system means that unauthorised clients cannot decrypt or retrieve health records. The data owner, for example the patient, does not control the health records stored in the cloud (Li et al., 2013). Authorised clients are the only users who can access the records; even CSPs are not allowed to access any information regarding the data. Furthermore, patients expect full control over their health records in the cloud, without any leakage to other legitimate users or attackers.

2.5.2 Access Controllability (R2)

Access controllability means that a data owner controls his/her data by implementing certain carefully constructed rules in order to ensure the security and privacy of

records, and by allowing only legal users to have controlled access (Tang et al., 2016). Other users cannot access health records without permission. Users have different access rights to access different parts of the data. This is called fine-grained access control. In an untrusted cloud system, the data owner is the only one permitted to grant access.

2.5.3 Integrity (R3)

Integrity is a security feature that ensures the completeness and accuracy of data. In other words, data must stay complete and must not be altered or deleted; users normally expect their data to be kept safe in cloud storage (Liu et al., 2015 (A)). Furthermore, users must be able to detect any unsolicited modification, loss, or corruption of this data, and to retrieve lost pieces.

2.5.4 Authenticity (R4)

Authenticity is assurance that a message, transaction, or other exchange of information is from the source it claims to be from (Abbas & Khan, 2014). Authenticity involves proof of identity. We can verify authenticity through authentication.

2.5.5 Reliability (R5)

Reliability means that the system performs as users expect (Lehr, 2015). One of the main factors of reliability is availability, which means that continuity of service is provided. In other words, availability means how long the system is expected to serve users without interruption (Lehr, 2015).

2.5.6 Accountability (R6)

As cited in (Felici & Pearson, 2015), “defining what exactly accountability means in practice is complex”. One definition is that the controller of the data must be responsible for acting in accordance with procedures that affect the privacy of data.

2.5.7 Auditability (R7)

Auditability means monitoring security, privacy, and all access activities on eHealth clouds (Abbas & Khan, 2014). To ensure that no errors accrue, auditing must be done from time to time.

2.5.8 Non-Repudiation (R8)

Non-repudiation means that no one can falsely deny any unethical behaviour (Chen et al., 2014; Mihaita et al., 2017). In eHealth cloud environment, patients and physicians cannot deny any misuse or mishandling of health records.

2.5.9 Anonymity (R9)

The anonymity of the user means preventing a third party from obtaining valid user information that leads to accessing the server (Jiang et al., 2016 (B); Sharma & Kalra, 2016). As the attacker is unable to learn any personal information, anonymity ensures the privacy of legitimate users in the cloud. A lack of anonymity means an attacker can be fake identity as an authenticated user.

2.5.10 Unlinkability (R10)

Unlinkability means that, in order to ensure a user's privacy, associating information with a particular user must be difficult (Wu et al., 2016 (A)). Although sometimes a group of words needs to be used for a particular function, this group of words should be different each time. Thus, a random generation function is required (Liu et al., 2015 (B)).

2.5.11 Maintainability (R11)

Maintainability means the ability to perform fast maintenance on a project as the development of very large projects is often not fully complete (Biswas et al., 2014). Maintainability can therefore ensure the delivery of services without error for different parties. In addition, a testing method is needed to decrease the time of maintenance.

2.5.12 Revocability (R12)

Revocability means that users' access rights should be revoked after a period of time so that they cannot access specific data later on using old keys (Thilakanathan et al., 2014). Revocability is a vital feature for eHealth cloud systems and needs to be well implemented to ensure the privacy of users and the secrecy of the contents (Yang, 2015). Once a manager chooses to revoke a particular user's rights, the corresponding keys need to be eliminated from the system.

Table 2.1 shows a comparison of security approaches for eHealth clouds in terms of data security requirements. Techniques, aims, limitations, and server assumptions

of twelve approaches have been compared in this table. For each approach, all listed security requirements have been checked to see whether these approaches have ensured them or not, each requirement can be valid, invalid or not as specified by the author. For example, the first approach used proxy re-encryption and El-Gamal encryption techniques (Thilakanathan et al., 2014). The aim of this approach is to design in-home monitoring. However, a data sharing service is assumed to take place between fully trusted parties. Server assumptions also have not been specified. In terms of security requirements, R1, R2, and R12 are valid in this approach, R5 is not specified, and all other security requirements are invalid.

In the next section, the literature of disaster recovery plans is reviewed.

Table 2.1 Comparison of security approaches for eHealth clouds

Ref.	Technique(s)	Aim(s)	Limitation(s)	Server assumption (s)	Data Security Requirements											
					R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
(Thilakanathan et al., 2014)	Proxy re-encryption, El-Gamal encryption	In-home monitoring	Data sharing service assumed to be fully trusted party, usability tests not available	—	√	√	X	X	—	X	X	X	X	X	X	√
(Castiglione et al.,	Least Significant Bit	3D medical images secure management, limited computational power	Unparalleled, missing many security requirements	—	√	√	√	√	X	X	X	X	X	X	√	X
(Fabian et al., 2015)	ABE, cryptographic secret sharing	Multiple cloud data distribution, reduction in attackers' abilities	Centralised ABE key authority, separated security duties, usability tests not available	Semi-trusted servers	√	√	√	√	√	—	√	√	√	√	√	√

(Hu et al., 2017)	Internet of Things (IoT) medical sensor	Secure elder monitoring system, medical resource reduction	Computationally inefficient, missing some vital security requirements such as revocation	Trusted servers	√	√	√	√	√	X	X	√	X	X	X	X
(Li et al., 2013)	ABE	Patient-centric PHR access control, multiple clouds, key complexity reduction	Computationally inefficient	Semi-trusted servers	√	√	√	√	√	√	X	—	X	X	—	√
(He et al., 2014)	Multiple hashes	Preventing Denial of Service (DoS) attacks, dissemination protocol for WBANs	Unparalleled as it uses CBC, many assumptions, missing vital requirements such as revocation	—	√	√	√	√	—	X	X	√	X	X	√	X
(Gope & Hwang,	IoT, Body Sensor Network (BSN)	Secure IoT-based healthcare system using BSN, computationally efficient	Missing vital security requirements such as revocation	—	√	√	√	√	—	X	X	√	√	√	X	X

(Wu et al., 2016 (A))	Bilinear pairing, Authenticated key Exchange	Secure anonymous authentication for WBAN, computationally efficient	Vulnerable to client impersonation attack	Trusted servers	√	√	√	√	—	X	X	√	√	√	X	X
(Jiang et al., 2016	Elliptic curve cryptosystem (ECC), bilinear pairing	Secure anonymous authentication for WBAN, computationally efficient	Missing vital security requirements such as revocation	Trusted servers	√	√	√	√	—	X	X	√	√	√	X	X
(Chen et al., 2014)	Symmetric encryption, MAC, RFC 2631	Anywhere anytime access	Missing vital security requirements such as anonymity and unlinkability	Trusted servers	√	√	√	√	—	X	X	√	X	X	X	√
(Sharma & Kalra,	Quantum key distribution	Claimed to resist all attacks, generate keys over distance of 100km of optical fibre	Missing vital security requirements such as revocation	Trusted servers	√	√	√	√	√	√	X	√	√	X	X	X
(Yang, 2015)	ABE, SE, bilinear pairing	Ciphertext retrieval, SE scheme supports fine-grained access control	Missing vital security requirements such as anonymity and unlinkability	Trusted servers	√	√	√	X	X	X	X	√	X	X	X	√

Note: √ = Valid X = Invalid — = Not specified

2.6 Disaster Recovery Plans

The CSPs must establish continuity and recovery plans to ensure that services will remain available and can recover all lost data even after disasters such as floods, earthquakes, or electricity power failures. The data recovery plan may be established solely by CSPs, or in consultation with clients.

Several suggestions have been made to facilitate disaster recovery, and some of these are discussed below.

Sahi et al. presented a disaster recovery plan to ensure the availability of PHRs and HERs in a health cloud environment (Sahi et al., 2016). The authors assume that the cloud storage consists of three or more data centres. Distributing signals called heartbeats are used between data centres and the CSP, in order to keep track of the status of these data centres. Each health record is divided into several parts, and multiple copies of each part are stored in different data centres. In the case of a disaster, the heartbeat from a data centre will stop if the data centre machine is damaged, which alerts the manager. The manager can recover or retrieve the records from the other data centres, with no need to access the damaged record. Finally, the authors point out that the data centres must be physically located in different geographic locations (for example in different countries) to ensure the availability of the data and the services (Sahi et al., 2016).

Another disaster recovery plan has been proposed based on three different techniques: TCP/IP, VM snapshots, and replication (Chang, 2015). The proposed plan claims to achieve 99.94% data recovery in the event of a disaster. The proposed approach was implemented with real data and tests involving backing up all sister site records in London, Southampton, and Leeds. However, the data centres in the proposed approach are not integrated with any existing centres. In addition, all data centres are located within the same geographical area, which could be considered a major drawback.

Gu et al. proposed backup and recovery models for implementing a disaster recovery plan (Gu et al., 2014). In terms of the backup model, clients are provided with accounts with limited rights. The CSP is responsible for sending and receiving data to/from clients. A client is able to request a backup from the CSP within a certain timeframe; the CSP will hold this request, make three copies of the data and store these in different locations. In the recovery model, the client can request a data recovery from the CSP. The CSP can retrieve the data from the stored three copies and send it

back to the client. However, storing the data in full at three different locations can significantly increase the backup data size.

Mansoori et al. presented a disaster recovery plan based on two servers, a local server and a disaster recovery server (Mansoori et al., 2014). The proposed plan considers four scenarios to provide availability and continuity of services. The authors implemented the proposed plan in a university hospital health system to ensure constant access to the picture archiving and communication system (PACS) application and its controlled radiology images. However, the authors did not consider a scenario in which a disaster affects a relatively wide geographic area leading to damage to the backup images.

In the cloud environment, users' data are controlled by the service providers rather than by users themselves. As a result, there is the potential for data leaks, either intentional or accidental, which is unacceptable (Gonzales, et al., 2012; Ma, 2012). Furthermore, data in the cloud are stored in geographically diverse locations. Thus, confidentiality, authentication, and communication between parties becomes an important concern (Hussain & Ashraf, 2014). When users use cloud services, they may not know precisely where their information is held (Gampala et al., 2012). In this situation, it is better for cloud service providers to offer high levels of encryption to secure the confidentiality of data wherever they are stored, and an appropriate key exchange method should be adopted.

Confidentiality is not enough to guarantee cloud computing security. Users want to ensure that their data cannot be modified or compromised by a third party. Cloud service suppliers should also apply methods to guarantee data integrity (Sugumaran et al., 2014). Big data are usually stored in distributed locations, therefore parallel encryption is needed to encrypt the large number of blocks or chunks of data separately, taking much longer when using sequential encryption.

In addition to encryption, availability of service must be ensured, flood attacks need to be detected and prevented, and excessive energy consumption must be reduced. Therefore, the following research questions have been identified:

1. Has a researcher in this area reviewed the major and relevant literature on cloud security and privacy?
2. How can the security of key exchange between parties be hardened, and different types of attacks prevented?

3. How can the performance of the adopted encryption algorithm be improved on top of security assurance?
4. How can security and privacy be preserved in eHealth clouds? How can a client be enabled to connect to the system at any time, even during a disaster?
5. How can DDoS TCP flood attacks be detected and prevented? How can the energy consumption caused by DoS attacks be reduced?

Subsequent chapters (4 to 7) review more literature related to each of the identified gaps.

2.7 Chapter Summary

This chapter reviews security and privacy issues and the state-of-the-art solutions proposed for eHealth clouds in five main directions: security and privacy, security controls, effective encryption, data security requirements, and disaster recovery plans.

This chapter collects, evaluates, and classifies the state-of-the-art eHealth security and privacy schemes. In addition, it covers the most recent studies in this area, discusses drawbacks of the existing proposals to help improve the security and privacy of eHealth clouds, and compares several research works in terms of data security requirements.

This chapter investigated and reviewed the relevant literature on eHealth clouds security and privacy. Research directions were decided, and research questions are listed. So, hypothesis H1 (*Investigation and review of the relevant literature on eHealth cloud security and privacy would facilitate the researcher's decisions on research directions*) is proved and research question 1 is answered.

3

CHAPTER 3

RESEARCH DESIGN

Chapter 2 summarised and analysed security and privacy issues and state-of-the-art solutions proposed for eHealth clouds. An extensive review was conducted and research questions were listed. This chapter introduces different research methods and describes the research design methodology and process used to carry out the research objectives.

3.1 Introduction

Research can be defined as an organised investigation undertaken to obtain new knowledge. The research philosophy refers to the assumptions that are made regarding ontology and epistemology. The technique or procedure that is applied in doing research is called the research method, and the study of research methods to solve research problems is called research methodologies (Edgar & Manz, 2017). To ensure research quality, research methods, methodologies and the research process are discussed in the following sections. Focusing on the actual research work is important but considering research methods and methodologies at the beginning of the process is also vital as they affect the outcome of the research (Håkansson, 2013).

Quantitative and qualitative are the main approaches to research. Researchers must choose one of these, or a combination of both (mixed methods). Quantitative research is utilised to investigate problems by generating numerical information that can be changed into usable statistics. While this research requires experiments and the testing and verifying of hypotheses and the functionalities of a system, the quantitative methods was chosen. “Qualitative research is an approach for exploring and understanding the meaning individuals or groups ascribe to a social or human problem.” (Creswell & Creswell, 2017, p.4). Qualitative methods support research that involves the collection and analysis of behaviours and opinions. To gain a more

complete view of the research subject area, mixed methods (sometimes called triangulation) can be used.

In general, there are four types of research: observational research, theoretical research, experimental research, and applied research (Edgar & Manz, 2017). Research where the researcher tries to observe or understand the ongoing behaviour of a real system, is called observational research. Observational research includes three kinds of studies: exploratory studies, descriptive studies and machine learning. Exploratory studies are conducted for the kind of problems that are not deeply studied and that have not achieved significant performance. Descriptive studies seek to improve performance by observation, description and the analysis of real problems. The third is machine learning, which is most popular where electronic data is the basis of problem analysis. Theoretical research is research that involves theories and/or definitions of how a system behaves and then explores the implications for the definition of that system's behaviour. This research can be in the form of a formal theory or a simulation. In experimental research, a researcher uses experiments to generate evidence that answers the research questions. There are two types of experimental research: hypothesis-based research that tests whether the hypotheses are acceptable or not (called hypothetic-deductive research); and quasi-experimental research which is used when the researcher is involved with independent variables that cannot be randomly allocated. The fourth research type is applied research. This research measures and evaluates how effectively we apply certain knowledge to solve problems. Applied research can be further classified as applied experimentation studies and applied observational studies. As this study uses computer experiments to test and verify the proposed methods, experimental research is an appropriate research method choice.

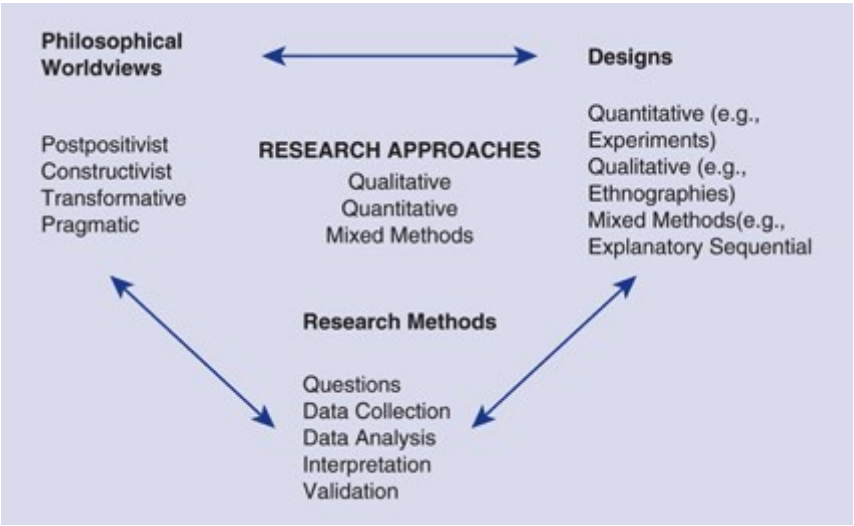


Figure 3.1. A framework for research – philosophy, design, and research methods (Creswell & Creswell, 2017, p.5)

As shown in Figure 3.1, a research framework is an interconnection of philosophy, design, and research methods (Creswell & Creswell, 2017). This research uses positivist assumptions (philosophy) in which outcomes are influenced by causes. In other words, the problems affecting the outcomes were identified and the research questions were then set. According to Creswell and Creswell (2017), "these (positivist) assumptions hold true more for quantitative research than qualitative research" (p.6). To help determine whether the research proposal could affect the research outcomes, an experimental quantitative design was adopted. Finally, pretest and posttest performance measures were included in the research method. These research methods are discussed in Section 3.4 using five different phases.

3.2 The Research Philosophy

The research philosophy adopted for a research project is of the utmost importance. If a phenomenon or an area is to be investigated in a systematic way, there needs to be a coherent research philosophy (Creswell, 2014). The philosophy of the study refers to the assumptions that are made regarding ontology and epistemology.

The concept of ontology refers to the examination of being the ultimate nature of reality and the relationship between the various properties in the world. In this study; the elements in a cloud security system. The researcher adopted a positivist approach to the nature of reality. This is the belief that there exists a fixed and independent reality that is outside the mental experience of the subject (Creswell, 2014). The positivist approach to, and theory of the nature of being, are based on the scientific method. That is, there exists an objective reality that is not dependent on the subject and that is fully independent of the human mind. In this case the positivist approach is applied to cloud computing security. It holds that reality is a set of fixed and regular laws and is highly predictable (Creswell, 2014). The positivist approach holds that reality outside the human mind can nonetheless be known by the subject. The laws that occur in reality can be comprehended by the human mind, which can measure and conceptualise these laws and understand them in an objective way. This means that the researcher can collect objective and verifiable data about information systems such as cloud computing security.

The second important element in the research philosophy is that of the epistemological stance. This is the researcher's theory of knowledge. The researcher adopted the objectivist theory of knowledge. The objectivist theory of knowledge tells that there exists an independent and objective reality. For example, cloud security is a phenomenon that exists in the world. It holds that the human mind can use reason to understand and to establish facts about reality. Reason can be used to understand the laws of nature and those relating to information systems

and cloud security (Williamson & Johanson, 2017). Objectivism holds that we can interact with the physical world, and from these interactions we receive impressions. We can then abstract these impressions to form ideas. For example, conceptualising the functions and the roles of cloud security computing. This means that there is a definite connection between what we experience and our knowledge of what we experience. This theory of knowledge holds that the concepts that we develop are accurate representations of reality, and that we can employ logic to determine the relationship between properties. This theory of knowledge means that the researcher can establish facts and principles in relation to the research questions.

This research project reviewed the literature in the area of eHealth cloud security and privacy. It studies the problems that influenced the security and privacy of eHealth clouds, such as problems related to confidentiality, authentication, and availability. Solutions to such problems may help in securing cloud projects in the health sector. After identifying the problems, five research questions are listed and shown in Table 3.1. The research hypotheses and assumptions were then created to help develop concepts that affect outcomes. Finally, facts were drawn from the outcomes according to the selected theory of knowledge and experience. These facts will potentially improve eHealth clouds and open new research directions to benefit the researcher and other researchers by identifying the first step in other research.

3.3 The Research Design

The main influence on the choice of effective research design is the nature of the research questions. Having identified the research questions, the researcher had to determine the research design most likely to achieve the research goals. Basically, using a research design is very important for any successful research as it can facilitate the extraction of appropriate information and knowledge from the data.

The research design is the methodology, or the strategy, used to perform the research. The research design is the overall methodology of the research that is used to integrate the different methods/parts of the research to implement in a particular case so that it leads to the best possible solution to the research problem. This study adopted an experimental quantitative design which determined whether the proposals could affect the outcomes.

Table 3.1: Mapping of research questions, objectives, methods and validations, and outcomes

	Research questions	Research objectives	Methods and validations	Outcomes
1	Has a researcher in this area reviewed the major and relevant literature on cloud security and privacy?	To investigate and review the state of the art in privacy and security in eHealth clouds, in order to fully explore current research directions	• Summarise and analyse state of the art publications in eHealth cloud security and privacy • Comparison	An extensive review was conducted, and over 100 studies from several peer-reviewed databases, such as IEEE Xplore, were investigated. The selected studies were reviewed and summarised in terms of their benefits and risks. Details given in Chapter 2; the content of this chapter has been submitted to the journal *Computer Science Review*.
2	How can the security of key exchange between parties be hardened, and different types of attacks prevented?	To develop a TPAKE protocol based on the CDH assumption to securely distribute keys between parties, and to protect systems against multiple attacks	• Theoretical proofs • AKE security • Random Oracle Model • Programming • Statistical tests • Prototyping • Lab experiment • Analysis • Results evaluation	The presentation of a TPAKE protocol based on the CDH that securely distributes keys among users and that protects systems against multiple attacks. Details given in Chapter 4; part of the content of this chapter has been accepted by the *International Journal of Communication Networks and Distributed Systems*, and the remainder was presented at the *International Conference on Telecommunications (ICT2015) 22nd: Proceedings of the IEEE.*
3	How can the performance of the adopted encryption algorithm be improved on top of security assurance?	To introduce an efficient hash-based PBC mode of operation to increase the performance of the encryption process while maintaining an assurance of security	• Programming • Prototyping • Lab experiment • Analysis • Results evaluation	A PBC mode of operation was introduced in which blocks of cipher were processed in parallel to ensure both high performance and security. Details given in Chapter 5; part of the content of this chapter was presented at the *4th International Conference on Computer and Communication Systems (ICCCS2018), Japan.* The remainder was presented at the *10th International Conference*

				on Information Technology and Applications (ICITA2015), Sydney.
4	How can security and privacy be preserved in the eHealth cloud? How can a client be enabled to connect to the system at any time, even during a disaster?	To integrate Objectives 2 and 3 into security and privacy-preservation approaches in eHealth clouds, and to ensure the connectivity of users during disasters	• Integration of the PBC and the TPAKE • Availability check • Lab experiment • Analysis • Comparison • Results evaluation	Security and privacy preservation approaches in eHealth clouds that ensure both the privacy and the security of the eHealth cloud. In addition, ensure connectivity for users during a disaster. Details given in Chapter 6; the content of this chapter has been published in the journal *Computers in Biology and Medicine*.
5	How can DDoS TCP flood attacks be detected and prevented? How can the energy consumption caused by DoS attacks be reduced?	To provide a classification-based system for detecting and mitigating DDoS TCP flood attacks in eHealth cloud environments and an energy-efficient TCP DoS attack mitigation method	• Classification • K-Fold-Cross validation • Accuracy, sensitivity and specificity • Kappa coefficient • Filtering • Programming • Lab experiment • Packets analysis • Comparison • Results evaluation	A classification-based security system that helps to detect and mitigate DDoS TCP flood attacks. A method to mitigate DoS attacks in the cloud by reducing excessive energy consumption via limiting the number of packets. Rather than a system shutdown, this method can ensure the availability of service. Details given in Chapter 7; part of the content of this chapter has been published in the journal *IEEE Access*, and the remainder was presented at the *First MoHESR and HCED Iraqi Scholars Conference in Australasia 2017, Melbourne*.

Many research design methodologies that can be used to process a research project. Some examples of these methodologies can be found in relevant studies (Gregg et al., 2001; Hevner et al., 2004; March & Smith, 1995; Nunamaker et al., 1991; Peffers et al., 2008; Purao, 2013).

This thesis adopted the Vaishnavi and Kuechler research methodology (Vaishnavi & Kuechler 2004) which includes five main phases: Awareness of Problem, Suggestion, Development, Evaluation, and Conclusion, as shown in Figure 3.2. This study requires each research question to be tackled following a similar procedure and an individual answer for each research question. The answers are collated to answer the ultimate question: "How to secure clouds?" The Vaishnavi and Kuechler research methodology was chosen because it can be easily applied to the methodology for each research question and eventually obtain the research questions' answers. For example, to answer the second research question; it is necessary to understand the problem, suggest a tentative design, develop the artefact, and then evaluate the performance. The same process is repeated for each research question. The individual conclusions may also be used to help achieve the other objectives and the final conclusion about "securing clouds using cryptography and traffic classification".

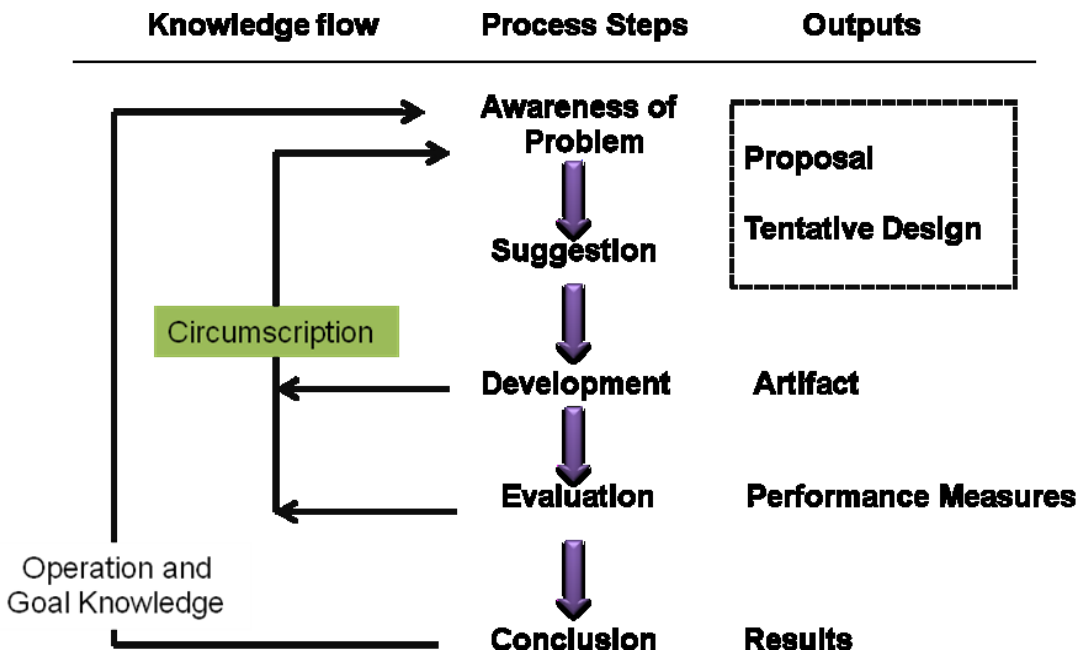


Figure 3.2. The selected research design methodology (Vaishnavi & Kuechler, 2004)

As shown in Figure 3.2, Awareness of Problem is the first phase in this methodology, and it represents the overall picture of the problems and some suggestions for problem solving. Problems normally come from searching in a linked discipline that may also give the opportunity to obtain new results. The output of the Awareness of Problem phase fell into two different parts of this study: First, the introductory chapter (Chapter 1), which highlighted the research questions and which set up the research objectives according to the listed questions and second, the literature review chapter (Chapter 2). In addition to Chapter 2, Chapters 4 to 7 include more literature specific to the chapter but not included in Chapter 2.

The Suggestion phase follows, and uses information from the Awareness of Problem phase to draw a tentative design for the research question. The Development phase is the implementation of the tentative design. Various implementation techniques may be used depending on the artefact to be created. Required software and tools to be used must be identified in the Development phase. In the Evaluation phase, the artefact obtained from the Development phase is tested and evaluated against the hypotheses. Evaluation results may generate a list of future work that can be used as input into another round of Suggestions. Conclusion is the last phase of the research design. This phase provides answers to the research questions by accepting or rejecting the hypotheses. The results and knowledge gained can be repeatedly applied to further research.

Problems are identified in Chapter 1. Chapter 2 provides an overview of cloud computing, in terms of its security and privacy, and some necessary background knowledge. Each subsequent chapter is offered as a cycle of its own. Chapter 8 concludes the research.

Table 3.1 maps the five research questions with their corresponding research objectives, methods and validations, and outcomes to provide an overall picture of the study.

3.4 The Research Design Process

This study uses pretest and posttest performance measures as research methods. These research methods are explained in this section using five different phases. Starting with the Awareness of Problem Phase, the overall research process is shown in Figure 3.2.

3.4.1 The Awareness of Problem Phase

The literature related to the scope of this study was reviewed and gaps identified. Factors influencing the outcomes of eHealth security and privacy are summarised as follows. Clouds are used by millions of people around the world. Cloud services give users the opportunity to store data in the cloud for easy access, anytime and anywhere (Carroll et al., 2011). However, the use of a cloud poses many security and privacy problems. Both security and privacy are vital for data distribution in the cloud (Chen & Zhao, 2012). In the cloud environment, a user's data are controlled by the service providers rather than by the users themselves. There is, therefore, the potential for data leaks (either intentionally or accidentally), which is unacceptable (Gonzales, et al., 2012; Ma, 2012). Furthermore, the data in the cloud are stored in geographically diverse locations. Thus, confidentiality, authentication, and communication between parties become important concerns (Hussain & Ashraf, 2014). When users use cloud services, they may not know precisely where their information is held (Gampala et al., 2012). In this situation, it is better for cloud service providers to offer high levels of encryption to secure the confidentiality of the data wherever they are stored, and an appropriate key exchange method should be adopted.

Confidentiality is not enough to guarantee cloud computing security. Users want to ensure that their data cannot be modified or compromised by a third party. Cloud service suppliers should also apply methods to guarantee data integrity (Sugumaran et al., 2014). Big data are usually stored in distributed locations, therefore parallel encryption is needed to encrypt the large number of blocks or chunks of data separately, taking much longer when using sequential encryption.

In addition to encryption, availability of service must be ensured; flood attacks must be detected and prevented, and excessive energy consumption must be reduced.

With the problems identified, tentative research designs can now be suggested.

3.4.2 The Suggestion Phase

Data security and privacy are sensitive and critical, and play a vital role in the information technology world (Ritchey et al., 2013). Cryptography can be used effectively to support and enhance data security and users' privacy in cloud computing (Jaber & Zalkipli, 2013). This study is focused on security, privacy, prevention and mitigation of attacks on clouds such as eHealth clouds using encryption and other techniques. As a result, the following is suggested: analyse and summarise the relevant

literature; propose a secure key distribution scheme; propose a high performance block cipher mode of operation; integrate the proposed key distribution scheme with the block cipher mode to preserve security and privacy in eHealth clouds; use a classification-based security system that can detect and mitigate DDoS TCP flood attacks and finally, use a method to mitigate DoS attacks in the cloud to reduce excessive energy consumption. This study aims to fulfil these suggestions by developing new approaches to preserving the security and privacy of eHealth clouds, and by ensuring the continuous availability of data by establishing a disaster recovery plan and preventing flood attacks.

3.4.3 The Development Phase

As mentioned above, Development is the implementation of the tentative design. Various implementation techniques are used depending on the artefact to be created. Software development and a package of tools are needed to begin the Development phase.

Chapter 4 answers Research Question 2 by achieving Objective 2. A TPAKE protocol based on the CDH assumption is developed using the Geffe generator, encryption, and a hash function. This TPAKE protocol is implemented using Microsoft Visual Studio 2010, the high level language C#, and some HTML code.

Chapter 5 answers Research Question 3 by achieving Objective 3. A block cipher mode of operation is developed to ensure both high performance and security. As an implementation, Microsoft Visual Studio 2010 and Dot Net Framework 4.5 are used to implement two console applications using C#. Microsoft Visual Studio 2010 is used for the execution of the CBC mode and Dot Net Framework 4.5 for the execution of the PBC mode. These applications are then tested on a simulated network with eight virtual machines that are created on a single machine that has Intel(R) Core(TM) i7-2600 CPU (8 CPUs) and 16GB RAM.

Chapter 6 answers Research Question 4 by achieving Objective 4. Two approaches are proposed based on the outcome of Chapter 4 (TPAKE protocol) and Chapter 5 (PBC mode) to preserve the security and privacy of eHealth clouds. A disaster recovery plan that ensures connectivity for users during disasters is also designed in the form of a flow chart.

Chapter 7 answers Research Question 5 by achieving Objective 5. A virtual network is developed using a virtual box. Then DDoS attacks are launched to attack

the victim network. Packets are captured using Wireshark Network Analyser 2.0.0 (Wireshark, 2017). An algorithm is developed using MATLAB R2015b to classify network packets using four different classifiers (Least Squares Support Vector Machine, Naïve Bayes, K-nearest, and Multilayer perceptron) to detect and prevent the DDoS attacks. In addition, an energy efficient TCP DoS attacks mitigation method is developed based on a packet filtering method to reduce excessive energy consumption by limiting the number of packets. Rather than system shutdown, this method ensures continued availability of the service.

3.4.4 The Evaluation Phase

This phase describes how the proposed methods can be evaluated. The Evaluation phase measures the performance of the developed methods and concludes with their outcomes. The Evaluation phase can also be used as feedback for another round of Awareness of Problem.

In Chapter 4, the presented TPAKE protocol is followed by theoretical proof and a practical demonstration. The proof section seeks to prove the security and correctness of the proposed protocol under the AKE security protocol. The TPAKE security protocol is then tested against relevant attacks. Having evaluated the protocol's ability to resist multiple attacks, the developed protocol is compared with related protocols. In the practical demonstration section, the protocol is demonstrated to show how it might look and why attackers cannot intercept the communication channels. Statistical tests also used as part of the evaluation process.

In Chapter 5, the PBC mode is tested and compared with the CBC mode in three different scenarios. First, a single data file is used as an input for the PBC mode that uses only one process. In the second scenario, the data file is split manually into data blocks and used as input for the PBC mode using multiple processes. Finally, a single data file is used as inputs for the PBC mode using multiple processes. In the last scenario, the PBC mode automatically splits the data file, processes the blocks, and re-combines encrypted blocks into a single encrypted data file. The output of these three scenarios is analysed and compared to show how the PBC performs when compared with the CBC.

In Chapter 6, security and privacy approaches are evaluated under five factors (security, privacy, revocation, break-glass, and the disaster recovery plan), and then compared with relevant approaches.

In Chapter 7, the performance of the CS_DDoS method is evaluated using the four classifiers: LS-SVM, Naïve Bayes, K-nearest, and Multilayer perceptron. Different training data sizes (window sizes) and different thresholds are used in the experiments. Algorithm 1 is applied to the training data for all the classifiers. To choose the most suitable classifier for the proposed method, the CS_DDoS system is evaluated in terms of accuracy, sensitivity (detection rate) and specificity (false alarm rate). The descriptive statistic Kappa coefficient is also used. The K-Fold-Cross validation is adopted for the validation and for conducting a performance comparison of the four predictive modelling algorithms used in CS_DDoS. The four algorithms are then compared for their prediction results. Sequence diagrams are also used as part of the evaluation process. Furthermore, the energy efficient TCP DoS attacks mitigation method is evaluated in the lab using another network. The Wireshark is used to capture the packets before they are analysed both logically and theoretically.

3.4.5 The Conclusion Phase

The Conclusion phase draws conclusions about the research questions from the results of the Evaluation phase. The results and knowledge gained from this phase can be repeatedly applied to further research cycles to produce research questions, and/or to provide answers to the research questions by accepting or rejecting hypotheses.

3.5 Chapter Summary

This chapter described the research philosophy and research design methodologies. The Vaishnavi and Kuechler research methodology was selected, and its five phases were described to provide an overall picture of the study's research design. Positivist assumptions were also adopted the research philosophy.

The following chapters present the Development and Evaluation phases for each of the four research objectives; Objective 2 to Objective 5. The next chapter begins with the second objective: "To develop a TPAKE protocol based on the CDH assumption to distribute keys securely between parties, and to protect systems against multiple attacks." In addition, the literature review for each particular objective continues in each one of the chapters. However, Objective 1 is ongoing and is achieved only after completing all of the other objectives.

4

CHAPTER 4

AUTHENTICATED KEY EXCHANGE PROTOCOL

This chapter presents a Three-party Password-based Authenticated Key Exchange (TPAKE) protocol that can be used to securely distribute keys to users and protect systems against multiple attacks. The proposed TPAKE protocol shares no plaintext data; data shared between the parties are either hashed or encrypted. Using the Random Oracle Model (ROM), the security of the proposed TPAKE protocol is formally proven under the Computational Diffie-Hellman (CDH) assumption. The Geffe binary sequence generator and several statistical tests are used in this protocol.

4.1 Introducing Password Authenticated Key Exchange Protocols

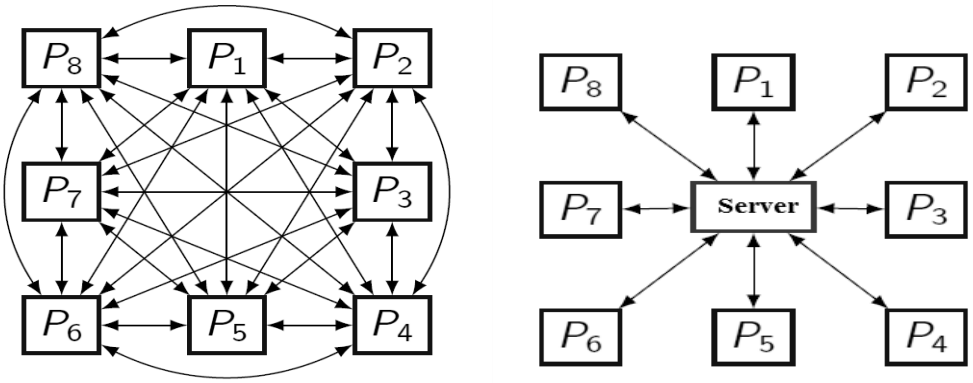
Password Authenticated Key Exchange (PAKE) protocols (Katz and Vaikuntanathan, 2013; Lee et al., 2013; Farash and Attari, 2014 (A); Farash and Attari, 2014 (B)) are user-friendly solutions that ensure the security of the session key for a cryptosystem. This gives end-users the freedom to choose their own password freely without machine intervention. The PAKE protocols were initially introduced in the form of a two-party protocol (2PAKE) (Jiang et al., 2013; Tang et al., 2013; Farash and Attari, 2013). However, 2PAKE protocols are impractical for high numbers of users in a large network since each user needs to recall numerous different passwords for the many different users with whom they communicate (Farash and Attari, 2014 (B)). In other words, in a group of m users, there are m (m-1)/2 user pairs. Its management of the passwords is O (m^2) which becomes impractical if m >>2.

3PAKE protocols were proposed to overcome the problems arising in 2PAKE protocols (Amin and Biswas, 2015; Deebak et al., 2015; Li et al., 2015). In the 3PAKE

protocols, two users typically communicate with each other through a trusted server to establish a key in a secure manner. Unlike the 2PAKE protocols, each user needs to remember only a single password to commence key sharing with partner through a server. Figure 4.1 illustrates 2PAKE and 3PAKE protocols.

Normally, users choose a natural word from a language or a simple phrase as a password, rather than a long random string. However, these kinds of passwords are vulnerable to password guessing attacks (Bellare and Rogaway, 1995) such as offline and online dictionary attacks (Lee et al., 2013). It is, therefore, desirable to use a confidential password authenticated protocol that can prevent password attacks.

Whitfield Diffie and Martin Hellman presented the Diffie-Hellman key exchange protocol in 1976 (Diffie and Hellman, 1976), and this protocol is now widely used for secure key exchange. The process underlying this protocol supposes that Alice and Bob have different private keys. They need to agree upon two relatively prime numbers p, g before each of them can use this information to calculate the public keys. Following agreement, they share their public keys with each other, and each uses these in conjunction with the private key, p and g, to obtain the same shared key. As a result, both Alice and Bob can obtain the shared key without sending their private keys through the channel (Kumar et al., 2012).



(a) 2PAKE protocol (b) 3PAKE protocol

Figure 4.1 The 2PAKE and 3PAKE protocols

Diffie-Hellman Key Exchange (DHKE) is one of the best protocols for the safe exchange of keys. However, this algorithm is vulnerable to Man-In-The-Middle attack (MITM) in which Eve can attack (actively, i.e. she can modify) all communications between Alice and Bob (Ibrahem, 2012).

4.2 Related Work

The literature related to the PAKE protocols and their associated issues is reviewed in this section.

The first two-party protocol was proposed by Bellovin and Merritt (1992). Using this protocol, two users can negotiate the session key using their passwords via public or insecure communication channels. However, in a large network, the management of passwords becomes challenging. If we assume that the network has *m* users and adopt Bellovin and Merritt's protocol to exchange a session key between two of these users, there are *m(m_1)/2* passwords to be shared, as shown in Figure 4.1-a. These passwords need to be kept safely (Bellare and Rogaway, 1995). Hence, many researchers have shifted their attention to 3PAKE protocols in which a trusted server serves as a hub for users, ensuring that each user needs to manage only his/her own password for the server, as shown in Figure 4.1-b. 3PAKE protocols are more scalable to large networks than the 2PAKE protocols.

Although many well-designed 3PAKE protocols have been proposed, cryptanalysis has revealed many issues and vulnerabilities of 3PAKE to various attacks.

Yang et al. proposed a 3PAKE protocol using two servers as intermediate agents. One of these servers connects directly to the users, while the other stays disconnected from the users. These authors claim that their protocol had a number of strong security features that secure each session key against various attacks, including offline dictionary attacks (Yang et al., 2006). However, Amin and Biswas found that this protocol is vulnerable to numerous security issues, such as insider attacks (Nose, 2011), offline password guessing attacks, and replay attacks (Amin and Biswas, 2015).

Lu and Cao (2007) proposed a simple 3PAKE protocol. They claimed that since the protocol did not require the public key for a server, it could prevent many known attacks. However, Guo et al. (2008) showed that this protocol is vulnerable to both MITM attacks which could possibly expose authenticated information to an attacker, and online dictionary attacks.

Another simple 3PAKE protocol that works without the public key for a server was proposed by Huang (2009). Huang claimed that his protocol was secure against various types of attacks and was also more efficient than many other 3PAKE protocols, but Yoon and Yoo (2011) showed that this protocol is vulnerable to many attacks such as online and offline password guessing attacks.

Wen et al. (2005) proposed a new, provably secure 3PAKE protocol. The authors claimed that this protocol had a security proof using a formal model and adversary capabilities. However, Nam et al. (Nam et al., 2007) pointed out that this proposed 3PAKE protocol was absolutely insecure and that the security proof was incorrect, giving a full analysis of the weaknesses in this protocol and its proof.

A communication-efficient 3PAKE protocol was also proposed by Chang et al. as mentioned in (Bellare and Rogaway, 1995). This protocol requires neither the server's public key nor a symmetric encryption scheme and has a security proof based on the computational Diffie-Hellman assumption. Although the authors claimed that their protocol was more practical than other 3PAKE protocols, Wu et al. demonstrated that it was vulnerable to partition attacks, in which attackers could guess the real password offline (Wu et al., 2012).

The weaknesses of these 3PAKE protocols have motivated the researcher to design a TPAKE protocol that can overcome the issues described above.

4.3 Preliminaries

23.9% of the 14.3 million HTTPS servers currently support the Diffie-Hellman protocol (Adrian et al., 2015), which is based on the Diffie-Hellman Problem (DHP). In this chapter, the proposed TPAKE protocol is also based on the DHP. The hardness of the traditional DHP is based on the Discrete Logarithm Problem (DLP).

Definition 1 (DLP): The DLP is the problem of determining an integer χ, where $1 \leq \chi \leq p-1$ (p is prime number), such that $\alpha^x \equiv \gamma$, where both the primitive element α and another element γ are elements in $\mathbb{Z}_p^*$ ($\alpha, \gamma \in \mathbb{Z}_p^*$, $\mathbb{Z}_p^*$ is a finite cyclic group).

The only known way to solve the DHP is to compute $\chi = log_\alpha \gamma$ or α^{xy}, where *x, y* are randomly picked from $\mathbb{Z}_p^*$. This computation has been shown to be infeasible (Chung and Ku, 2008). Thus, to prevent attacks we need to choose a large prime number *p*.

The proposed protocol adopts a well-known complexity assumption known as the Computational Diffie-Hellman (CDH) assumption (Diffie and Hellman, 1976; Chevalier et al., 2008).

Definition 2 (CDH assumption): To ensure that the DLP defined in $\mathbb{Z}_p^*$ is hardened to attack, *p* should be a large prime number. Let $\mathbb{G} \subseteq \mathbb{Z}_p^*$ be a finite cyclic group of prime

order q with generator g, where $p = 2q+1$. Given (g, p), $A = g^a$, $B = g^b$ (public keys), where a, b (private keys) are randomly picked from $\mathbb{Z}_p^*$ *by Alice* ($\hat{A}$) and *Bob* ($\hat{B}$). $\hat{A}$ and $\hat{B}$ represent the initiator client and the responder client, respectively, of a key exchange protocol run. We consider that the CDH assumption holds for $\mathbb{G}$ if the CDH attacker β is given a challenge $\psi = (g^a, g^b)$, in order to compute g^{ab}. The probability of β successfully retrieving g^{ab} in time t is given by:

$$suc_G^{CDH}(t) = \mathbb{P}_{ab}[\beta(g^a, g^b) = g^{ab}] \leq \varepsilon$$

where ε is negligible.

Definition 3 (Negligible function): A function $\varepsilon: \mathbb{Z}^+ \longrightarrow \mathbb{R}^+$ is negligible $\Leftrightarrow \forall\, c > 0, \exists \lambda_0 > 0$, such that $\forall \lambda \geq \lambda_0, \varepsilon(\lambda) \geq \lambda^{-c}$.

In other words, a function ε is negligible if it approaches zero faster than any inverse polynomial (Guo et al., 2011; Cash et al., 2009).

Note that modulo operations (*mod p*) in this chapter are omitted for clarity. All operations are carried out under modulo p.

Definition 4 (Encryption scheme): $\Gamma\{Enc, Dec\}$ is a symmetric encryption/decryption scheme that involves two algorithms (Yi et al., 2013):

- Encryption: $Enc_k(m) \rightarrow c$, where k is the session key, m is the entered plaintext message, and c is the resulting ciphertext.
- Decryption: $Dec_k(c) \rightarrow m$, where k is the session key, c is the entered ciphertext, and m is the resulting plaintext.

Subsequent sections present the proposed TPAKE protocol followed by the theoretical proof and a practical demonstration of how the protocol works. The theoretical proof part proves the security and correctness of the proposed protocol. TPAKE protocol security against various attacks is then discussed, followed by the evaluation of the ability of this protocol to resist multiple attacks and a comparison with related protocols. For the demonstration part, an instance of the protocol is implemented and why attackers cannot intercept the communication channels is highlighted.

The next section presents the proposed TPAKE protocol, based on the CDH assumption.

4.4 The Proposed TPAKE Protocol

Suppose that $\hat{A}$ and $\hat{B}$ wish to exchange an agreed session key *SK* through a trusted server *S*. Agreement is reached in two rounds, as described below:

Round 1. Private Key setup:

In this round, users $\hat{A}$ and $\hat{B}$, set up their private keys *a* and *b* based on private information, such as passwords PW_U, the Geffe generator, and statistical tests.

The process of using the Geffe generator to generate a private key from the user's information is shown in Figure 4.2.

Note: *S/U* means that both the server and the users are carrying out the same process.

Step 1. *S/U:* $(PW_U) \xrightarrow{convert} (PW_U^{base2})$

The user's password PW_U is converted from a printable string to a binary PW_U^{base2}.

Step 2. *S/U*: $(PW_U^{base2}) \xrightarrow{Split} (L_1, L_2, L_3)$

PW_U^{base2} is split into three sequences. The lengths of these sequences must be relatively prime, for example, a 64 bit PW_U^{base2} is divided into sequences of length L_1=20, L_2=21 and L_3=23. Since L_1, L_2, and L_3 are relatively prime, the Greatest Common Divisor (GCD) of $(L_1, L_2, L_3) = 1$, and these form legitimate input for the Geffe generator.

Step 3. *S/U:* $(L_1, L_2, L_3) \xrightarrow{feed\ Geffe\ generator} (seq^{base2})$

The Geffe generator is used to generate a new binary pseudorandom sequence seq^{base2}. The Geffe generator has three Linear Feedback Shift Registers (LFSRs) which use L_1, L_2, L_3, as their initial values. The feedback functions of these LFSRs are assumed to be $K_1 = S_{10}+S_{19}$, $K_2 = S_{10}+S_{20}$, $K_3 = S_{10}+S_{22}$. The outputs of the LFSRs, K_1, K_2, and $K3$, have length 2^L-1 without repetitions, since L_1=20, L_2=21 and L_3=23. This is a valid choice as the pseudo random sequence passed the frequency test, serial test and Poker test in Section 4.5.2.3.2. According to Table 4.3, the lengths of K_1=1048575, K_2=2097151 and K_3=8388607. The Geffe generator then processes K_1, K_2, and K_3, as shown below:

$Z_1 = K_1 \wedge K_2$

$Z_2 = \neg K_2 \wedge K_3$

$seq^{base2} = Z_1 \oplus Z_2$

The resulting seq^{base2} has a length of $(2^{L1}-1)$ $(2^{L2}-1)$ $(2^{L3}-1)$ = *18446715486418763775* without repetitions. As mentioned in (Khader and Lai, 2015), the best length to use from seq^{base2} is only the first 256 bits, in order to reduce time consumption. The Geffe generator is explained briefly in Section 4.4.1.

Step 4. *S/U:* $(seq^{base2}) \xrightarrow{tests}$ (test, result)

a. Success; continue

b. Failure; go to Step 1.

In this step, three types of randomness test are used to check whether the sequence has sufficient randomness. If all the tests are passed successfully, the next step is carried out. If any of these tests fail, the calculation is aborted and the process restarted. The randomness tests are briefly explained in Section 4.4.2.

Step 5. *S/U:* $(seq^{base2}) \xrightarrow{convert} (seq^{base10})$

We now convert the 256 bits from binary to decimal using the ASCII code table, and then extract only eight digits from seq^{base10}, following (Khader and Lai, 2015).

Step 6. *S/U:* $(seq^{base10}) \xrightarrow{obtained}$ (*a, b)* private keys, where $a, b \in \mathbb{Z}_p^*$

We compute *a, b*, which represent the private keys of $\hat{A}$ and $\hat{B}$, respectively.

After this step, $\hat{A}$ and $\hat{B}$ have obtained their private keys, *a* and *b,* using their passwords. In addition to the passwords, the server *S* has a copy of private keys *a* and *b.*

Step 7. *S:* $(PW_U) \xrightarrow{hash \| salt} (\hbar(PW_U) \| salt)$

Server *S* hashes the PW_U to $\hbar(PW_U)$ for use as a checksum, in order to ensure the identity of the initiator or the responder in the subsequent key exchange communications. It then concatenates $\hbar(PW_U)$ with the *salt* (a random number used to add complexity and security) and stores it in a hash table.

Step 8. *S:* $(a, b) \xrightarrow{encrypt\ using\ admin\ key \| salt} (Enc(a, b) \| salt)$

S encrypts *a* and *b* using the admin key *AK*, and then concatenates the ciphertext of *a* and *b* with *salt*.

$$C_a = Enc_{AK}(a)$$
$$C_{as} = C_a \| salt$$
$$C_b = Enc_{AK}(b)$$

$C_{bs} = C_b \parallel salt$

Step 9. End. *a* and *b* were successfully calculated.

Round 2. Session key negotiation:

In this round, $\hat{A}$ and $\hat{B}$ begin negotiation about the *SK* through the trusted server *S*. The process of negotiating the session key is illustrated in Figure 4.3.

Step 1. Set counter *i=1*

Step 2. *S: (p, g, a, b)* $\xrightarrow{generate\ public\ keys}$ *(A, B)*

S generates the public keys, *A* and *B* of $\hat{A}$ and $\hat{B}$ as follows (Kumar et al., 2012; Farash and Attari, 2013):

$$A = g^a$$
$$B = g^b$$

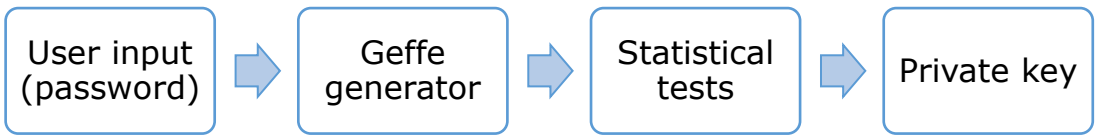


Figure 4.2 Private Key setup

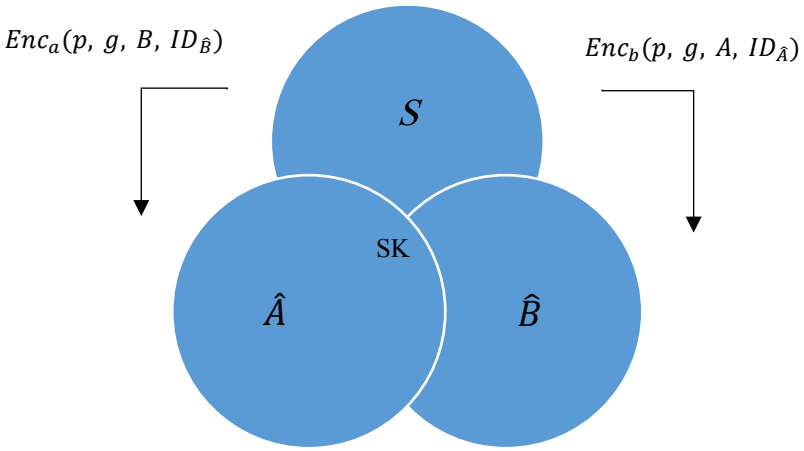


Figure 4.3 Session key negotiation

Step 3. *S: (p, g, A|B, ID$_U$)* $\xrightarrow{encrypt\ \&\ sends}$ *U* $(C_{\hat{A}Info}, C_{\hat{B}Info})$

ID$_U$ is the user's identification name *ID*. *S* sends the following encrypted information packages to $\hat{A}$ and $\hat{B}$:

- *S* to $\hat{A}$

$$C_{\hat{A}Info} = Enc_a(p, g, B, ID_{\hat{B}})$$

- *S* to $\hat{B}$

$$C_{\hat{B}Info} = Enc_b(p, g, A, ID_{\hat{A}})$$

Step 4. *U:* $(C_{\hat{A}Info}, C_{\hat{B}Info})$ $\xrightarrow{decrypt}$ *(p, g, A|B, ID$_U$)*

Upon receiving $C_{\hat{A}Info}$ and $C_{\hat{B}Info}$, $\hat{A}$ and $\hat{B}$ use their private keys *a* and *b* to decrypt the packages as follows:

$$\hat{A}Info = Dec_a(p, g, A, ID_{\hat{B}})$$
$$\hat{B}Info = Dec_b(p, g, B, ID_{\hat{A}})$$

Step 5. *S/U: (p, g, A|B)* $\xrightarrow{compute}$ *(SK)*

In this step *S,* $\hat{A}$ and $\hat{B}$ compute the session key *SK,* as follows (Kumar et al., 2012; Li, 2010):

- *S* & $\hat{A}$

$$SK_{\hat{A}} = B^a$$

- *S* & $\hat{B}$

$$SK_{\hat{B}} = A^b$$
$$where\ SK_{\hat{A}} = SK_{\hat{B}}$$

Step 6. *S/U: (SK)* $\xrightarrow{hash\ and\ match\ the\ SKs}$ $(\hbar(SK_U), h(SK_U))$

a. True; continue
b. False; go to Step 8.

S, $\hat{A}$ and $\hat{B}$ hash the generated *SK*s, send these hashes to each other, and match the checksums received from others to ensure the correctness of the session key. In other words, this protocol provides non-repudiation property as it can identify the sender and receiver from their user information. At this stage, a time stamp can be used to identify replay attack.

Step 7. SK is ready; go to Step 10.

Step 8. If *i*=<*3* then *i*++ and go to Step 2; else go to Step 9.

The user has three attempts to generate the *SK* if this fails after three attempts, the user's account will be locked for 24 hours and a warning message will be sent.

Step 9. Lock account for 24 hours and send a warning message: "Your data was compromised and your account will be locked for 24 hours".
Step 10. End.

A detailed practical demonstration is presented in Section 4.5.2.3.

4.4.1 The Geffe Generator

As shown in Figure 4.2, the Geffe generator is used to generate a pseudo-random sequence with a balanced distribution of zeros and ones (Wei, 2006) from the user input. This sequence will be used as a user private key if it passes the statistical tests for randomness. The process of the Geffe generator is illustrated in Figure 4.4.

4.4.2 Statistical Tests

Random sequences derived from user inputs using the Geffe generator are tested using the frequency, serial and poker tests (Shehata et al., 2003) to ensure their randomness. Once a sequence passes these statistical tests, it will be converted to a user private key.

The frequency test checks the distributions of zeros (n_0) and ones (n_1) in the sequence (n). The serial test checks the distribution of two-digit patterns (n_{00}, n_{01}, n_{10}, and n_{11}), and the poker test checks the distribution of patterns with an arbitrary length. The statistical formulas used are:

Frequency test: $$X_1 = \frac{(n_0 - n_1)^2}{n} \quad (4.1)$$

Serial test: $$X_2 = \frac{4}{n-1}\sum_{i=0}^{1}\sum_{j=0}^{1}(n_{ij})^2 - \frac{2}{n}\sum_{i=0}^{1}(n_i)^2 + 1 \quad (4.2)$$

Poker test: $$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} {n_i}^2\right) - k \quad (4.3)$$

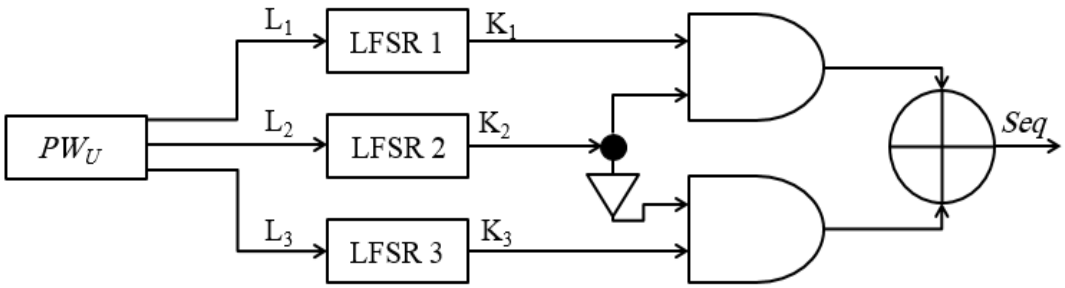


Figure 4.4 The Geffe generator

The accepted threshold values for these randomness tests are ($X_1 < 3.8415$), ($X_2 < 5.9915$) and ($X_3 < 14.0671$), respectively (Hosseini et al., 2014). The proposed TPAKE protocol is shown in Table 4.1.

4.5 Security Analyses

Typically, a security prove investigates the communication of many parties. One of these parties is an attacker β. In a traditional prove, most of the users are authentic, meaning they will work exactly as defined in the protocol. However, β may do anything she likes to break the protocol.

The claim that the TPAKE protocol is provably secure, means that the key exchange remains confidential under certain assumptions. This section proves that the proposed protocol is secure under the CDH assumption. The CDH assumption is hard to solve, and there is no algorithm that can solve it efficiently (Yao and Zhao, 2014; Yantao and Jianfeng, 2010).

4.5.1 Formal Analysis

This section analyses the security of the TPAKE protocol. Typically, authentication between parties and the setting up of session keys *SK* are essential to the security of an authenticated key exchange (AKE) protocol. This section shows that the TPAKE protocol is secure under the AKE security protocol, using the same approach as in (Bellare et al., 2000). In other words, the proposed TPAKE protocol is secure under the AKE security protocol, in which both communication parties $\hat{A}$ and $\hat{B}$ obtain an authenticated session key *SK* after running the TPAKE protocol. However, no other parties are able to discover any information about this session key *SK*, except the trusted server *S*. Therefore, when the TPAKE protocol achieves an AKE security protocol, this means that it has completed both essential security authentication and the setting up of *SK*. This chapter proves that the proposed TPAKE protocol is secure under the AKE protocol using the ROM (Bellare et al., 2000; Bellare and Rogaway, 1993; Bellare and Rogaway, 1995; Canetti et al., 2004).

The ROM (also called the black box) is a random function introduced by Bellare and Rogaway (1993), and can be a perfect hash function. However, a random function is impractical since it is very large to store and very slow to compute. Bellare and Rogaway (1993) showed that using ROM as a random function is practical for the

purposes of a security proof. As a result, the ROM has been adopted in this chapter to prove the security of the proposed TPAKE protocol.

The formal analysis is discussed in three parts, as follows.

Table 4.1 The proposed TPAKE protocol

	$\hat{A}$	Trusted S	$\hat{B}$
1	Request connection channel with $\hat{B}$ $\xrightarrow{PW_{\hat{A}},\ ID_{\hat{B}}}$	-	Request connection channel with $\hat{A}$ $\xleftarrow{PW_{\hat{B}},\ ID_{\hat{A}}}$
2	Retrieve $a \in \mathbb{Z}_p^*$, a extracted from $PW_{\hat{A}}$	Retrieve $a, b \in \mathbb{Z}_p^*$, a extracted from $PW_{\hat{A}}$ & b extracted from $PW_{\hat{B}}$	Retrieve $b \in \mathbb{Z}_p^*$, b extracted from $PW_{\hat{B}}$
3	-	Store $\hbar(PW_{\hat{A}}) \parallel salt$, $\hbar(PW_{\hat{B}}) \parallel salt$, C_{as} & C_{bs} Where: $C_a = Enc_{AK}(a)$, $C_{as} = C_a \parallel salt$ $C_b = Enc_{AK}(b)$, $C_{bs} = C_b \parallel salt$ & the AK is the admin l	-
4	-	Set counter $i=1$ & Calculate A, B $A = g^a$, $B = g^b$	-
5	-	$\xleftarrow{Enc_a(p,\ g,\ B,\ ID_{\hat{B}})}$ $\xrightarrow{Enc_b(p,\ g,\ A,\ ID_{\hat{A}})}$	-
6	$Dec_a(p, g, B, ID_{\hat{B}})$		$Dec_b(p, g, A, ID_{\hat{A}})$
7	Calculate $SK_{\hat{A}}$ $SK_{\hat{A}} = B^a$	Calculate $SK_{\hat{A}}$, $SK_{\hat{B}}$ $SK_{\hat{A}} = B^a$, $SK_{\hat{B}} = A^b$	Calculate $SK_{\hat{B}}$ $SK_{\hat{B}} = A^b$
8	-	If $SK_{\hat{A}} = SK_{\hat{B}}$ then $\xleftarrow{\hbar(SK_{\hat{A}})}$ $\xrightarrow{\hbar(SK_{\hat{B}})}$	-
9	-	If the matching failed, then $i \leq 3$, i++, else continue	-
10	If $h(SK_{\hat{A}}) = \hbar(SK_{\hat{A}})$ then $SK_{\hat{A}}$ is secure	-	If $h(SK_{\hat{B}}) = \hbar(SK_{\hat{B}})$ then $SK_{\hat{B}}$ is secure

4.5.1.1 Characteristics of Participants

The characteristics of participants are participants and the long-lived keys, the characteristics terms of the adversary are explained below.

***Participants*:** The proposed protocol ($\prod$) is divided into two sets of participants: a client *C* and a trusted server *S*. Two clients $\hat{A} \in C$ and $\hat{B} \in C$ authenticate each other and set up an *SK* via *S* in $\prod$. During the execution of $\prod$, the participants (*C* | *S*) may have many instances (oracles). An instance *i* of a participant *U* is symbolised as $\prod_U^i$.

***Long-lived keys*:** $\hat{A} \in C$ stores her password $PW_{\hat{A}}$ as her long-lived key, $\hat{B} \in C$ stores his password $PW_{\hat{B}}$ as his long-lived key, and *S* stores its admin password *AK* as its long-lived key.

In relation to the adversary's capabilities, it is assumed that β (the adversary) is attacking $\prod$, and that β has full control over all communication channels between participants. β's capabilities are modelled in the following Oracle queries:

- Send $(\prod_U^i, m)$: This query models β sending a message m to $\prod_U^i$. $\prod_U^i$ computes a response using the protocol algorithm and sends the result back to β. Moreover, β can send $(\prod_U^i, start)$ to start a new execution of $\prod_U^i$ with another user
- Execute $(\prod_{\hat{A}}^i, \prod_{\hat{B}}^j, \prod_S^k)$: This query models β gaining an authentic $\prod$ execution among $\prod_{\hat{A}}^i, \prod_{\hat{B}}^j$ and $\prod_S^k$. β should have an authentic execution access. This query deals with dictionary attacks
- Reveal $(\prod_U^i)$: This query models β yielding an *SK* from $\prod_U^i$. If $\prod_U^i$ has accepted the *SK*, it sends the *SK* back to β; otherwise, it sends nothing to β. This query deals with known key attacks
- Corrupt (*U*): This query models β sending a query to *U*, yielding his/her long-lived key. The previous SK must not be revealed after losing the long-lived key. This query demonstrates the concept of perfect forward secrecy
- $\Gamma(\{Enc, Dec\}, k, \{m, c\})$: This query models β accessing the oracle of encryption and decryption Γ set out in Definition 4. This means that when β sends $\Gamma(Enc, k, m)$ to Γ, Γ encrypts m using k and returns c to β. When β sends $\Gamma(Dec, k, c)$ to Γ, Γ will decrypt c using k and returns m to β
- $h(m)$: This query models β obtaining a hash value from this query. If the message m has not been queried before, a completely random hash will be sent back to β and

stored in the hash table with m; otherwise, the previous hash value will be sent to β. The ROM is used as a cryptographic hash function (Bellare and Rogaway, 1993)

- Test $(\prod_U^i)$: This query models β distinguishing between a *SK* and a random string. β can send a single test query to $\prod_U^i$, and a value b of the query will be flipped by $\prod_U^i$. The result returned from $\prod_U^i$ will be conditioned. If b=1, then the query returns the *SK*; otherwise, it returns a random string with the same length as that of the *SK*. The test query is accessible only when $\prod_U^i$ is fresh (the term 'freshness' will be explained in the next section).

4.5.1.2 Definitions

This section defines the terms used in proving the security of $\prod$.

- **Definition 5** (Partnership): $\prod_{\hat{A}}^i$ and $\prod_{\hat{B}}^j$ are defined as partnered when the following conditions are satisfied (Wei, 2009):
    - $\prod_{\hat{A}}^i$ and $\prod_{\hat{B}}^j$ exchange *m* directly
    - $\prod_{\hat{A}}^i$ and $\prod_{\hat{B}}^j$ have the same *SK*
    - No one else besides $\prod_{\hat{A}}^i$ and $\prod_{\hat{B}}^j$ has the *SK*, except *S*

- **Definition 6** (Freshness): $\prod_U^i$ is fresh when the following conditions are satisfied (Guo et al., 2011):
    - $\prod_U^i$ has an accepted *SK*
    - No one in the protocol has been sent a corrupted query before $\prod_U^i$ accepts
    - Neither $\prod_{\hat{A}}^i$ nor $\prod_{\hat{B}}^j$ has been sent a reveal query.

A *SK* is fresh if and only if $\prod_U^i$ is fresh**.**

4.5.1.3 AKE security

In the execution of $\prod$, β wins the game and breaks the security of the AKE of $\prod$ if he creates a fresh instance $\prod_U^i$ from a single test query and the selected bit (*b)* is guessed correctly by $\prod_U^i$ (Bresson et al., 2007; Wen et al., 2016). It is assumed that the guessed bit by β is $\grave{b}$ after making several instance queries. Thus, the probability of β guessing b correctly is denoted by $\mathbb{P}(\grave{b} = b) = 1$, and the probability of β guessing b incorrectly

is denoted by $\mathbb{P}(\grave{b} = b) = 0$. The advantage of β in breaking the security of the AKE in $\prod$, denoted by $adv_{\prod}^{AKE}(\beta)$, is defined as:

$$adv_{\prod}^{AKE}(\beta) = |2\,\mathbb{P}(\grave{b} = b) - 1| \tag{4.4}$$

The TPAKE protocol $\prod$ is secure under the AKE protocol only when $adv_{\prod}^{AKE}(\beta)$ is negligible.

Theorem 1. *Let $adv_{\prod}^{AKE}(\beta)$ be the advantage of β in breaking the security of the AKE security of $\prod$ within a time bound t, and creating the queries s_q (Send), e_q (Execute), and h_q (Hash). Then:*

$$adv_{\prod}^{AKE}(t, s_q, e_q, h_q) \leq \frac{s_q}{n} + h_q \cdot (s_q + e_q) \cdot suc_G^{CDH}(\grave{t}) \tag{4.5}$$

where $\grave{t}$ is the running time of suc_G^{CDH} and *n* is the number of possible passwords.

Proof. Let assume that $\mathbb{P}[x]$ is the probability of β breaking the AKE security of $\prod$. Let $\mathbb{P}[x_1]$ be the probability of β breaking the AKE security of $\prod$ by breaking the password security, and let $\mathbb{P}[x_2]$ be the probability of β breaking the AKE security of $\prod$ without breaking the password security. Then:

$$\mathbb{P}[x] = \mathbb{P}[x_1] + \mathbb{P}[x_2] \tag{4.6}$$

$\mathbb{P}[x]$ is constructed from $\mathbb{P}[x_1]$ and $\mathbb{P}[x_2]$ as follows:

The probability $\mathbb{P}[x_1]$ of β breaking the AKE security of $\prod$ by breaking the password security: β could break the security of the password (either $PW_{\hat{A}}$ or $PW_{\hat{B}}$) in $\prod$ in two ways: online dictionary attacks or offline dictionary attacks. The probability of β breaking the security using an online dictionary attack is denoted as $\mathbb{P}[x_1^{on}]$, and the probability of β breaking the security using an offline dictionary attack as $\mathbb{P}[x_1^{off}]$. Then:

$$\mathbb{P}[x_1] = \mathbb{P}[x_1^{on}] + \mathbb{P}[x_1^{off}] \tag{4.7}$$

The probabilities of online and offline dictionary attacks are analysed below.

Online dictionary attacks: β can check the correctness of a choice of either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ by executing it in Round 1 of the algorithm to retrieve either *a* or *b*. Then, β uses *a* or *b* as a decryption key in Round 2 of the algorithm (Step 4) to retrieve *p, g, A | B* and ID_U. Finally, β uses the obtained information to calculate the *SK* and then sends $\hbar(SK)$ as in Round 2 of the algorithm (Step 6) to check the accuracy of either $PW_{\hat{A}}$ or $PW_{\hat{B}}$. Therefore, $\mathbb{P}[x_1^{on}]$ is limited by s_q and *n* as follows:

$$\mathbb{P}[x_1^{on}] \leq \frac{s_q}{n} \tag{4.8}$$

Offline dictionary attacks: As mentioned in Definition 2, β can break the CDH assumption with probability $\grave{\varepsilon}$ within time bound $\grave{t}$, where $suc_G^{CDH}(\grave{t}) = \grave{\varepsilon}$. In the CDH assumption, β is given a challenge $\psi = (g^a, g^b)$. Since in the proposed protocol, *a* and *b* are driven from $PW_{\hat{A}}$ and $PW_{\hat{B}}$, β then has to attack $PW_{\hat{A}}$ and $PW_{\hat{B}}$. $PW_{\hat{A}}$ and $PW_{\hat{B}}$ are stored as hash values in the hash table in *S*, and are concatenated with *salt*, as in Round 1 of the algorithm (Step 7) $\hbar(PW_{\hat{A}}) \parallel salt$ and $\hbar(PW_{\hat{B}}) \parallel salt$. Therefore, the probability ε_1 that β can correctly pick the hash value $\hbar(PW_{\hat{A}})$ from the possible hash $(\hbar(PW_{\hat{A}}) \parallel salt)$ queries or pick the hash value $\hbar(PW_{\hat{B}})$ from the possible hash $(\hbar(PW_B) \parallel salt)$ queries from the hash table on *S* is:

$$\varepsilon_1 \geq \frac{1}{h_q} \tag{4.9}$$

The probability ε_2 that β correctly picks PW_U to retrieve any user's password is equivalent to the probability of β picking the value *i*:

$$\varepsilon_2 \geq \frac{1}{e_q} \tag{4.10}$$

As a result, the probability $suc_G^{CDH}(\grave{t})$ of β breaking the CDH assumption is equivalent to the probability $\mathbb{P}[\mathrm{x}_1]$ of β breaking the AKE security of $\prod$ by breaking the password security using offline dictionary attacks. This probability is equal to the probability $\mathbb{P}\left[x_1^{off}\right]$ that β correctly picks the password of either $\mathrm{PW}_{\hat{\mathrm{A}}}$ or $\mathrm{PW}_{\hat{\mathrm{B}}}$ using offline dictionary attacks, multiplied by the probability ε_1 that β can correctly pick the hash value $\hbar(PW_{\hat{A}})$ from the possible hash $(\hbar(PW_{\hat{A}}) \parallel salt)$ queries or pick the hash

value $\hbar(PW_{\hat{B}})$ from the possible hash ($\hbar(PW_B) \parallel salt$) queries from the hash table on *S,* multiplied by the probability ε_2 that β correctly picks the PW_U:

$$suc_G^{CDH}(\grave{t}) = \grave{\varepsilon} = \mathbb{P}\left[x_1^{off}\right] \cdot \varepsilon_1 \cdot \varepsilon_2 \geq \mathbb{P}\left[x_1^{off}\right] \cdot \frac{1}{h_q} \cdot \frac{1}{e_q},$$

$$suc_G^{CDH}(\grave{t}) \geq \mathbb{P}\left[x_1^{off}\right] \cdot \frac{1}{h_q} \cdot \frac{1}{e_q},$$

$$\mathbb{P}\left[x_1^{off}\right] \leq suc_G^{CDH}(\grave{t}) \cdot h_q \cdot e_q,$$

$$\because \mathbb{P}[x_1^{on}] \leq \frac{s_q}{n},$$

$$\therefore \mathbb{P}[x_1] = \mathbb{P}[x_1^{on}] + \mathbb{P}\left[x_1^{off}\right] \leq \frac{s_q}{n} + suc_G^{CDH}(\grave{t}) \cdot h_q \cdot e_q \tag{4.11}$$

The probability $\mathbb{P}[x_2]$ *of* β *breaking the AKE security of* $\prod$ *without breaking the password security.* β could break the security of the AKE of $\prod$ without knowing either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ by breaking either *A* or *B* (public keys) or by guessing the private key (either *a* or *b*) by sending fake public key queries to either $\hat{A}$ or $\hat{B}$. Breaking *A* or *B* can lead to a MITM attack (Khader and Lai, 2015). However, β needs to know the hash value of either *A* or *B* in order to attack. The probability ε_3 that β correctly picks a hash value is:

$$\varepsilon_3 \geq \frac{1}{h_q} \tag{4.12}$$

The probability ε_4 that β correctly guesses the private key (either *a* or *b*) by sending fake public key queries to either $\hat{A}$ or $\hat{B}$ is equivalent to the probability of β picking the value *j*:

$$\varepsilon_4 \geq \frac{1}{s_q} \tag{4.13}$$

As a result, the probability $suc_G^{CDH}(\grave{t})$ of β breaking the CDH assumption is equivalent to the probability $\mathbb{P}[x_2]$ that β breaks the AKE security of $\prod$ without breaking the password security, multiplied the probability ε_3 that β knows the hash value of $PW_{\hat{A}}$ or $PW_{\hat{B}}$ in order to generate *a* or *b,* multiplied the probability ε_4 that β

correctly guesses the private key (either *a* or *b*) by sending fake public key queries to either $\hat{A}$ or $\hat{B}$:

$$suc_G^{CDH}(\grave{t}) = \grave{\varepsilon} = \mathbb{P}[\mathrm{x}_2] \cdot \varepsilon_3 \cdot \varepsilon_4 \geq \mathbb{P}[\mathrm{x}_2] \cdot \frac{1}{h_q} \cdot \frac{1}{s_q},$$

$$suc_G^{CDH}(\grave{t}) \geq \mathbb{P}[\mathrm{x}_2] \cdot \frac{1}{h_q} \cdot \frac{1}{s_q},$$

$$\mathbb{P}[\mathrm{x}_2] \leq h_q \cdot s_q \cdot suc_G^{CDH}(\grave{t}),$$

$$\because \mathbb{P}[x] = \mathbb{P}[x_1] + \mathbb{P}[x_2]$$

$$\because \mathbb{P}[x_1] \leq \frac{s_q}{n} + suc_G^{CDH}(\grave{t}) \cdot h_q \cdot e_q$$

$$\therefore \mathbb{P}[x] \leq \frac{s_q}{n} + suc_G^{CDH}(\grave{t}) \cdot h_q \cdot e_q + h_q \cdot s_q \cdot suc_G^{CDH}(\grave{t}) \qquad (4.14)$$

$$\therefore \mathbb{P}[x] = adv_{\prod}^{AKE}(t, s_q, e_q, h_q) \leq \frac{s_q}{n} + h_q \cdot (e_q + s_q) \cdot suc_G^{CDH}(\grave{t}) \quad \blacksquare$$

To sum up, as in Definition 2, the CDH assumption holds for suc_G^{CDH}, and suc_G^{CDH} is less than the negligible ε. Also, $h_q, e_q,$ and s_q cannot be large due to the limited number of attempts and length of time in Round 2. Therefore, $adv_{\prod}^{AKE}$ is negligible and the CDH assumption holds in this case. Hence, $\prod$ is AKE secured.

4.5.2 Discussion

This section discusses the security of the TPAKE protocol against various attacks. The ability of the protocol to resist multiple attacks was evaluated and it was then compared with recently proposed related protocols (Yang et al., 2006; Lu and Cao, 2007; Huang, 2009; Wen et al., 2005; Xie et al., 2017; Amin et al., 2016; Islam, 2016; Farash and Attari, 2014b; Lee et al., 2013). Table 4.2 shows performance comparisons for the proposed protocol and several other related protocols. From Table 4.2, it is clear that the TPAKE is more secure than other protocols as it is protected against multiple attacks and provides more security features. The following six proven propositions ensure the security of the proposed protocol against many attacks, and provide perfect forward secrecy.

4.5.2.1 Propositions

Proposition 1. *The proposed protocol* $\prod$ *can mitigate MITM attacks.*

Proof. The public keys A and B are necessary for establishing a MITM attack. In order to begin attacking protocol $\prod$ using a MITM attack, β needs to know both A and B as well as g and p. However, as shown in Round 2 of the algorithm (Step 3) in $\prod$, all these parameters are encrypted using the user private key (a or b), which is known only by user ($\hat{A}$ or $\hat{B}$) and S. Thus, the proposed protocol $\prod$ can mitigate the MITM attack. ■

Proposition 2. *The proposed protocol* $\prod$ *can mitigate online dictionary attacks.*

Proof. Online dictionary attacks can be mitigated by validating the correctness of the input every time. In this proposed protocol, integrity checks are provided. $\prod$ checks the SK sent by matching the hashes stored on S, and also checks them with the end users $\hat{A}$ and $\hat{B}$. In other words, S, $\hat{A}$ and $\hat{B}$ can validate one another using the stored hash values, as in Round 2 of the algorithm (Step 6). Thus, the proposed protocol $\prod$ can mitigate online dictionary attacks. ■

Proposition 3. *The proposed protocol* $\prod$ *can mitigate offline dictionary attacks.*

Proof. Offline dictionary attacks can be performed by a passive attacker β who may have control over the session between $\hat{A}$ and $\hat{B}$. β tries to guess one of the passwords, either $PW_{\hat{A}}$ or $PW_{\hat{B}}$, in order to compute the SK. However, β cannot check the correctness of the password until the verification process in Round 2 of the algorithm is completed (Step 6). To do so, β needs to know the SK before knowing either $PW_{\hat{A}}$ or $PW_{\hat{B}}$, which is not possible. Thus, the proposed protocol $\prod$ can mitigate an offline dictionary attack. ■

Proposition 4. *The proposed protocol* $\prod$ *can mitigate replay attacks.*

Proof. Replay attacks can be performed by an active attacker β who may compromise the message sent between the parties. It is assumed that β can capture $\hat{A}$'s message and replay the compromised message to $\hat{B}$. Due to the time stamp feature provided by the proposed protocol, $\hat{B}$ can easily detect that the message was compromised by β (Khader and Lai, 2015). Thus, the proposed protocol $\prod$ can mitigate replay attacks. ■

Proposition 5. *The proposed protocol* $\prod$ *can mitigate known key attacks.*

Proof. An attacker β may use a previous SK in order to create known key attacks. However, p and g are changed at each session (Khader and Lai, 2015) therefore the public keys A and B will differ. Consequently, the SK will differ at each session.

Furthermore, the private keys *a* and *b* are hard to retrieve due to the CDH assumption. Thus, the proposed protocol ∏ can mitigate known key attacks. ■

Proposition 6. *The proposed protocol* ∏ *can ensure perfect forward secrecy.*

Proof. We can define perfect forward secrecy as follows: if either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ is exposed by β in order to compute the *SK*, the previous *SK* should remain safe. The proposed protocol can ensure perfect forward secrecy as β cannot find out a previous *SK* even when the current *SK* is known, since all the old *SK*s are stored as hashes. Thus, the proposed protocol ∏ can ensure perfect forward secrecy. ■

4.5.2.2 Comparison

The TPAKE protocol can also be compared with several existing protocols in terms of attacks, perfect forward secrecy, and security proofs. A comparison of these results is shown in Table 4.2.

Table 4.2 Comparison with existing protocols

	Protocols	Attacks					Perfect forward secrecy	Security prove
		MITM	Online dictionary	Offline dictionary	Replay	Known key		
1	Yang et al. (2006)	×	√	×	×	√	×	√
2	Lu and Cao (2007)	×	×	√	√	√	√	×
3	Huang (2009)	√	×	×	√	√	×	×
4	Wen et al. (2005)	×	×	×	√	√	×	×
5	Xie et al. (2017)	×	√	√	×	×	√	√
6	Amin et al. (2016)	√	×	√	√	√	×	√
7	Islam (2016)	√	×	√	√	×	√	√
8	Farash & Attari (2014) (B)	×	√	√	√	√	√	×
9	Lee et al. (2013)	×	√	√	√	×	×	×
10	TPAKE	√	√	√	√	√	√	√

4.5.2.3 Demonstration

Users typically have usernames and passwords, and these are used to connect them with their systems. Passwords comprise a mixed string, that can contain numbers, symbols, lowercase and uppercase letters. These mixtures can be represented in binary

using ASCII code. This can be used to generate a different binary sequence using random number generators and to extract new information that can be helpful to overcome the problem.

The proposed work aims to distribute keys between Alice and Bob without being compromised by Eve. Figure 4.5 briefly explains the way in which Alice and Bob communicate with the server and how the data would be sent through the channels to retrieve the shared key.

Example: Assume that Alice's password contains eight characters (password = abcdefgh (a character string), 64 bits). The binary representation of the ASCII code for "abcdefgh" is:

a=97 (01100001) + b=98 (01100010) + c=99 (011000011) + d=100 (01100100) + e=101 (01100101) + f=102 (01100110) + g=103 (01100111) + h=104 (01101000).

We have then obtained the password in binary digits. The sequence will be:

0110000101100010011000110110010001100101011001100110011101101000 (64 bits)

To make this more secure, a pseudo-random sequence generator is used to generate a longer sequence.

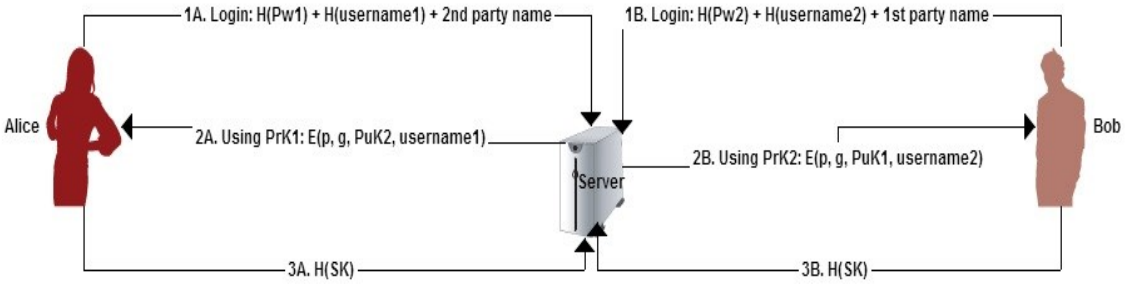


Figure 4.5 Alice and Bob communicate with the server to obtain the shared key

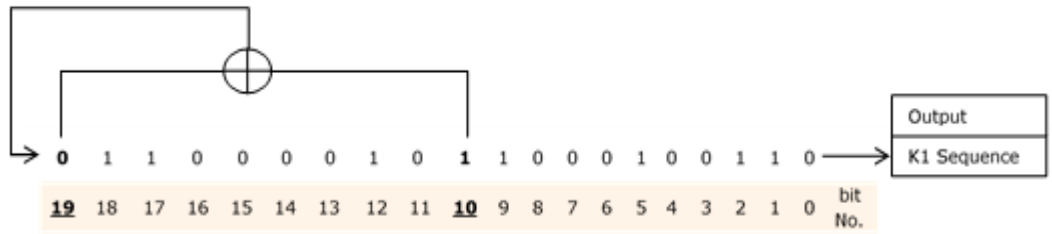


Figure 4.6 LFSR 1

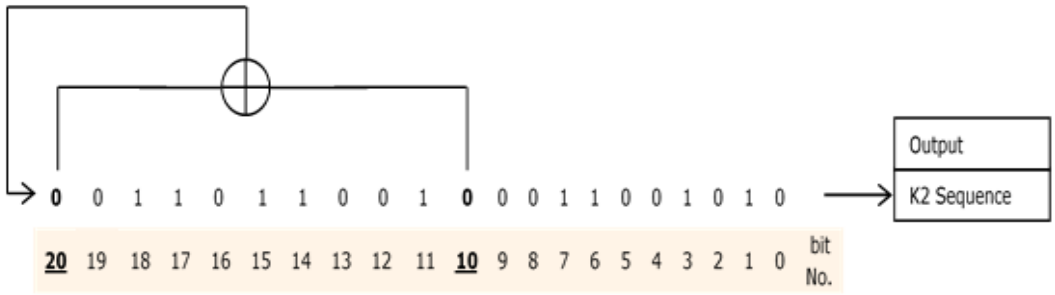


Figure 4.7 LFSR 2

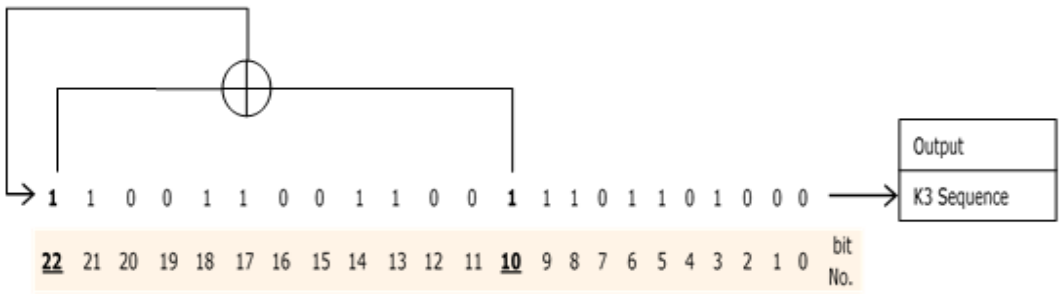


Figure 4.8 LFSR 3

4.5.2.3.1 Generating a random sequence

As described in Section 4.4.1, the Geffe generator uses three registers to generate a random sequence. The Geffe generator was chosen to generate the new sequence as it is in stream cipher cryptography. This generator was chosen for several reasons. First, the complexity of this generator could be superior for a different configuration of the steps. Second, it has certain desirable features. For example, it has a balanced distribution of zeros and ones in its output. It also has the benefit of being useful as a module when constructing similar arrangements. It could play the role of LFSR1 in the same arrangement with similar generators, and this complexity would increase

accordingly (Wei, 2006). The Geffe generator needs three LFSRs, the lengths of which should be relatively prime, meaning that the GCD of the length of the initial values that can be input into the three registers LFSR1 (Len1), LFSR2 (Len2) and LFSR3 (Len3) is one:

$$GCD (Len1, Len2, Len3) = 1 \qquad (4.15)$$

We assume that the first register has 20 bits for input (LFSR1), the second has 21 bits for input (LFSR2), and the last register has 23 bits for input (LFSR3). Together, this gives 64 bits (it is possible to choose any three relatively prime numbers to give 64 bits).

20 bits 21 bits 23 bits
01100001011000100110, 001101100100011001010, 11001100110011101101000

These should be connected as shown in Figure 4.4.

The initial values of these three registers with their feedback functions (K1= S10+S19, K1= S10+S20, K1= S10+S22) are shown in Figures 4.6, 4.7, and 4.8.

As shown in Section 4.4, the feedback functions are: K1= S10+S19, K2= S10+S20 and K3= S10+S22, respectively. The output of these registers is three long binary sequences K1, K2 and K3, each of which have length 2^n-1 without repetition, meaning that K1=1048575, K2=2097151 and K3=8388607. The Geffe generator will then mix these together to generate a single binary sequence S of length $(2^{K1}-1)(2^{K2}-1)(2^{K3}-1)$ = 18446715486418763775 without repetition, as shown in Table 4.3.

Table 4.3 Period of the registers and their feedback functions

Bits	Feedback Function	Period
n		**2^n-1**
20	K1= S_{10}+S_{19}	1048575
21	K2= S_{10}+S_{20}	2097151
23	K3= S_{10}+S_{22}	8388607
64	S = K_1 K_2+K_2 K_3+K_3	**$(2^{K1}-1)(2^{K2}-1)(2^{K3}-1)$** =18446715486418763775

If we generate this extremely large number (*S*), the system will slow down or stop. It is therefore assumed that sequences (*S*) that are 1024, 512, 256 or 128 bits long can be generated, which would be more convenient to deal with. After obtaining the binary sequence from the Geffe generator, three basic types of statistical tests are used to check whether the randomness of the sequence is sufficient. If the test succeeds, we will continue with the next steps; otherwise, the system will ask the user to change the password. Finally, when the tests are complete, which length will give better results (1024, 512, 256 or 128 bits) will be decided.

4.5.2.3.2 Testing the generated sequence

- **Frequency test:** For every binary random sequence, we expect half of the digits to be zeros and the other half ones. The purpose of this test relies on the number of zeros (n0) and ones (n1) in the sequence (n), which is what we need to test. The statistic used is given in Equation 4.1.

  For the sequence to pass this test, the value of X_1 should be less than the acceptance threshold values of the test (X1 < 3.8415) for one degree of freedom (Hosseini et al., 2014).

- **Serial test:** This test depends upon the repetition of (n00, n01, n10, and n11), which denote the numbers (00, 01, 10, and 11) appearing in S, respectively. It is expected that each of these will represent approximately a quarter of n. The statistic used is given in Equation 4.2.

  For the sequence to pass this test, the value of X_2 should be less than the acceptance threshold values of the test (X2 < 5.9915) for two degrees of freedom (Hosseini et al., 2014).

- **Poker test:** This test procedure involves dividing the sequence *S* into a number of blocks (*K*) of length (*M*), and then checking the repetition of these blocks to determine whether they appear approximately the same number of times as would be expected in a random sequence, i.e. the number of blocks $K=n/M$ (without fractions). The statistic used is given in Equation 4.3.

For the sequence to pass this test, the value of X_3 should be less than the acceptance threshold values of the test (X3 < 14.0671), for 2^m-1 degrees of freedom (Hosseini et

al., 2014). The poker test is a generalisation of the frequency test; for *m=1*, the same results as for the frequency test is obtained.

Table 4.4 Statistical results

Length of Pseudo	Statistical test	Value of statistical test	Degree of freedom	Acceptance threshold
1024 bits	Frequency test	0.03515625	1	3.8415
	Serial test	2.04011256720423	2	5.9915
	Poker test	9.19354838709677	7	14.0671
512 bits	Frequency test	0.0703125	1	3.8415
	Serial test	0.452192392367863	2	5.9915
	Poker test	7.2235294117647	7	14.0671
256 bits	Frequency test	1	1	3.8415
	Serial test	1.42745098039217	2	5.9915
	Poker test	7.51764705882353	7	14.0671
128 bits	Frequency test	**5.28125**	1	3.8415
	Serial test	4.56914370078741	2	5.9915
	Poker test	7.90476190476191	7	14.0671

The current work used these three tests to check sequences *S* of length 1024, 512, 256 and 128 bits (for the current password), and the results are shown in Table 4.4.

When the sequence *S* was 128 bits long, it failed the frequency test (the sequence generated by password=abcdefgh did not pass the test). A length for *S* of 256 bits was chosen since this was the shortest successful length that could still ensure that the sequence had good randomness and a small size. As illustrated in Table 4.4, the frequency test increased each time the length decreased, and this was the reason behind a choice of 256 bits rather than 128 bits. The 256-bit sequence obtained from the password is as follows:

0100010011100100011001000001001000000010101000001111111011101010101010100001100010011000100011111001110000000110101010100100001000111111001101100110101000100011010101110010100101101101000010010111000010101101111100100100100101100011110111001110111111000111010

This sequence was derived from Alice's password and passed all the tests. Next, each group of 8 bits is converted into a decimal number using ASCII code, giving the following numbers: 68 228 100 18 5 65 253 213 84 49 49 30 112 26 169 8 252 217 168 141 92 165 180 37 194 183 201 37 143 115 190 58. Each of these is then moduloed by 10 to ensure that all of the numbers can be represented by a single digit (0, 1, 2, 3, 4, 5, 6, 7, 8 or 9). Merging these together generates a single number with 32 digits, 88085533499026982781250743173508. It is then assumed that the user's private key is eight digits long (the private key will be between *2* and $P-2$ (Paar and Pelzl, 2009), where P is provided by the server). The 32-digit number is therefore divided into eight blocks, and one number is taken from each block; these eight digits will be the private key (taking the first number from each block (the choice is random, a different number can be chosen), the private key will be PrK=85422243). At this stage, each user has his/her own private key, and the server stores all of these together in a single table.

Alice and Bob obtain two numbers, g and p, where p is a prime and g (generator) is a primitive root *modulo* p. These two numbers are received from the server with the public key: Alice will receive *p, g*, Bob's public key and her username, and Bob will receive *p, g*, Alice's public key and his username. These will be encrypted using the receiver's private key. As mentioned above, the private key should be less than the prime p (Paar and Pelzl, 2009); this private key with fewer than nine digits satisfies these requirements. For this example, let p=100000007 and g=5. The public key will be *PuK*=15071649. If Alice's password=abcdefgh and Bob's password=12345678 (a character string), then Alice's private key *PrK*=85422243 and Bob's private key *PrK* = 68203955. After combining these with p and g, we get Alice's (*PuK*=15071649) and Bob's (*PuK* = 6629794). As a result, the shared key will be *SK*=68202249 (for both Alice and Bob).

Eve can therefore only obtain hashed and encrypted data, and cannot use these data to intercept the communication channels.

In summary, MITM attacks occur when keys are shared in plaintext. Eve can sit in the middle and pretend she is the intended destination for both Alice and Bob; Alice and Bob have no way of knowing that Eve is there and believe they are communicating directly with each other. The proposed method proves that Alice and Bob can generate their keys without sending them in plaintext, using data obtained from the server that will be controlled by the server. Although some may argue that Eve can compromise all the data on the channels, there are insufficient data for her to obtain the shared key,

and all data are hashed and encrypted. In addition, the private key is never sent to any party. If an attacker steals the private key from the server, this would not be useful, as the entire collection of private keys on the server are encrypted using the admin key. This offers high levels of security and helps prevent MITM attacks in the Diffie-Hellman protocol.

Many applications use Diffie-Hellman as a key exchange mechanism. Examples of these are secure socket layers (SSL), secure shell (SSH) and IP security (IPSec) (Ahmed et al., 2012). The proposed infrastructure can distribute the keys in a secure manner, meaning that applications using Diffie-Hellman as a key exchange mechanism would benefit from the proposed infrastructure.

Source code for an implementation of the TPAKE practical demonstration are shown in Appendix A. Screen shots of sample runs are also provided in the same appendix. Microsoft Visual Studio 2010 was used to develop and run the code.

It is notable that the central server would not use existing identification techniques such as X.509, for several reasons: X.509 certificates require digital signatures for integrity, and thus the same problems arise as in the use of Digital Signature (Toma, 2009). In addition, X.509 certificates experience problems when certificates expire; sometimes it is not clear that a certificate has expired until the website or server goes down, meaning that their use is challenging in practice.

4.6 Chapter Summary

This chapter introduced a TPAKE protocol based on the Computational Diffie-Hellman assumption and the ROM. The proposed protocol performs better than other 3PAKE protocols as it never shares information in plaintext through insecure channels. A security analysis using the ROM shows that the proposed protocol achieves mutual authentication, and a safe and secure session key ensures perfect forward secrecy and can prevent several attacks. Therefore, hypothesis H2 (*Developing a key exchange protocol based on the Computational Diffie-Hellman assumption should secure key distribution between parties and protect systems against multiple attacks*) has been proven in this chapter. TPAKE key exchange protocol has been developed based on the Computational Diffie-Hellman assumption. Analysis and tests show that the TPAKE protocol is a secure key distribution protocol and can protect systems against multiple attacks.

Hardening the security of communications between parties and preventing attacks, such as MITM, is not enough to secure a cryptosystem. Other factors need to be considered, such as enhancing the adopted encryption algorithm. This will be the main focus in Chapter 5.

5

CHAPTER 5

PARALLEL BLOCK CIPHER MODE

Chapter 4 presented an authenticated key exchange protocol TPAKE, which securely distributes keys among users, and protects systems against multiple attacks. This chapter presents a Parallel Block Cipher (PBC) mode in which blocks of cipher can be processed in parallel to ensure high performance on top of security.

5.1 Introducing Block Cipher Modes of Operation

A block cipher is a generic process in which data are handled in fixed sized blocks. It can be used to handle a single chunk as well as a stream of data. Different ways of handling the data blocks will end up in different modes of operation for the block cipher. Five modes of operation for block cipher have been approved (Stallings, 2010) for encryption use: Electronic Codebook (ECB) mode, Cipher Block Chaining (CBC) mode, Cipher Feedback (CFB) mode, Output Feedback (OFB) mode and Counter (CTR) mode.

The Electronic Codebook (ECB) mode is the simplest of the encryption modes. The plaintext is split into blocks, and each block is processed independently (Dworkin, 2001). The mathematical formulae for the ECB are:

$$C_i = E_k(P_i) \tag{5.1}$$

$$P_i = D_k(C_i) \tag{5.2}$$

The Cipher Block Chaining (CBC) mode was developed in 1978 by Ehrsam, Meyer, Smith and Tuchman (Ehrsam et al. , 1978). In the CBC mode, each plaintext block is XORed with the previous ciphertext block before being encrypted. To be probabilistic, an initialisation vector (IV) should be utilised in the first block (Dworkin, 2010 (A,

B)). The mathematical formulae for the CBC are:

$$C_i = E_k(P_i \oplus C_{i-1}) \quad (5.3)$$

$$P_i = D_k(C_i) \oplus C_{i-1} \quad (5.4)$$

where, $C_0 = IV$

The Cipher Feedback (CFB) mode uses the previous ciphertext as input for encryption. The encryption output is XORed with the current plaintext block to produce the ciphertext of the current block. The mathematical formulae for the CFB are:

$$C_i = E_k(C_{i-1}) \oplus P_i \quad (5.5)$$

$$P_i = D_k(C_{i-1}) \oplus C_i \quad (5.6)$$

where, $C_0 = IV$

The Output Feedback (OFB) mode uses block cipher as a key stream generator. The OFB is XOR the plaintext blocks with the generated stream to obtain the ciphertext. Since the XOR operation results in symmetry, encryption and decryption are the same. The mathematical formulae for the OFB are:

$$C_i = P_i \oplus O_i \quad (5.7)$$

$$P_i = C_i \oplus O_i \quad (5.8)$$

$$O_i = E_k(I_i) \quad (5.9)$$

where, $I_i = O_{i-1}$ & $I_0 = IV$

The Counter (CTR) mode was introduced in 1979 by Whitfield Diffie and Martin Hellman (Lipmaa et al. , 2000). It is similar to the OFB, and suitable for implementation on multi-processor devices where many blocks can be encrypted in the same time. Furthermore, successive values of a counter are applied to a block cipher encryption process to generate the key stream.

Another mode of encryption was proposed by the Institute of Electrical and Electronics Engineers (IEEE Std, 2008) and recommended by the National Institute of Standards and Technology (NIST of USA) (Paar and Pelzl, 2009) called XTS. XTS mode is XEX-based tweaked-codebook mode with ciphertext stealing.

Generally speaking, cryptography can be the best solution for data security (Paar and Pelzl, 2009, Van Tilborg and Jajodia, 2014). In fact, cryptography can help

improve data security in many scenarios. First, the provider of services cannot acquire any information about encrypted user data (Kamara and Lauter, 2010). Second, cryptography applications can easily be installed in computers, smart phones and other mobile devices so users can conveniently protect and share their data with trusted parties. Thus, a cryptosystem can provide authentication, confidentiality and integrity, as well as trusted data sharing (Kahate, 2013, Kamara and Lauter, 2010).

There are two main cryptographic schemes, deterministic encryption and probabilistic encryption. For deterministic encryption, encrypting identical plaintext multiple times yields identical ciphertext every time. On the other hand, encrypting identical plaintext many times using probabilistic encryption yields a different ciphertext every time even if the same encryption key is used (as different Initial Vectors (IV) are used). For example, the ECB mode uses deterministic encryption while the CBC mode uses probabilistic encryption.

Deterministic encryption is vulnerable to substitution attack, which means it is easy to attack the ciphertext if the same message is sent twice (Paar and Pelzl, 2009). Therefore, Goldwasser and Micali brought in probabilistic encryption in 1984 (Goldwasser and Micali, 1984). This is a scheme for encryption where a plaintext is encrypted into one of many possible ciphertext (not only a single ciphertext as in deterministic scheme) and is immune to substitution attacks (Fuchsbauer, 2006).

Table 5.1 shows the definitions of symbols, operations, functions, inputs and outputs for the PBC mode used in this chapter.

If a hacker can launch a successful attack to an exposed service, he/she may be able to get all the user's information. Furthermore, if the hacker tries to use or re-sell the stolen information, it would be a privacy and security problem. To overcome these issues, service providers often use secure and reliable encryption techniques to secure their systems. While cryptography can offer a high level of data security, there are security concerns (Chen and Zhao, 2012, Gonzalez et al. , 2012). Cryptanalysis also shows some weaknesses in various block cipher modes, such as the sequential nature of the CBC mode (Beeputh et al. , 2010). In response, a parallel block cipher mode was developed (Sahi et al., 2015).

In the next section, the literature of the block cipher operation modes will be reviewed.

Table 5.1 Definitions of symbols, operations, functions, inputs and outputs

	Symbols	Operations and Functions
1	P is the plaintext;	$\oplus$ *(XOR)* is the bitwise exclusive-OR operation;
2	P_i is the i^{th} plaintext block;	$E_K(P_i)$ is the encryption function that encrypts P_i under the primary secret key K;
3	C_i is the i^{th} ciphertext block;	$E_{K1}(P_i)$ the encryption function that encrypts P_i under the 2nd key *K1*;
4	$C_0(W)$ is the 1st ciphertext block which represents the encrypted hash value W;	$E_{K2}(P_i)$ the encryption function that encrypts P_i under the 3rd key *K2*;
5	$C_1(Y_0)$, $C_2(Y_1)$, *etc.* is the rest of the ciphertext block (starting from the 2nd to the end) which represents the encrypted plaintext;	$D_K(C_i)$ is the decryption function that decrypts C_i under the primary secret key K;
6	IV is the initialisation vector;	$D_{K1}(C_i)$ is the decryption function that decrypts C_i under the 2nd key *K1*;
7	K is the primary secret key;	$D_{K2}(C_i)$ is the decryption function that decrypts C_i under the 3rd key *K2*;
8	$K_1 = IV \oplus K$ is the 2nd key (obtained from K), used to encrypt the hash value $H(P)$;	$H(P)$ is the hash function that change the plaintext P to hash value $H(P)$;
9	$K_2 = ((IV \oplus K) \oplus H(P))$ is the 3rd key (obtained from K & K_1), used to encrypt the plaintext P_i;	$H_2(P)$ is the hash value of the decrypted P;
10	W is the encryption of the hash value $H(P)$;	*Combination* is the operation of combining the encrypted hash value block W and the encrypted plaintext blocks Y_i to deliver the ciphertext blocks C_i;
11	Y_i is the encryption of the plaintext P_i;	*Extraction* is the operation of extracting the encrypted hash value block W and the encrypted plaintext blocks Y_i from the received ciphertext blocks C_i;
12	O_i is the output of the OFB mode before XORing with the plaintext;	*AES128* is the encryption process using Advanced Encryption Standard with a block size of 128 bits;
13	I_i is the input of the OFB mode;	$AES^{-1}128$ is the decryption process using Advanced Encryption Standard with a block size of 128 bits.
14	*ID* is the bank identification number;	$CIPH_k$ *and E* are the encryption functions that encrypt under the key K**;**
15	# is the customer account number;	α^j is sequence number of the 128bit block inside the data unit raises to the power j which is counter, in XTS mode.
16	$ is the money amount;	
	Inputs and Outputs	
	Input *(K, IV, P)*:	
1	K whose length is 128 bits;	
2	IV with the length of 128 bits;	
3	P of any length.	
	Output *(C)*:	
4	C having the same length of P plus one block of hash value $H(P)$;	

5.2 Related Works

The five approved block cipher modes (ECB, CBC, CFB OFB and CTR) are divided into two types. The ECB and CBC modes use block cipher for encryption, while the CFB, OFB and CTR use block cipher as a key stream generator.

Each of these five modes has its own shortcomings. The deterministic properties of

the ECB is the biggest issue in this mode leading to a substitution attack (Paar and Pelzl, 2009), as shown in the following example.

Assume the data shown in Figure 5.1 are to be transferred between banks. Let assume that each of the fields has exactly the same sized block used in the block cipher (16 bytes in the case of AES). The encryption key shared between the two banks does not change too frequently. Due to the nature of the deterministic properties of ECB mode, an attacker can easily substitute block number 4 (which contains the receiving account number) without deciphering with the ciphertext block with his own account number. As a consequence, all transfers from bank A to bank B will be redirected to the attacker's account. Interestingly, the attack works without having to attack the block cipher (e.g. AES) itself.

The CBC mode does not allow the encryption process to be parallelised to make it work faster as there is reliance on the previous ciphertext block (Beeputh, Doomun, 2010). Furthermore, in the CBC mode, chaining between the blocks means one corrupted cipher block will result in two corrupted blocks (Stamp, 2011). An example of breaking the CBC chain is shown in Table 5.2.

The CFB and OFB modes have similar issues as the CBC mode, in that they cannot parallelise their encryption processes to make them faster (Paar and Pelzl, 2009). Therefore, they will struggle in terms of speed, especially with big data. In addition, there is a common problem for the CFB, OFB and CTR stream modes; the flipping bits. If a bit flip error arises in ciphertext in a particular block, then after decryption, the error is limited to the exact bit of the exact block of plaintext (Lipmaa et al. , 2010). Flipping a ciphertext bit flips the corresponding plaintext bit. The attack aims to change a bit in the decrypted plaintext by flipping a bit in the ciphertext thereby changing the decrypted plaintext.

For example, assume an attacker knows the ciphertext for an electronic fund transfer, which contains the ASCII string "$1000.00" for the transfer amount. The attacker can alter the amount to "$9000.00" by XOR the part of cipher text for the string with the result of ("$1000.00" ⊕ "$9000.00"):

Blocks				
1	2	3	4	5
Sending Bank A	Sending Account	Receiving Bank B	Receiving Account	Amount
ID	**#**	***ID***	**#**	
				$

Figure 5.1 Example for a substitution attack against the ECB mode

Table 5.2 Example of breaking the CBC chain

CBC Encryption:	CBC Decryption:	If C_1 is corrupted to X then:
$\mathbf{C_0 = E_k (P_0 \oplus IV)}$	$P_0 = D_k (C_0) \oplus IV$	$P_0 = D_k (C_0) \oplus IV$
$\mathbf{C_1 = E_k (P_1 \oplus C_0)}$	$P_1 = D_k (C_1) \oplus C_0$	$P_1' = D_k (X) \oplus C_0 \neq P_1$
$\mathbf{C_2 = E_k (P_2 \oplus C_1)}$	$P_2 = D_k (C_2) \oplus C_1$	$P_2' = D_k (C_2) \oplus X \neq P_2$
$\mathbf{C_3 = E_k (P_3 \oplus C_2)}$	$P_3 = D_k (C_3) \oplus C_2$	However:
$\mathbf{C_4 = E_k (P_4 \oplus C_3)}$	$P_4 = D_k (C_4) \oplus C_3$	$P_3 = D_k (C_3) \oplus C_2$
		$P_4 = D_k (C_4) \oplus C_3$

"$1000.00" ⊕ ("$1000.00" ⊕"$9000.00")

= "$1000.00" ⊕ "$1000.00" ⊕ "$9000.00"

= "$9000.00"

Note that:

"$1000.00" ⊕ "$1000.00" = all zeros

All zeros ⊕ "$9000.00" = "$9000.00"

In addition, there is no chaining between blocks in the CTR mode. Thus, if Eve (an attacker) modifies any block, Bob (a receiver) will never know about that. Moreover, the CTR requires synchronisation of the initial counter value between the encryption party and decryption party. A policy to ensure that the next communication's initial counter value is unique and correct should be in place and enforced. This might require a private communication channel (Tarhuni et al. , 2003).

XEX-based Tweaked-codebook mode with ciphertext Stealing (XTS) mode is was developed based on the XOR Encrypt XOR (XEX) mode (Andreeva et al. , 2013, Rogaway, 2004) with Ciphertext Stealing (CTS) mode feature. However, it is slower than the CBC mode due to more mathematical operations, such as XOR and multiplication (Alomari et al. , 2009). XTS mode uses the XOR function, before and

after each encryption process (like XEX), and uses two keys instead of one key (Liskov and Minematsu, 2008). Since there is no built-in mechanism to detect alterations (an active attack) (Dworkin, 2010 (A)), any altered ciphertext (by attacker) will be decrypted to some legitimate plaintext.

Some may argue that the CTR_CBC-MAC mode (CCM) can provide an integrity check in addition to performing encryption (Zukarnain, 2014). However, the inherently sequential nature of the CBC mode, and the need for two passes with AES, can make CCM-AES a slower scheme (Rogaway, 2011). CCM mode requires a longer processing time than the CBC mode (CBC is a special case of CCM).

A Galois/Counter mode of operation (GCM) was proposed by McGrew and Viega in 2004 (McGrew and Viega, 2004). The GCM mode has been approved by the NIST of USA as an authenticated mode of operation (Dworkin, 2007). The GCM mode ensures data confidentiality using the CTR mode. It delivers a guarantee of the authenticity of the confidential data using a hash function which is defined over a binary Galois field (Dworkin, 2007). However, while the CTR mode is a part of the GCM mode, it could be vulnerable to the flipping bits attack mentioned above. In addition, the GCM mode is vulnerable to replay attacks (Dworkin, 2007). In other words, similar to numerous authentication modes, GCM does not naturally stop an attacker from interrupting the message and replaying it for authenticated decryption later. For example, in an attempt to imitate a party that has access to the key.

The Extended Codebook (XCB) Mode of Operation was proposed by McGrew and Fluhrer in 2004 (McGrew and Fluhrer, 2004). The XCB was the first block cipher that was constructed using the hash-CTR-hash. The XCB used the CTR mode of operation in order to ease tackling variable-length messages. However, Chakraborty and Sarkar reported that the XCB has limitations (Chakraborty and Sarkar, 2008). While the XCB mode adopts the ciphertext block as a key, the key length and the length of the block must be identical (Chakraborty and Sarkar, 2008). Thus, it is impossible to use the XCB mode once the length of the key is different from the length of the block, as is the case in AES with 192-bit key and 128-bit block.

Parallel ciphering can improve the speed performance of the encryption. Big data are usually stored in distributed locations, therefore parallel encryption is needed to encrypt the large number of blocks or chunks of data separately, which takes much longer when using sequential encryption.

The next section presents the proposed Parallel Block Cipher (PBC) mode.

5.3 Parallel Block Cipher Mode

Data encryption is one of the most common practical choices for computing developers and database designers (Gampala et al. , 2012) working to protect data. The biggest database management systems like Microsoft's SQL Server and Oracle adopted the CBC mode in block cipher algorithms (Dansereau et al. , 2006, Fazackerley et al. , 2012). Furthermore, most cryptosystems nowadays adopt the CBC in their block cipher. Because of its popularity, the CBC has been chosen by this research study as the sequential block cipher mode to compare with Parallel Block Cipher (PBC) mode. Before presenting the comparison, PBC is briefly introduced below.

The PBC mode parallelises the encryption process. It converts plaintext blocks to ciphertext blocks without any chaining occurring between data or outputs from another block. The hash of the file is utilised to generate encryption and decryption keys. Multiple blocks can be encrypted in one round and each plaintext block does not need to wait for outputs from a previous block to start or complete the encryption process; making parallelisation possible. The proposed PBC mode makes use of 128 bit key AES and a hash function (Wang et al. , 2011). The encryption diagram is shown in Figure 5.2.

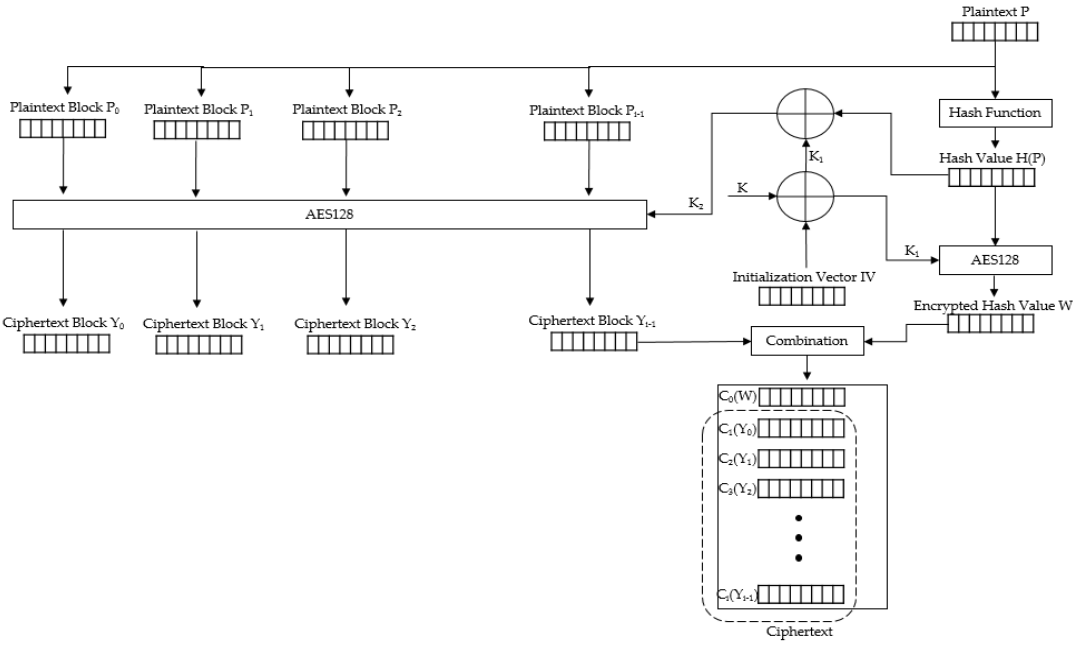


Figure 5.2 The PBC encryption diagram

The following steps are used to encrypt data using the PBC mode:

Step 1. $P \xrightarrow{hash(P)} H(P)$

First of all, the plaintext *P* will be hashed to obtain the hash value of the data *H(P)* in order to mix it with the key.

Step 2. $H(P) \xrightarrow{E_{K1}(H(P)),\ AES128} W$

$P_i \xrightarrow{E_{K2}(P_i),\ AES128} Y_i$

In the second step, the hash value *H (P)* is encrypted using *K1 = IV* $\oplus$ *K* (where *K* is the session key) to obtain W. Then, encrypt the plaintext P_i using *K2 = ((IV* $\oplus$ *K)* $\oplus$ *H (P))* to yield the ciphertext of the data Y_i.

Step 3. $W + Y_i \xrightarrow{Combination} C_i$

Finally, *W* and Y_i are combined to produce the complete ciphertext C_i which involves the ciphertext of the data and the hash value.

The following steps are used to decrypt data using the PBC mode:

Step 1. $C_i \xrightarrow{Extraction} W + Y_i$

To decrypt the ciphertext C_i using the PBC mode, the ciphertext of the hash value *W* which is located in the first block of the ciphertext must be extracted, the rest of the ciphertext is Y_i.

Step 2. $W \xrightarrow{D_{K1}(W),\ AES^{-1}128} H(P)$

Second, the ciphertext of the hash value *W* will be decrypted to retrieve a copy of the original hash value *H (P)*.

Step 3. $Y_i \xrightarrow{D_{K2}(Y_i),\ AES^{-1}128} P_i$

Then decrypt using K_2 the rest of the ciphertext Y_i to retrieve P_i which represents the original data.

Step 4. *Integrity check,* $H_2(P) = H(P)$, (optional)

Furthermore, the PBC mode facilitates a message integrity check (Step 4). The PBC mode can check the integrity of the message by comparing the hash of the copy of the original file which derived from *Ci* with H(P). This step is optional, because the proposed PBC mode is an encryption mode and not an authenticated encryption mode.

Decryption diagram is shown in Figure 5.3.

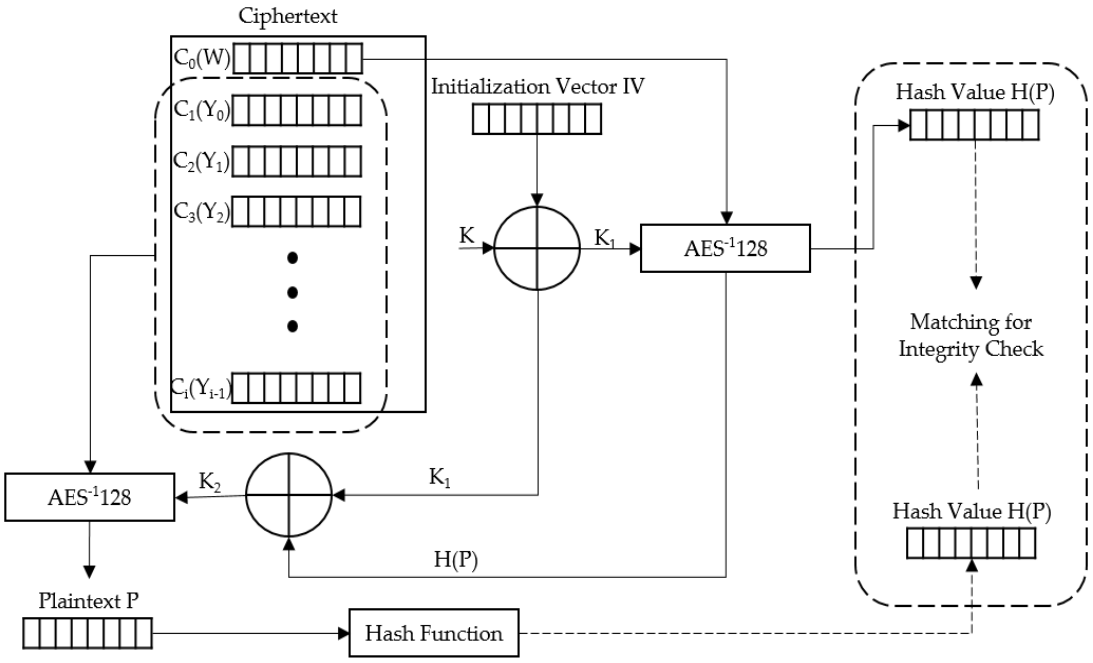


Figure 5.3 The PBC decryption diagram

5.4 Implementation

Microsoft Visual Studio (MVS) 2010 and Dot Net Framework 4.5 were used to implement the CBC mode and the PBC mode. Two console applications were developed. The first for the execution of the CBC mode and the other for the execution of the PBC mode. System.Security.Cryptography Namespace is used to utilise cryptographic services in MVS such as secure encryption and decryption of data using AES, as well as calculating hash values using hash functions.

To parallelise the PBC mode, a Task Parallel Library (TPL) was also used. The TPL is a set of public types and Application Programming Interfaces in the System.Threading and System.Threading.Tasks Namespaces. The TPL aims to simplify the process of adding parallelism to applications and efficiently use all the CPUs that are available. In this project's case, the TPL can let a multiple-cores computer run eight CPUs at the same time to allow the program to run in parallel.

The executions of the sequential CBC mode and the parallel PBC mode have been conducted using thirteen data samples with different sizes (100MB, 200MB, 300MB, 400MB, 500MB, 600MB, 700MB, 800MB, 900MB, 1GB, 1.1GB, 1.2GB and 1.3GB).

The next section discusses and evaluates the results obtained.

5.5 Results and Evaluation

5.5.1 Experimental Results

The PBC mode has been tested and compared with the CBC mode in three different scenarios. First, a single data file was used as input for the PBC mode that uses only one process. In the second scenario, the data file was manually split into data blocks and used as input for the PBC mode using one process from one server at one time. Finally, a single data file was used as input for the PBC mode using multiple processes. In the last scenario, the PBC mode automatically splits the data file, processes the blocks, and re-combines encrypted blocks to a single encrypted data file.

5.5.1.1 Scenario 1

The PBC mode has been tested using a single process and compared with the CBC mode (Sahi et al., 2015) in computers running Windows 7, with an Intel(R) Core(TM) i7-2600 CPU and 16GB RAM. CrypTool 2.1 and AES128 algorithm was also used. The CBC_AES parameters were: action is encryption, key size is 128 bytes, block size is 128 bytes, chaining mode is the CBC, and padding mode is zeros. The PBC and CBC modes both run with a single process and a single input file, which was not split into blocks. The PBC mode was running like the CBC mode in a sequential manner. The results, as shown in Figure 5.4, indicate that the PBC mode is slightly faster than the CBC mode even when both were running on a single process. However, the time difference between the CBC mode and the PBC mode is quite small; only 1.61% on average.

5.5.1.2 Scenario 2

When a single process is used, parallel processing for the data blocks were in fact not tested. So, the PBC mode was tested with input data broken manually into 8 sub-files. Each sub-file was fed sequentially to a single process of one single server. Outputs from all eight processes were manually combined to generate the encrypted file. The time difference between the CBC mode and the PBC mode increased by 19.61% on average. The PBC can save approximately one fifth of the execution time as compared to the CBC mode.

5.5.1.3 Scenario 3

Nowadays, most performance computers have more than one processor (CPU) and running multi-processes. These computers can be treated as multiple computers in a single box.

This experiment uses one computer with Intel(R) Core(TM) i7-2600 CPU (8 CPUs),

16GB RAM, MS Visual Studio 2010, Dot Net framework 4.5, CrypTool 2.1, and AES128 algorithm. Eight processes were running in parallel.

Assumption 1. This computer has multiple processors, can be multiple servers in a single box simulating eight servers, as shown in Figures 4.5a and 4.5b.

Assumption 2. Every server mentioned in this chapter uses 1 CPU.

Limitation 1. Using test data files ranges from 100MB to 1.3GB due to computer hardware limitation.

Limitation 2. A multi-processor machine (single machine due to hardware limitation) was used to simulate multi-servers. Issues such as memory contention, queueing delays, blocked access to memory and secondary storage, concurrency issues and scheduling overheads may affect the overall performance of the PBC to a certain extent.

To produce the results of this chapter, one computer with Intel(R) Core(TM) i7-2600 CPU (8 CPUs) and 16GB RAM were used. Windows 7 as an operating system, and Microsoft Visual Studio 2010 as well as Dot Net Framework 4.5 were used to implement the CBC mode and the PBC mode. Two console applications for CBC and PBC modes were developed. The first was used to execute encryption using the CBC mode and the other for the PBC mode. Files with sizes ranging from 100MB to1.3GB were used as input. The implementation code for PBC are listed in Appendix B.

While the CBC cannot run in parallel (Yeh et al. , 2009) due to the block chaining feature, it was run on a single server, as shown in Figure 5.5b. There were two options to create a single server using a multi-processor computer. The first option was to use a virtual machine (VM) that runs on one server. Figure 5.6a shows the CPU usage when the CBC mode is run using of VM which uses one server.

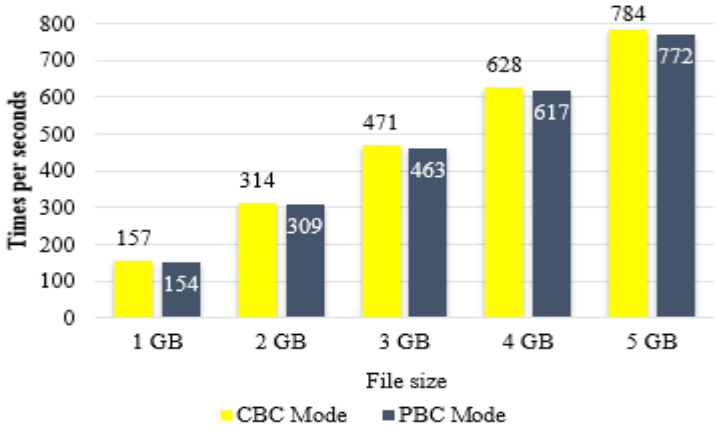
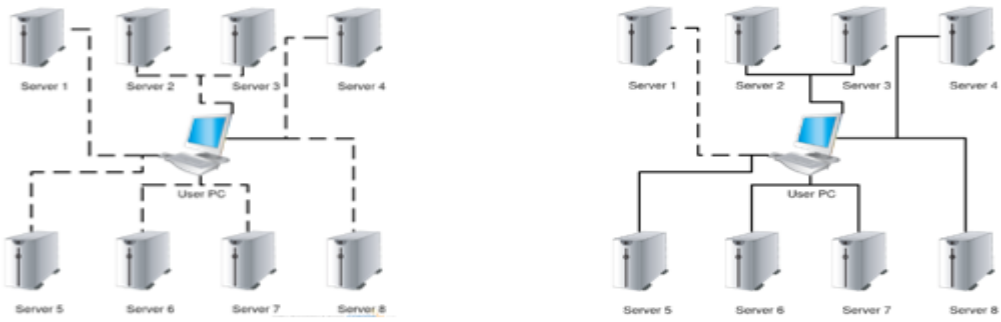


Figure 5.4 Execution times of the PBC and the CBC modes using single process



(a) The PBC servers' box (multiple servers are running)

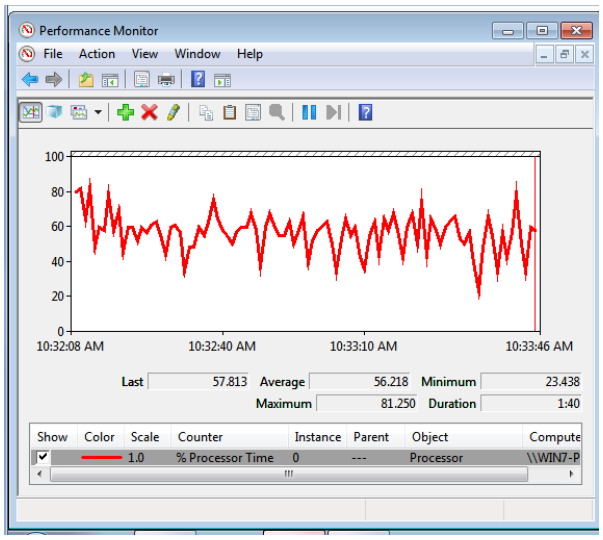
(b) The CBC servers' box (single server is running)

Figure 5.5 Using single machine with multiprocessor to provide a single server and multiple servers

Encrypting 1.3GB file takes about 99.49 seconds (1:39.49), as shown in Figure 5.6b.

The second option was to use a CPU Affinity Mask (AM) to force the multi-processor computer to use a single CPU and 16GB RAM. As a result, encrypting a 1.3GB file using the CBC mode took about 92.84 seconds (1:32.84), as shown in Figure 5.7b. This is very close to the previous result and the improvement may be attributed to the larger RAM used.

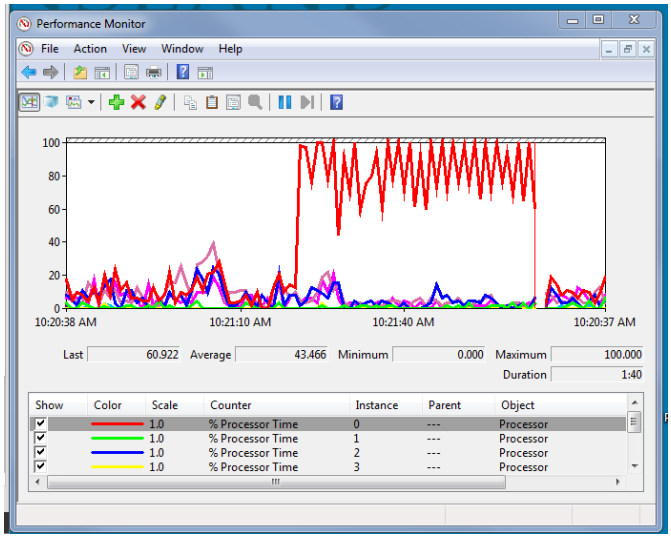
In scenario 3, the PBC can run processes in parallel using all the servers in the box (eight servers in this test), as shown in Figure 5.8a. The whole duration of hashing, breaking the file into parts, encrypting and combining the encrypted parts into one encrypted file using the PBC mode takes an average 41.4 seconds for a 1.3GB data file. The result is shown in Figure 5.8b, which is significantly better than the previous results. It shows that the PBC mode is more than twice as fast as the CBC mode.



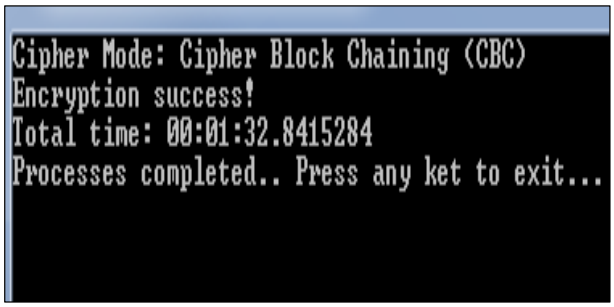
(a) Single server using VM

(b) The execution result of CBC mode using virtual machine

Figure 5.6 Running the CBC mode using single server in a virtual machine

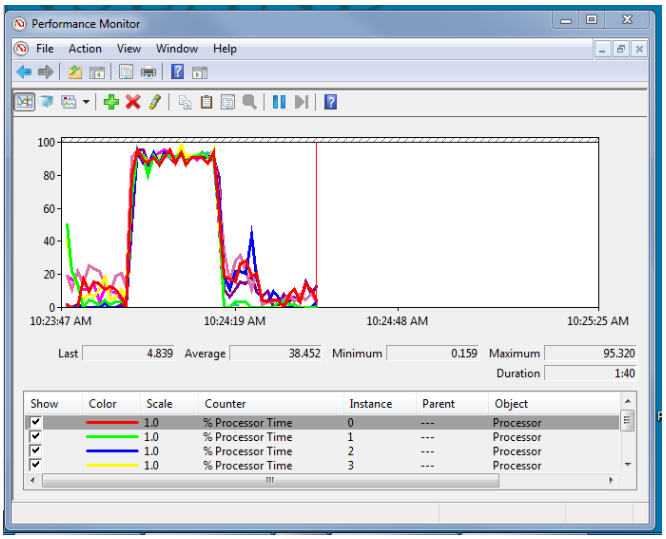


(a) Single server using AM

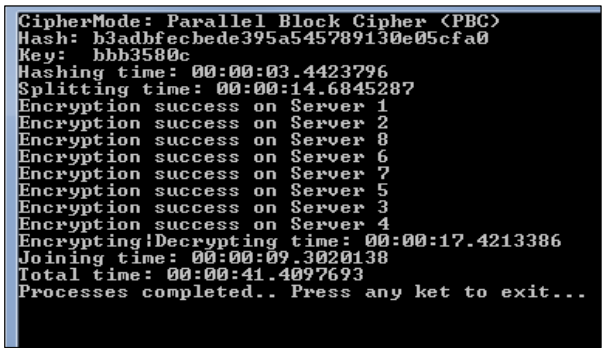


(b) The execution result of CBC mode using Affinity Mask

Figure 5.7 Running the CBC mode using Affinity Mask



(a) Eight servers running in parallel



(b) The execution result of the PBC mode

Figure 5.8 Running the PBC mode on multiple servers

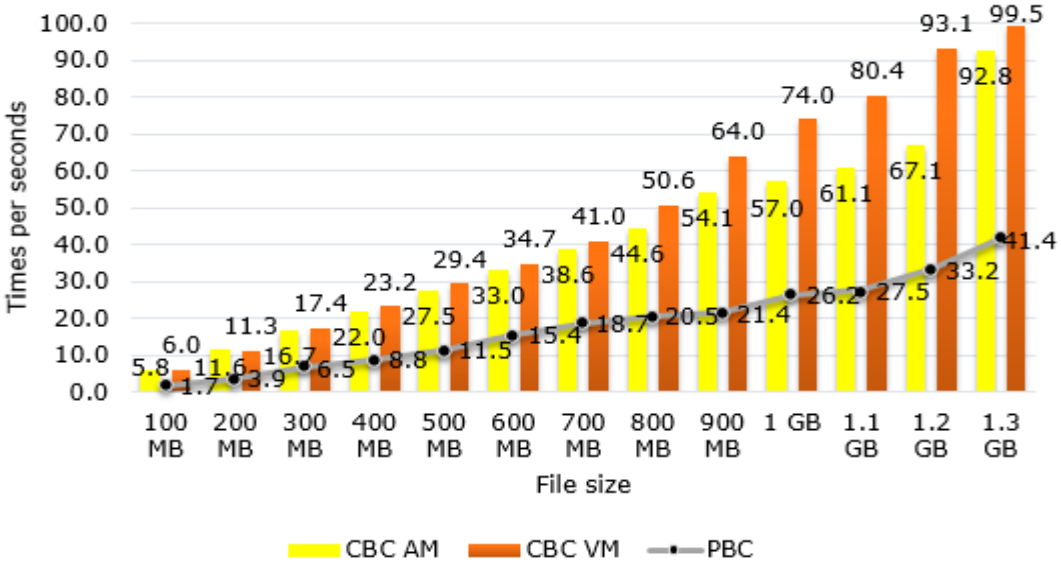


Figure 5.9 Execution times of the PBC and the CBC modes using multiple processes

Note that when 8 servers (processors) are used, the expected encryption time is around one eighth of that used by the CBC mode (approximately 13 s). The actual time used was around 17.42 s. With overheads for splitting the data file (14.68 s) and joining the cipher blocks (9.30 s), the total time span adds up to 41.40 s.

Multiple servers' tests were executed using data files from 100MB to 1.3GB, with an increment of 100MB for each data file. Execution time was measured in seconds. The percentage time saving between the PBC mode and the CBC mode was about 60% on average for all data file sizes. In other words, the PBC mode takes only 40% of the CBC mode execution time to encrypt the same amount of data. The test results are shown in Figure 5.9.

5.5.2 Discussion

The PBC mode is faster than the standard CBC mode in terms of execution time. As mentioned in the experimental results section, the proposed mode proves to be superior to the original CBC mode in terms of execution time under a multiple server's environment. As shown in Figure 5.9, the PBC can save up to 60% of the CBC mode's encryption time.

Furthermore, the proposed PBC mode provides confidentiality. The PBC mode has an essentially similar concept of chaining blocks as the CBC block cipher in terms of security performance. The PBC mode chains all the blocks together using the hash value of the plaintext. The key used to encrypt the plaintext blocks is generated using the hash value therefore, the hash value is used in encrypting every single block. If one cipher block is corrupted, then the corresponding plaintext block will also be corrupted. The hash of the decrypted plaintext will differ from the original hash sent to the receiver. This will provide an integrity check on top of the confidentiality provided by encryption. Accordingly, the security concept of the CBC can be applied straight away to those of the PBC mode. Since the CBC mode of operation is revealed to be safe against numerous attacks such as chosen ciphertext attacks, ciphertext only and, known plaintext, the proposed PBC mode can also be said to be safe against these attacks. The key stream of the CBC mode involves two inputs (key and IV), while the PBC mode key stream involves key, IV, and the hash value of all blocks, which adds more degree of randomness, and is more secure.

Moreover, the proposed PBC mode has faster error recovery than the CBC mode. As mentioned in Table 5.2, in the CBC mode one corrupted ciphertext block will affect

two plaintext blocks as a result. However, in the PBC mode one corrupted ciphertext block will affect the corresponding block only. As a result, in the case of AES, corrupting 16 bytes block in the CBC mode affects 32 bytes blocks, and corrupting 16 bytes block in the PBC mode affects only 16 bytes block.

A multi-processors machine (single machine due to hardware limitations) was used to simulate multi-servers. Issues such as memory contention, queueing delays, blocked access to memory and secondary storage, concurrency issues and scheduling overheads may affect the overall performance of the PBC to a certain extent. However, a significant 60% improvement using a single machine with multi-processors, and PBC demonstrated that PBC (parallel processing) has a better performance than sequential processing.

In relation to the CTR mode that is intended for parallelised processes and has been commonly utilised in high speed network standards, the PBC mode does not require counter synchronisation. The unique values of IV and the hash deliver better security than the CTR mode.

The implementation code for PBC are listed in Appendix B.

5.6 Chapter Summary

This chapter presented the PBC mode which considerably increases the speed of the encryption process. Results of speed performance tests of the PBC mode using various settings were presented and compared with the standard CBC mode. While maintaining security, the PBC mode was shown to offer a significant increase in performance when compared with the CBC mode. Therefore, hypothesis H3 (*A new parallel hash-based block cipher mode of operation would be able to improve the encryption process on top of security assurance*) has been proven and research question 3 is answered in this chapter.

The next chapter gives the details about integrating the PBC mode presented in this chapter and the TPAKE presented in Chapter 4 to preserve security and privacy of eHealth clouds and answer research question 4. Chapter 6 also presents a disaster recovery plan.

6

CHAPTER 6

SECURITY AND PRIVACY APPROACHES OF EHEALTH CLOUDS

Chapter 4 presented an authenticated key exchange (TPAKE) protocol. Chapter 5 presented a Parallel Block Cipher (PBC) mode. This chapter adopts the PBC mode and the TPAKE protocol to introduce a security-preserving approach and a privacy-preserving approach to preserve the privacy and the security of an eHealth cloud.

The chapter also presents a disaster recovery plan which ensures the connectivity of users during disaster time (Sahi et al., 2016).

6.1 Introducing eHealth Cloud Security and Privacy Approaches

The technologies of cloud computing (CC) provide simple and easy on-demand network access to a shared group of computing resources which are simple to install and maintain with minimal effort. They have become an important keystone technology. Many scientists and researchers claim that cloud computing has changed the computing processes and IT markets. When access is powered by cloud computing, users can adopt comprehensive sets of tools for assessing various applications, storage and platforms through the Internet, as well as using other services offered by cloud producers.

The National Institute of Standards and Technology (NIST) stated that CC is a model for using computer resources and other modern technological functionality in the information technology world to provide services such as storage and applications (Mell and Grance, 2011). Users can access and use cloud computing services without the need to acquire knowledge, expertise or even administration of infrastructure that

support these services. There are three main types of services offered by the cloud (Sugumaran et al., 2014): Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) (Kushida et al., 2015).

In addition, four deployment models have been identified for cloud architecture solutions; namely Private cloud, Community cloud, Public cloud and Hybrid cloud (Hsu and Cheng, 2015; Zissis and Lekkas, 2012). Since cloud computing shares resources distributed throughout the Internet and among intranets, security is an important issue.

CC can be applied in many sectors. This research focuses on the health sector. CC gives healthcare providers the ability to diagnose and evaluate a patient's health even though the patient is not at the hospital. This requires the patient's medical data to be distributed between physicians, nurses, insurance companies as well as the data owner, so delivering flexible and prompt care to patients. The distribution of the patient's medical data raises security and privacy issues. Therefore, this research proposes two approaches designed to preserve privacy and security in eHealth clouds.

Before presenting the approaches, it is necessary to define Personal Health Records (PHRs) and Electronic Health Records (EHRs).

Personal Health Records (PHRs): A PHR is medical data owned and managed by the patient himself/herself (Abbas and Khan, 2014). A good PHR contains a precise and short record of the patient's medical history data collected from various sources (such as EHRs). Everyone having the required authorisations to display the PHR can easily reach these data.

Electronic Health Records (EHRs): According to Zapata (Zapata et al., 2014), the International Organisation for Standardisation (ISO) stated that the EHR is a "repository of information regarding the health status of a subject of care, in computer processable form". Using the words of ISO/TS 18308 (ANSI, 2003), the main purpose of an EHR is to deliver a registered record of healthcare which supports current and upcoming healthcare received by the patient from various healthcare providers.

Today, cloud computing is used by millions of people around the world. Cloud computing gives users the opportunity to store data in the cloud for easy access anytime and anywhere (Carroll et al., 2011). However, in the cloud environment, the user's data are controlled by service providers and not by the users themselves. The potential for data leaks is real, either intentionally or accidentally, which is unacceptable (Gonzalez et al., 2012). Common problems of security and privacy in

eHealth clouds include confidentiality, integrity, authentication, access control, and non-repudiation (Rosado et al., 2012).

The overall cloud computing theme is that we should be able to use all kinds of functionality and services provided by the cloud, but we would like to maintain our data privacy and security. Therefore, issues regarding privacy and security of data are the main factors limiting the widespread use of cloud computing. Much research has been done on these issues and several researchers claim that a versatile cryptography system may handle data security and data privacy issues more effectively than other methods (Talbot, 2010). With the use of cryptography systems, the security and privacy approaches proposed by this research can tackle security and privacy issues.

Although security is the most important factor in any cloud project, disaster recovery (DR) planning must also be considered.

There are three types of disaster that may cause major damage to any system: (1) natural disasters, such as flood, earthquake, and volcano, (2) man-made disasters, such as cybercrime and technological terrorism, and (3) technological accidents, such as infrastructure failure, and transportation failure (Snedaker, 2013). To overcome these kinds of disaster, a recovery plan must be set up.

In the next section, the literature of the health cloud security and privacy will be reviewed.

6.2 Related Work

This section reviews related work on the preservation of security and privacy in eHealth clouds as well as the DR planning.

Users are using CC in various ways, such as checking email by Yahoo, writing documents by Google Docs, and storing data in iCloud. CC delivers numerous benefits. For example, low costs due to pay-as-you-go models, extraordinary availability as data is commonly distributed between a numbers of servers, and load balancing (Mell and Grance, 2011). Furthermore, CC is benefiting health organisations (Giniat, 2011).

Health organisations have been quick to move to CC for the obvious advantages of data storage and sharing. These organisations are keen to store and share PHRs and EHRs using the cloud, thereby eliminating the geographical boundaries between health organisations and patients (Wu, 2012). Sharing data using CC has rapidly become a very important component for healthcare providers and many other organisations.

According to Thilakanathan et al. (Thilakanathan et al., 2014), for most organisations, the percentage of the data shared with clients using CC is about 74% and with dealers is about 64%.

On the other hand, it is important for medical data to be safe from unauthorised access and unwanted modifications. CC, however, is vulnerable to various security and privacy attacks. Consequently, many healthcare providers are unwilling to implement CC technologies as a patient's information privacy may be breached. According to Van et al. (Van Gorp et al., 2014), the main hurdle delaying the growth and extensive acceptance of CC is privacy and security issues. Actually, most privacy and security attacks are caused by the Cloud Service Providers (CSP) themselves (Rocha et al., 2011) as they commonly have access to the Cloud Storage (CS) and they may also sell the data records to gain profits. Indeed, insider attacks are one of the main problems related to CC, as pointed out by El-Gazzar et al. and Pasupuleti et al. (El-Gazzar et al., 2016; Pasupuleti et al., 2016).

Fujisaki et al. proposed a PKE-based (Public Key Encryption) approach named RSA-OAEP (Fujisaki et al., 2004) however, PKE-based approaches are computationally inefficient because of the larger key size and slower operation.

Jafari et al. introduced an approach which gives the patient the possibility of controlling his EHRs. This approach limits the patient to managing records authored by other parties, such as physicians and nurses (Jafari et al., 2011). On the other hand, the cloud service provider cannot retrieve the records in plaintext format. The patient himself and data consumers are given the private and public keys for encryption and decryption (Khalil et al., 2014).

Another approach presented by Zhang et al. (Zhang et al., 2014) is a time-based approach. The approach is efficient in ensuring the privacy of the EHRs at the cloud storage and enhances the operation of key distribution between trusted parties. This approach adopts time-bound hierarchical key management (Bertino et al., 2008). Time-bound hierarchical key management permits trusted parties to gain short-term access to the EHRs, which are encrypted using Symmetric Key Encryption (SKE). However, Zhang's approach is logically inadequate due to the fact that users have to take on multiple roles. Therefore, the users are required to hold and control multiple keys.

Tran et al. (Tran et al., 2011) proposed an approach based on the proxy re-encryption idea. A trusted user can obtain a data record as the proxy will convert the

encrypted data on the data owner's side to differently encrypted data which can be decrypted to plaintext by the trusted user's key. However, because Tran's approach uses ElGamal public key cryptography, the encryption or decryption of very large data is not practical and, unfortunately, very large data is a feature of medical data (Thilakanathan et al., 2014). In addition, this approach does not solve the situation where a revoked party re-joins using another access key.

Liang et al. presented an approach which adopts Ciphertext-Policy Attribute-Based Encryption (CP-ABE) for data sharing in the situation of a revocation operation, which at the same time allows high flexibility, access control and revocation (Liang et al., 2015). This approach, however, is also not effective when considering very large data (Thilakanathan et al., 2014).

An Access Control technique is a policy or rule that allows the restriction of access to a cloud project (Khan, 2012). It also detects unauthorised users who try to access a cloud project. Access Control allows one application to trust the identity of another application (Sosinsky, 2010). While a robust authentication technique is a compulsory requirement for any cloud project, access control cannot secure data at rest and in transit (Sen, 2013; Younis et al., 2014), and it is not satisfactory enough to achieve privacy for PHRs (Yang et al., 2016). Encryption methods are definitely a better choice for protecting data at rest, as well as the choice for protecting data in transit (Sen, 2013). In addition, cryptography offers an integrity check to verify that the data is not compromised or corrupted in transit.

Regarding DR plans, Wood et al. presented a DR plan that utilises three servers and one database, as shown in Figure 6.1. One of the servers is nominated to be used in the event of a disaster, and all users are redirected to that server during the disaster (Wood et al., 2010). However, the authors did not consider the case in which the disaster also affects the nominated server itself. In addition, the redirecting may influence the performance of the system and all users will be disconnected while they are being redirected to the nominated server.

In eHealth clouds environment, a user's data are controlled by the service providers rather than by users themselves. There is therefore a potential for data leaks, either intentional or accidental, which is unacceptable (Gonzales, et al., 2012; Ma, 2012). Furthermore, data in the cloud are stored in geographically diverse locations. Thus, confidentiality, authentication, and communication between parties become an important concern (Hussain & Ashraf, 2014).

Confidentiality is not enough to guarantee eHealth clouds security. Users want to ensure that their data has not been modified or compromised by a third party. Cloud service suppliers should also apply methods to guarantee data integrity (Sugumaran et al., 2014). In addition to encryption, availability of service must be ensured.

Therefore, the following research question have been raised "How can security and privacy be preserved in eHealth clouds? How can a client be enabled to connect to the system at any time, even during a disaster?"

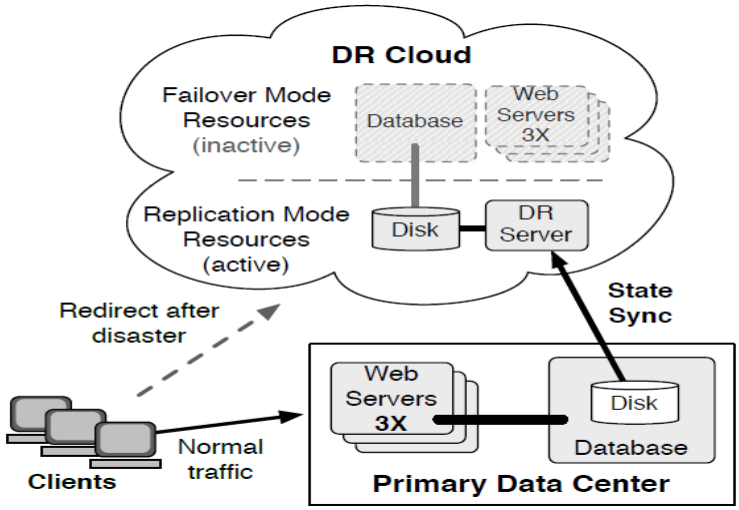


Figure 6.1 Wood et al. DR plan (Wood et al., 2010).

This chapter is concerned with the following issues when strengthening the security and privacy of health records in the cloud:

- How to design compatible security and privacy approaches which can preserve the security of the EHR and the privacy of the PHR at the same time in eHealth clouds:
  - How the proposed approaches perform in emergency situations
  - How to revoke old session keys
  - How to ensure the availability and the continuity of the system during a disaster.

From the point-of-view of this project, security, privacy, and disaster recovery plans are crucial and should be designed together to be homogeneous, accurate, and easy to implement. The lack of any one of these will certainly affect the performance of the others. Thus, designing one of them only, as in most of the related studies, is not enough to deal with the real world.

To build a better cryptography cloud project, four requirements need to be satisfied: authentication, non-repudiation, integrity, and confidentiality (Zissis and Lekkas, 2012). Most previous studies focus on either the security of the EHR or the privacy of PHR, which is not enough to achieve the requirements. Many researchers propose good security approaches to ensure confidentiality, while others propose good privacy approaches to ensure authentication. However, these approaches may not integrate seamlessly. Besides, non-repudiation and integrity must be provided by the same system. Therefore, the researcher was motivated to propose approaches that can be easily implemented and integrated in any distributed system to cover requirements for both security and privacy. Moreover, on top of the security of the EHR and privacy of the PHR, this research study designed a disaster recovery plan to guarantee the availability and the continuity of the system during a time of disaster. While disaster recovery is not considered by most of the studies in the eHealth domain, the proposed approaches and disaster recovery plan will enable data owners and patients to have full and safe control over their records.

This chapter will:

- Propose a security-preserving approach which can ensure the security and integrity of Electronic Health Records
- Propose a privacy-preserving approach which can ensure the privacy of Personal Health Records
- Provide a break-glass access feature to be used in emergency situations. A revocation feature is also provided
- Design a disaster recovery plan to guarantee the availability and the continuity of the system during a disaster.

6.3 Preliminaries

6.3.1 PBC-AES

Sahi et al. (Sahi et al., 2015) first introduced the Parallel Block Cipher (PBC) as a block cipher mode of operation. The PBC adopted the Advanced Encryption Standards (AES) as an encryption algorithm. The PBC mode significantly enhances the encryption process in terms of speed and provides a data integrity check. In the PBC, each block uses the hash value of the shared data to ensure that the key stream has very good randomness. Fast parallel processes and integrity checks are the reasons for

choosing the PBC-AES. Figure 6.2 together with Equations 6.1–6.5 briefly describe the process of the algorithm.

Encryption:

$$W = E_{(K \oplus \mathrm{IV})}(\mathrm{H}(X_i)) \tag{6.1}$$

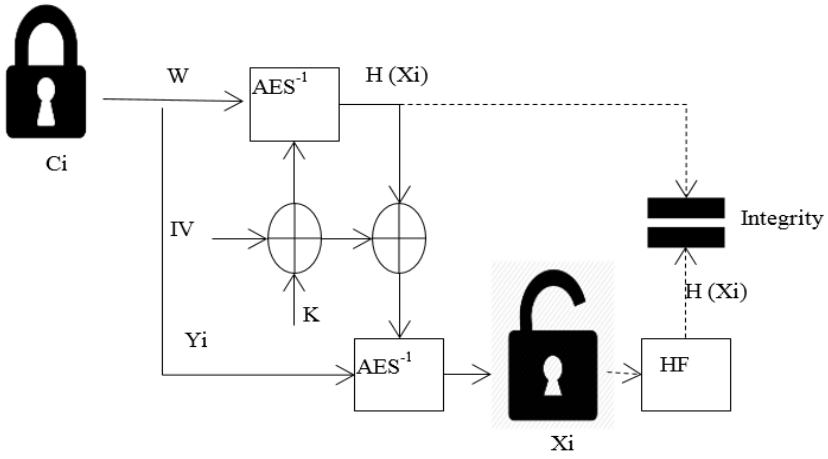
$$Y_i = E_{(K \oplus \mathrm{IV}) \oplus \mathrm{H}(X_i)}(X_i) \tag{6.2}$$

$$C_i = W + Y_i \tag{6.3}$$

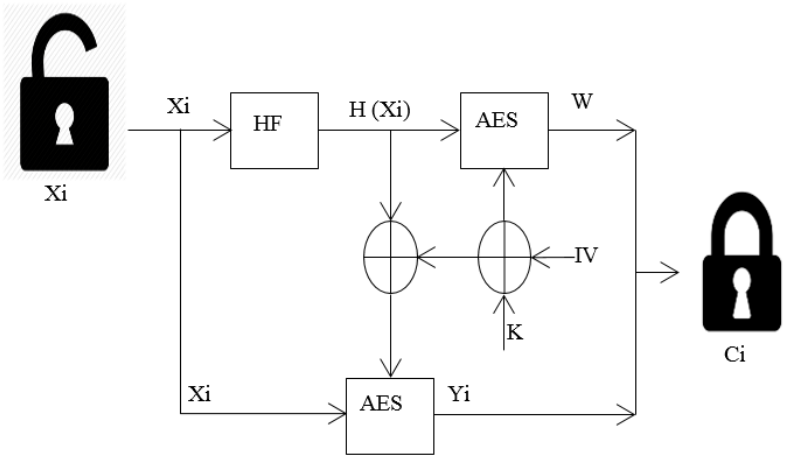
Decryption:

$$H(X_i) = D_{(K \oplus \mathrm{IV})}(\mathrm{W}) \tag{5.4}$$

$$X_i = D_{H(X_i) \oplus (K \oplus \mathrm{IV})}(Y_i) \tag{5.5}$$



(a) Encryption process.



(b) Decryption process.

Figure 6.2 PBC-AES processes (Sahi et al., 2015).

6.3.2 Key exchange protocol

To ensure the privacy of users, the proposed approach adopts a three-party password-based authenticated key exchange (TPAKE) protocol which was introduced in (Khader and Lai, 2015). This protocol ensures both authentication and non-repudiation features. A Geffe generator is used to produce a pseudorandom binary sequence. The resulting sequence is tested using statistical tests including a frequency test, serial test and poker test. Private keys are then generated from the success sequence. In this protocol, data will have a non-repudiation property and no clear data will be sent via the channel. Figure 6.3 shows the process of Khader and Lai protocol.

The next section presents the proposed approaches.

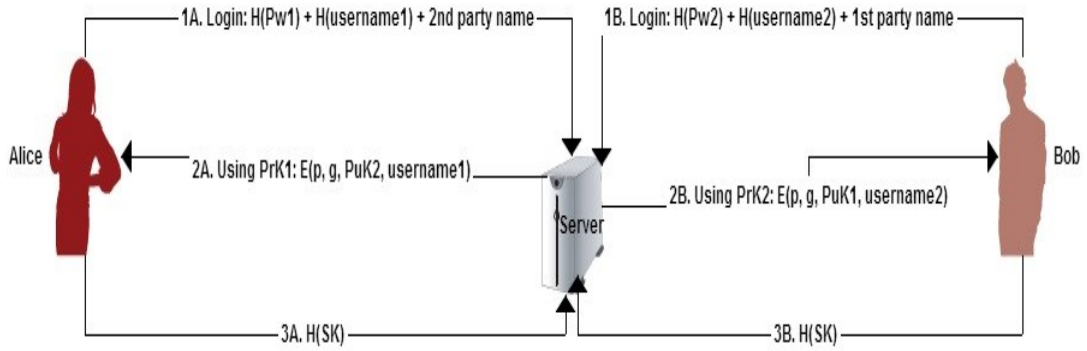


Figure 6.3 Key exchange protocol (Khader and Lai, 2015)

6.4 The Proposed Approaches

While CC is facing numerous privacy and security issues, and despite the fact that most of these issues are caused by Cloud Service Providers (CSPs) themselves, all data in CSPs and Cloud Storage (CSs) must be encrypted. In other words, all medical records including PHRs and EHRs must be encrypted before CC stores and shares it. This will resist any attacks from outsiders as well as from insiders (the CSPs themselves) trying to obtain any valuable data without permission (Khader and Lai, 2015). The patient's privacy needs to be ensured on top of the security for medical records.

The proposed approaches are shown in Figure 6.4 and Figure 6.5. They consist of data consumers, trusted party, patients, and cloud which are defined as follows:

Data consumers: Data consumers are people or companies which are interested in using PHR or EHR data. In other words, data consumers are the healthcare providers, including physicians and nurses.

Controller: The controller is responsible for negotiating and generating session keys in order for them to be used by parties.

Data owner: The data owner is the owner of the data in the proposed system and is the only party who has full access to the EHR data.

Trusted party: The controller and data owner are trusted by all parties in the proposed approaches.

Patients: The patient is the owner of his/her PHR and has complete control over the privacy of his/her PHR information. He/she can delegate his/her patient role to other parties, such as a family member or friend in order to access the PHR in an emergency situation.

Cloud: The cloud consists of the cloud service provider and cloud storage. The cloud service provider responds to the demands from the data consumers and provides corresponding services. The cloud storage is used to store the shared encrypted data from the data owner.

To achieve the goals of the proposed approaches, two main points must be satisfied. Since the patient is the only one with full control over the access to his/her PHR information, the patient's privacy must be ensured in the first place. Second, all data consumers must be able to access an up-to-date version of their EHRs in the cloud at any time and in a secure manner, hence data security must be guaranteed.

The following sections explain the privacy-preserving approach, which can be used to ensure the privacy of the PHRs in eHealth clouds. It then describes the security-preserving approach, which can be applied to ensure the security of the EHRs in eHealth clouds.

6.4.1 Privacy-preserving approach

Authentication is vital for archiving and retrieving information from PHRs (Kaelber et al., 2008). Since PHRs are controlled by patients themselves (Abbas and Khan, 2014), an efficient authentication approach which ensures the privacy of PHRs is required.

In order to ensure the PHRs' privacy, the TPAKE protocol by Khader and Lai (Khader and Lai, 2015) has been adopted.

According to the TPAKE protocol, the primitive root (p) and the generator (g) should be changed in each communication session (Khader and Lai, 2015). This suits the proposed approach as it ensures that the patient is the only one who has complete

access to his PHR, and all data consumers will be revoked after the session. Otherwise, if p and g are not changed, data consumers will be able to access the patient's PHRs using an old session key.

The privacy-preserving approach works as follows:

1. The data owner stores an encrypted PHR at the CS. The data are stored according to the disaster recovery plan in Section 6.6
2. Therefore, the CSP has a copy of the PHR. However, this copy is encrypted and the privacy of the PHR is secured. To make any modification to the PHR, the patient's permission is required. Data are retrieved according to the disaster recovery plan in Section 6.6
3. Data consumers ask the controller to begin a Session Key (SK) negotiation in order to access the PHR
4. The controller will control the communication between the data consumers and the patient. In this step, the controller will ask the patient to initiate the SK agreement
5. The patient will then send the relevant information through the channel back to the controller to authorise the data consumer to gain access to his/her PHR
6. The data consumer calculates the SK using the information received from the controller
7. Data consumers can access the PHR at the CSP once they get permission.

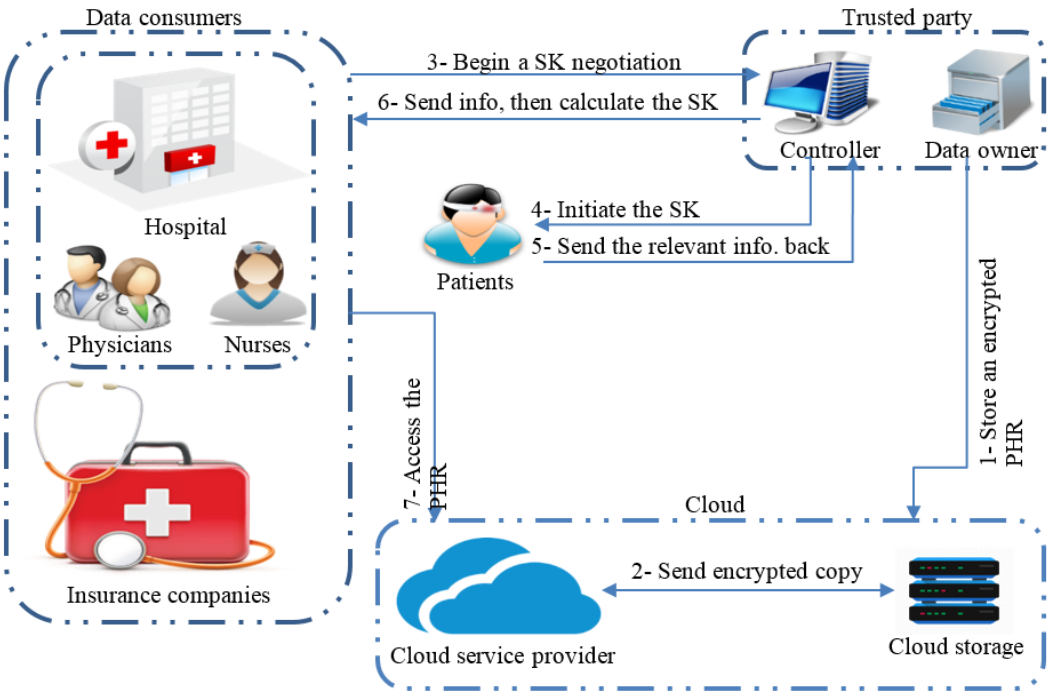


Figure 6.4 Privacy-Preserving approach

6.4.2 Security-preserving approach

EHRs are managed by healthcare providers (Abbas and Khan, 2014; Huang et al., 2009) (in the proposed system, healthcare providers are data consumers). However, patients may have to follow various policies, such as medical, dental, and vision policy or be registered with different insurance companies which make it hard for all parties to access up-to-date EHR records every time. Therefore, the proposed approach shifts the management of the data to the data owner or his delegate, thus ensuring that all the consumers can access an up-to-date version of the EHR in the cloud at any time, and in a secure manner.

The data owner stores the EHRs at their own preferred CC storage servers, and cloud projects must retrieve the EHRs from these servers. Therefore, a robust approach which can ensure the security of the EHRs is required.

The AES encryption algorithm with the PBC mode was adopted to secure the data at the data owner database as well as at the cloud storage (Sahi et al., 2015).

The security-preserving approach works as follows:

1. The data owner stores encrypted EHRs at the CS. Data are stored according to the disaster recovery plan in Section 6.6
2. Therefore, the CSP has a copy of the EHRs. However, this copy is encrypted to ensure the security of the EHRs. To use the EHRs, the CSP needs the data owner's permission to do so. Data are retrieved according to the disaster recovery plan in Section 5.6
3. Data consumers ask the controller to begin the SK negotiation to access the EHRs
4. The controller will control the communication between the data consumers with the data owner. In this step, the controller will ask the data owner to initiate the SK agreement
5. The data owner will send the relevant information through the channel back to the controller in order to authorise the data consumer to gain access to the EHR
6. The data consumer calculates the SK using the information received from the controller
7. Data consumers can access the EHR at the CSP once they get permission.

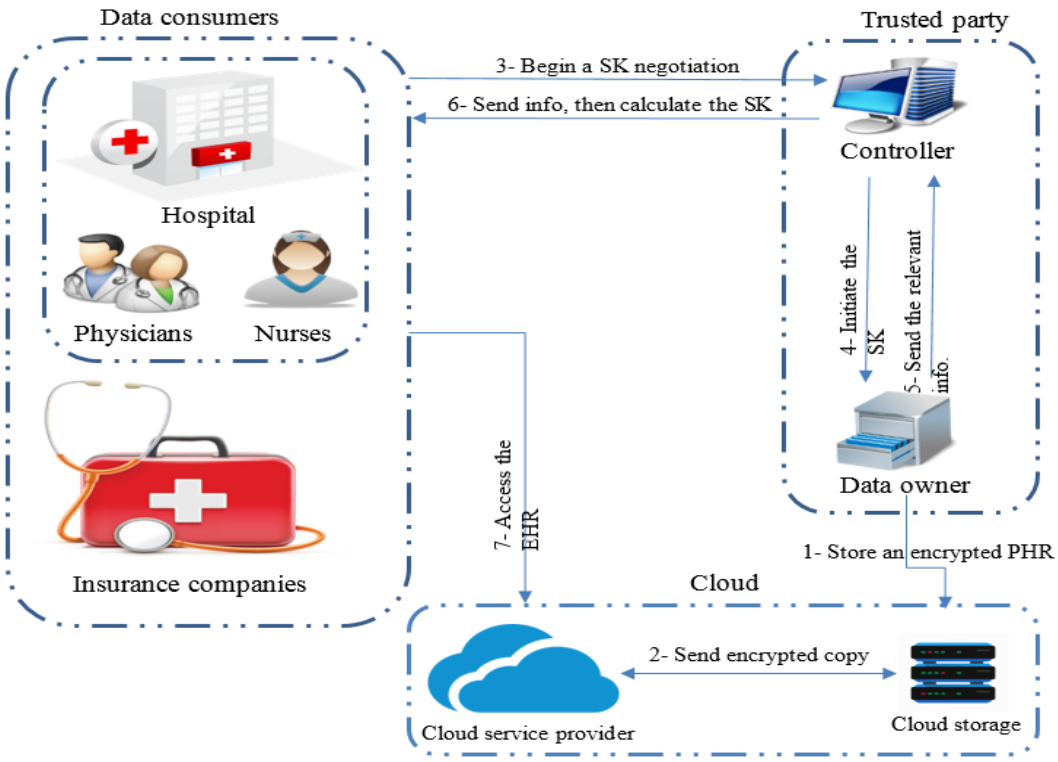


Figure 6.5 Security-Preserving approach

6.4.3 Break-glass access

In an emergency situation, such as in a life-threatening situation for an unconscious patient, healthcare providers may require temporary access to a patient's PHR. Those staff members must have temporary authorisation to decipher the PHR information. While the patient is the only one in this system who has complete control of his/her PHR, as mentioned earlier, the patient can also delegate the role to a family member or friend in advance. Therefore, a family member or friend can play the patient's role to authorise access to the patient's PHR. In the proposed privacy and security preserving approaches this can be achieved by encouraging a patient to delegate an emergency key to a family member or friend when the patient registers for the first time in the system. The process of delegation can be as simple as entering a user name and password, just like using a phone to send a text message. If the patient has difficulty entering the required information, he/she can easily get help from a health practitioner.

6.5 Discussion

To ensure the security and privacy of any cryptography-based system, four requirements must be considered: authentication, non-repudiation, integrity, and confidentiality (Zissis and Lekkas, 2012). This section analyses and evaluates the security of the proposed approaches from three different perspectives: security requirements, comparison with existing work, and when under several kinds of attacks.

6.5.1 Security requirements

CSP should utilise robust authentication and non-repudiation techniques to guarantee authentication and non-repudiation. According to Khader and Lai in (Khader and Lai, 2015), these two features are ensured by the TPAKE key exchange protocol. Since the protocol is employed in this chapter, the proposed privacy-preserving approach inherited the authentication and non-repudiation features from the adopted TPAKE protocol.

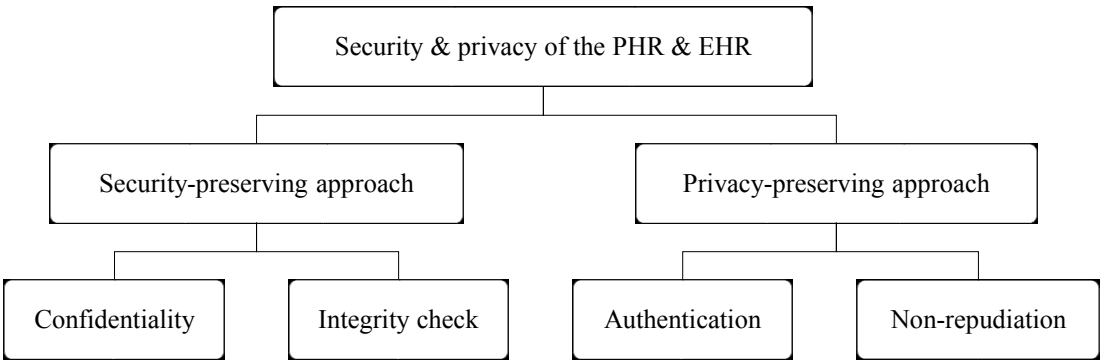


Figure 6.6 Security and privacy of the PHR and the HER

In addition, as mentioned in Chapter 5 (Step 4, decryption algorithm) the PBC-AES can check the integrity of the message by comparing the hash of the copy of the original data with the hash received from the other party.

The public keys are necessary for establishing a MITM attack. In order to begin attacking the system using a MITM attack, the attacker needs to know public keys of both parties, generator and primitive root. However, in the proposed TPAKE protocol all these parameters are encrypted using the user private key. Thus, it can be claimed that the proposed system does mitigate the MITM attack.

Figure 6.6 illustrates the security and privacy features that can be provided by combining the privacy-preserving approach with the security-preserving approach in the cloud environment.

6.5.2 Comparison with existing work

This section compares the proposed approaches with several existing approaches in terms of security, privacy, revocation, break-glass access and DR plan.

Table 6.1 shows the comparison. As shown in the table, the proposed approaches together with the disaster recovery plan achieved all the listed features, whereas some vital features are missing in other approaches. On top of this, some of the other approaches have limitations in various aspects. In the RBTBAC approach (Zhang et al., 2014), each user is required to hold and control multiple keys, which is logically inadequate.

Table 6.1 Comparison of delivered security features

	Proposed approaches	Security	Privacy	Revocation	Break-glass	DR plan
1	Jafari et al. (Jafari et al., 2011)	×	√	×	√	×
2	RBTBAC (Zhang et al., 2014)	√	√	×	×	×
3	Tran et al. (Tran et al., 2011)	√	×	√	×	×
4	New CP-ABPRE (Liang et al., 2015)	√	×	√	×	×
5	Wang et al. (Wang et al., 2012)	×	√	×	×	×
6	Fabian et al. (Fabian et al., 2015)	√	√	×	×	×
7	Chen et al. (Chen et al., 2012)	×	√	√	×	×
8	CP-ABE (Ibraimi et al., 2009)	√	√	×	×	×
9	Zheng (Zhen, 2011)	×	√	√	×	×
10	Proposed approaches	√	√	√	√	√

The approach by Tran et al. and the new CP-ABPRE approach (Liang et al., 2015; Tran et al., 2011) are not effective when considering very large data, and unfortunately very large data is a feature of medical data (Thilakanathan et al., 2014). Furthermore,

the approach by Fabian et al. is not well-suited for emergency cases, as stated by the authors themselves (Fabian et al., 2015). Finally, the approach of Chen et al. gives the ability to manage the PHR to every healthcare staff member in addition to patients and doctors. Apart from the requirement that PHR needs to remain safe at all times and be able to be managed by the patient and his/her doctor (Chen et al., 2012), a revocation feature is needed to cancel the management rights of a doctor later.

6.5.3 When Under Attack

This section shows how the proposed approaches work when the PHR and EHR are under different attacks.

Outside attacks: In outside attacks, an adversary attempts to gain electronic records without any identity or access credentials. However, he can only get the PHR/EHR records in an encrypted format, which he cannot decrypt without an appropriate key. While keys are held only by the patient (and delegate health practitioner) and the controller, obtaining the correct key is impossible. Thus, this approach can claim to prevent outside attacks.

Man-in-the-middle attacks: In the proposed approaches, it is impossible for an adversary to perform a man in the middle attack. The adversary needs to know at least the public keys to perform a man in the middle attack, whereas these public keys are encrypted using private user keys according to the adopted TPAKE protocol (Khader and Lai, 2015). In the TPAKE protocol, only the user (either the patient or controller) can know the public and private keys, so he/she is the only one who can decrypt the electronic records. Thus, the proposed approaches can prevent man-in-the-middle attacks.

Offline and online dictionary attacks: The TPAKE protocol can also be secure against offline and online dictionary attacks. For an adversary to perform an offline dictionary attack, he needs to guess the password of the user to calculate the session key. This is going to fail during the verification process due to the fact that the session key is required for verification of the identity credential in advance.

An integrity check is provided in this protocol, and it helps prevent on-line dictionary attacks using checksum validations. As parties are required to validate one another using a stored checksum, online dictionary attacks can thus be prevented.

The next section investigate a disaster recovery plan in case something unexpected occurs.

6.6 Disaster Recovery Plan

In the event of a disaster, several questions will require answering. For example, how long will the system take to resume service? Therefore, a DR plan must be prepared to answer this and other questions.

First of all, data such as PHRs and EHRs are stored in the cloud storage, which should include a minimum of three nodes or data centres. At the beginning of the process, the data owner will send a heartbeat signal (a heartbeat is a signal sent between the data owner, nodes, and the CSP to check whether those nodes are still in working condition or not) to check the status of the nodes. After that, the data owner asks the controller to break the data records into three partitions (three in this example, but it could vary), distribute and store them between the nodes. When a client attempts to use the data through the CSP, the CSP will send a heartbeat signal to the nodes, and then ask the controller to retrieve the three partitions from the nodes and combine them in one record file.

The controller is responsible for the number of partitions and the size of each. Each partition must be stored in several nodes (in this case, three nodes). The three nodes must be located at three different physical locations to ensure that the cloud project can be continued even in a disaster situation.

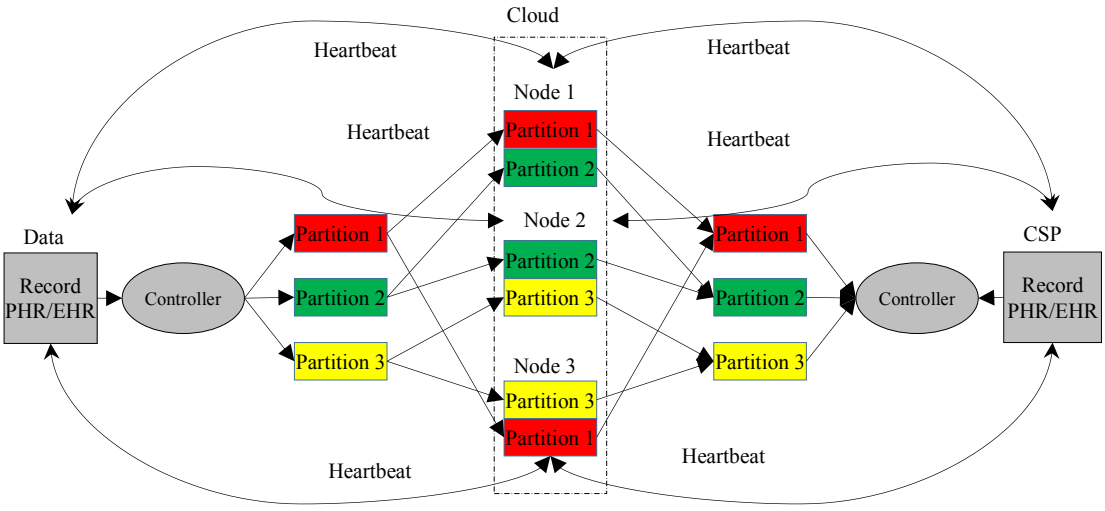


Figure 6.7 The proposed DR plan

For example, let assume that the area where the second node is located has an earthquake which causes complete damage to this node. The original record can be retrieved immediately from node 1 and node 3 with no time wasted or client disconnection. Therefore, this DR plan can answer the above questions and ensure the continuity of the system, as shown in Figure 6.7.

6.7 Chapter Summary

This chapter set up new approaches to harden the security of the EHRs and the privacy of the PHRs in eHealth cloud projects, as well as ensuring the continuity of projects during times of disasters. These two approaches are able to satisfy the most important requirements in cryptosystems, such as authentication, non-repudiation, integrity, and confidentiality. In addition, a DR plan was introduced to enable a client to connect to its system at any time, even at a time of disaster. The two approaches and the DR plan all together provide a private, secure and robust cloud computing environment for the health sector.

As the TPAKE protocol and the PBC mode have been integrated to preserve privacy and security of eHealth clouds, Hypothesis H4 (*Integration of potential key exchange protocol and block cipher mode would preserve privacy and security on eHealth clouds. A disaster recovery plan with heartbeat signals will result in 24/7 services availability*) has been proven in this chapter. In addition, the presented DR plan ensures the connectivity of users 24/7.

To this stage, the proposed system is supposed to be secure against many inside and outside attacks however, it still vulnerable to flood attacks, which is concern of research question 5. Therefore, the next chapter presents a security system that helps to mitigate flood attacks such as Distributed Denial of Service (DDoS) TCP flood attacks. It also presents a method to mitigate the Denial of Service (DoS) attacks in a cloud, reduces excessive energy consumption during attack and ensures the availability of service.

7

CHAPTER 7

MITIGATION OF DDOS TCP FLOOD ATTACKS IN EHEALTH CLOUDS

Chapter 6 presented a security-preserving approach and a privacy-preserving approach based on the PBC mode and the TPAKE protocol. A disaster recovery plan ensuring the connectivity of users in disaster time was also presented (Sahi et al., 2016). This chapter presents two different systems: a classification based security system that can help to mitigate Distributed Denial of Service (DDoS) TCP flood attacks (Sahi et al., 2017 (A)) and an energy efficient DoS attacks mitigation method. Instead of system shutdown, the proposed method ensures availability of service.

7.1 Classification Based System for Detecting and Preventing DDoS TCP flood Attacks

7.1.1 Introducing DDoS Attacks Detection and Prevention

Distributed Denial of Service (DDoS) TCP flood attacks are a kind of DoS attack in which attackers flood a victim machine with packets to exhaust the resources of the victim machine or cause bandwidth consumption (Girma et al., 2016). As the attack may be distributed over multiple machines, it is very hard to differentiate authentic users from attackers. In fact, a DDoS flood attack is not only a widespread attack, it is the second largest cybercrime attack causing financial losses (Cisco, 2018), according to the United States Federal Bureau of Investigation (FBI).

The use of cloud computing is growing quickly in many sectors, especially the health sector, as a result of its vital features such as availability and on demand services (Yan et al., 2016). Most people think of cloud computing as virtual networks which can offer flexible and accessible on demand services (Laplante et al., 2008). Whereas,

Balding (Balding, 2012) pointed out that cloud computing is much more than that, which leads this research to re-consider its security more seriously. Also, as mentioned in an electronic cybercrime study published by KPMG in collaboration with the eCrime Congress in 2009, most of clouds' virtual clients are threaten and that threats grow as time goes (E-CrimeCongress, 2009).

Many procedures (Wang et al., 2015 (D)) can be adopted to detect and/or mitigate DDoS flood attacks. These include classifications used as a detection method (Taravat et al., 2015; Xia et al., 2015), and encryption techniques used as a mitigation method (Khader and Lai, 2015; Sahi et al., 2015; Sahi et al., 2016). As DDoS flood attacks can be implemented in many forms, we cannot foresee the form of attacks. Therefore, this new proposed Classifier System for detecting and preventing DDoS TCP flood attacks (CS_DDoS) is classification based, and can identify the attacks on data no matter the form in which they come to the cloud system. Classification can be defined as a common procedure to classify, distinguish, and differentiate multiple objects. Different classifiers, such as Least Squares Support Vector Machine (LS-SVM), Naïve Bayes, K-nearest-neighbour, and Multilayer perceptron (Acharya et al., 2017; Hameed et al., 2016), are used in this study to perform the classification process.

In the next section, the literature of DDoS security will be reviewed.

7.1.2 Related Work

A DDoS attack is basically a number of DoS attacks running together to attack the same victim. According to Darmohray and Oliver (2000), in a test for the effectiveness of firewall against DoS attacks, a rate of 500 SYN flood per second will cripple a server (Darmohray & Oliver, 2000). In addition, Moore et al. (2006) performed a quantitative study of the prevalence of DoS attacks on the Internet from 2001 to 2004. They found that only 4% of the DoS attack were over 14,000 packets per second and a server can be disabled by a flood of 14,000 packets per second (Moore et al., 2006).

Many detection and prevention methods to mitigate the DDoS flood attacks were reported by Zargar (Zargar, 2013).

The Rank Correlation-based Detection (RCD) scheme was proposed by Wei et al. (Wei et al., 2013). The authors of the RCD claimed that their scheme can distinguish whether the incoming requests are from genuine users or attackers. An ALPi algorithm was introduced by Ayres et al (Ayres et al., 2006), increasing the accuracy of detecting and differentiating attacks by extending the concept of packet scoring. Therefore, the

ALPi raises the percentage of detection accuracy of attacks. Another DDoS attacks prevention architecture, Secure Overlay Services (SOS), was presented by Keromytis (Keromytis et al., 2004). The SOS architecture is a combination of three parts: secure overlay tunnelling, routing via consistent hashing, and filtering. The authors claimed that the SOS can proactively prevent DoS and DDoS attacks using a combination of secure overlay tunnelling, routing via consistent hashing, and filtering.

Moreover, Wang and Reiter (Wang and Reiter, 2010) proposed the Web Referral Architecture for Privileged Service (WRAPS). The WRAPS adopted the structure of a web graph to resist DDoS flood attacks. The WRAPS requires authentic users to be authenticated using a referral hyperlink from a trusted site. Another approach, called the 'group testing based approach', was introduced to detect application DoS attacks on backend servers (Xuan et al., 2010). The authors extended the old group testing approach and reallocated user requests to several servers. The probability theory of Markov Chain was adopted by Salah et al. when proposing an analytical queueing approach which examines the performance of firewalls when under DDoS attacks (Salah et al., 2012).

In addition, Dou et al (Dou et al., 2013) presented a Confidence Based Filtering (CBF) scheme for cloud projects. In the CBF, information about authentic user packets is gathered during the non-attack period to extract features for generating an information profile during non-attack periods. The CBF scheme will decide to remove the packets or not using the information profile. Another approach to detect flood attacks, is the fast lightweight detection approach presented by Yu et al. (Yu et al., 2008). This approach utilises the SNMP-MIB (Simple Network Management Protocol-Management Information Base) statistical data as an alternative to the raw data, as well as a SVM classifier for attack classification. Lee et al (Lee et al., 2008) introduced a practical DDoS detection scheme based on DDoS architecture. In this scheme, they selected variables based on particular features that were extracted from a DDoS architecture. A Cloud Trace Back (CTB) method was proposed in (Chonka et al., 2011). The authors of the CTB claimed that their method could identify the sources of the attacks. Also, they proposed a Cloud Protector (CP) that made use of a back-propagation classifier in order to detect such attacks.

Furthermore, a new framework was presented by Lu et al in (Lu et al., 2007 A). This framework was able to effectively identify the compromised packets. The framework uses a perimeter-based DDoS prevention system in which traffic is analysed only at

the edge routers of an internet service provider (ISP) network. Wang et al. introduced a DDoS attack mitigation architecture that integrates programmable network monitoring to enable attack detection and a flexible control structure to allow a fast and specific attack reaction - (Wang et al., 2015 (D)). In (Khanna et al., 2012), an adaptive selective verification (ASV) system was proposed. The ASV does not need network assumptions, and it utilises bandwidth efficiently. Another approach was presented based on five features (average number of packets per flow, percentage of correlative flow, one direction generating speed, ports generating speed, and percentage of abnormal packets) combined with Bloom filter (Feng et al., 2009). In this approach, only users in the white list are allowed to reach their destinations. This white list is generated to include legitimate users only. However, this approach was implemented on switches side (hardware), making any future amendments or update problematic (Braga et al., 2010).

While there are many mechanisms proposed to detect and prevent the DDoS flood attacks, most of them do not provide high accuracy and are not efficient fast detection and prevention techniques (Bhuyan et al., 2015). Furthermore, many of the presented DDoS attacks protection mechanisms face scalability issues due to the fact that networks are getting larger and faster (Ayres et al., 2006).

Therefore, cloud computing (especially eHealth clouds) needs an efficient DDoS mitigation approach that can offer fast and accurate detection, and is scalable. The proposed CS_DDoS was designed with these factors in mind.

7.1.3 The DDoS TCP Flood Attacks

The DDoS attacks can be established in two different ways: direct and/or indirect (Al-Shaer and Gillani, 2016; Somani et al., 2016). Direct attacks target the weakness in the system of the victim machines and damage the machines directly. On the other hand, indirect attacks are not targeting victim machines directly. They prey on other elements associated with the victim machines and hinder their work (Kumarand Gomez, 2010). The following discussion uses the TCP flood attack which is an indirect attack as it consumes most of the network resources and make them unavailable to other users.

A TCP flood attack was performed using software on a virtual cloud network and Wireshark Network Analyser 2.0.0 (Wireshark, 2017) to capture and analyse the traffic before and during the attack.

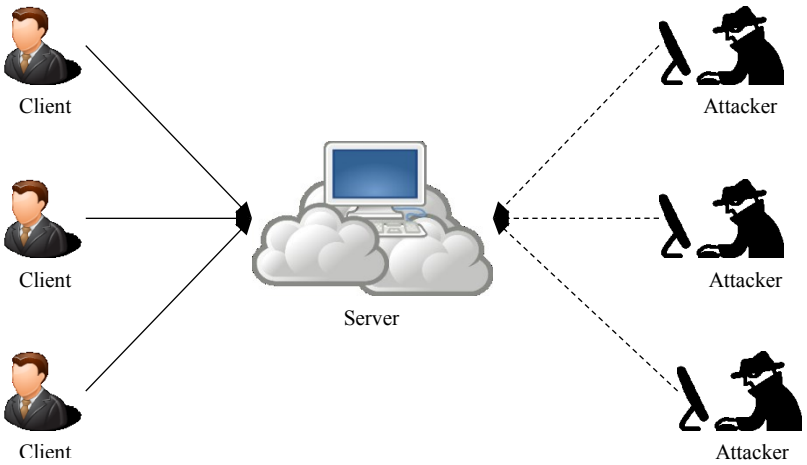


Figure 7.1 Test network architecture

7.1.3.1 Before the attack

A network was simulated as shown in Figure 7.1.

First of all, using TCP Ping, a 50 TCP probes (pings) test was sent to a server (server machine, 10.25.129.5:80), the reply was 1.3ms on average, as shown below:

```
Ping statistics for 10.25.129.5:80
    50 probes sent.
Approximate trip times in milliseconds:
    Minimum = 0.25ms, Maximum = 26.065ms, Average = 1.323ms
```

TCP protocol uses several flags to manage the state of a connection in the packet header (Albanese et al., 2016). Testing focused on two that were used in establishing TCP connections:

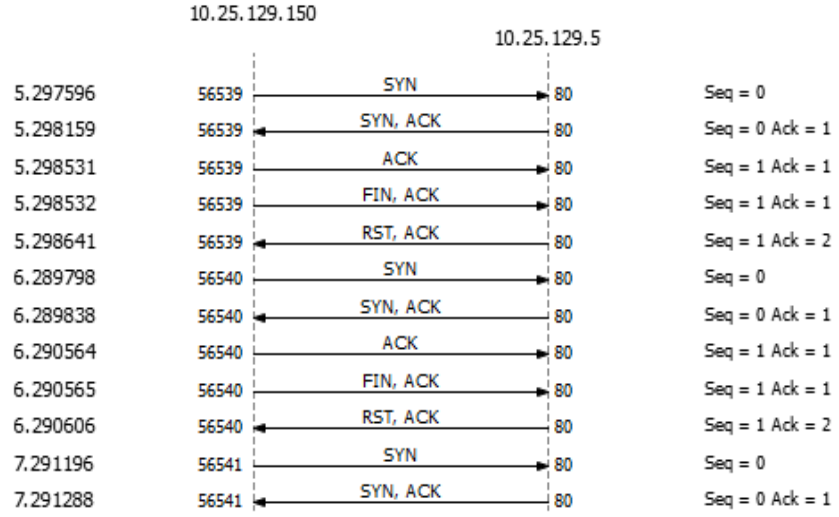
- SYN (Synchronise) which represents the connection initiation
- ACK (Acknowledge) which represents the data receiving.

The traffic of those 50 probes were monitored at the server machine using Wireshark by capturing the packets that were associated with the server using the filter of "ip.addr = = 10.25.129.5". As the traffic was normal, the server machine replied to all requesting packets according to the TCP protocol as shown in Figure 7.2 (a and b).

ip.addr == 10.25.129.5 Expression... + Apply this filt

No.	Time	Source	Destination	Protocol	Length	Info
275	36.355443	10.25.129.5	10.25.129.150	TCP	66	80 → 56570 [SYN, ACK] Seq=0 Ac…
276	36.355652	10.25.129.150	10.25.129.5	TCP	60	56570 → 80 [ACK] Seq=1 Ack=1 W…
277	36.355653	10.25.129.150	10.25.129.5	TCP	60	56570 → 80 [FIN, ACK] Seq=1 Ac…
278	36.355699	10.25.129.5	10.25.129.150	TCP	54	80 → 56570 [RST, ACK] Seq=1 Ac…
279	37.356926	10.25.129.150	10.25.129.5	TCP	66	56571 → 80 [SYN] Seq=0 Win=819…
280	37.357022	10.25.129.5	10.25.129.150	TCP	66	80 → 56571 [SYN, ACK] Seq=0 Ac…
281	37.357418	10.25.129.150	10.25.129.5	TCP	60	56571 → 80 [ACK] Seq=1 Ack=1 W…
282	37.357419	10.25.129.150	10.25.129.5	TCP	60	56571 → 80 [FIN, ACK] Seq=1 Ac…
283	37.357525	10.25.129.5	10.25.129.150	TCP	54	80 → 56571 [RST, ACK] Seq=1 Ac…
287	38.359532	10.25.129.150	10.25.129.5	TCP	66	56572 → 80 [SYN] Seq=0 Win=819…
288	38.359629	10.25.129.5	10.25.129.150	TCP	66	80 → 56572 [SYN, ACK] Seq=0 Ac…
289	38.360030	10.25.129.150	10.25.129.5	TCP	60	56572 → 80 [ACK] Seq=1 Ack=1 W…
290	38.360031	10.25.129.150	10.25.129.5	TCP	60	56572 → 80 [FIN, ACK] Seq=1 Ac…
291	38.360137	10.25.129.5	10.25.129.150	TCP	54	80 → 56572 [RST, ACK] Seq=1 Ac…

(a) Captured packets



(b) TCP flags

Figure 7.2 Captured packets and TCP flags (normal)

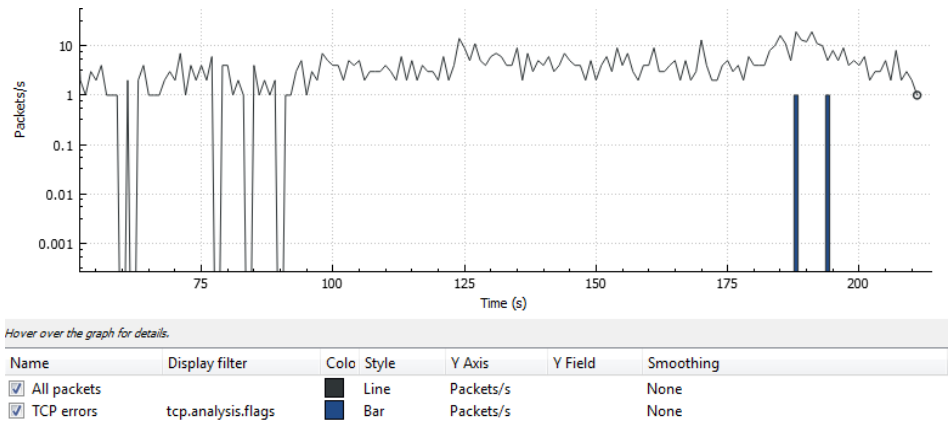


Figure 7.3 The I/O graph (no TCP errors)

In addition, the I/O graph was stable. All packets were answered and almost no TCP errors occurred. Note that the number of requesting packets was approximately less than 10 per second, as shown in Figure 7.3.

7.1.3.2 During the attack

An attack was launched using software, which performed a DDoS TCP flood attack on a particular server. Once the DDoS TCP flood attack on the victim machine in the cloud was commenced, the arriving packets were seen to be much more than the server could handle. Consequently, the server could not respond to all requesting packets, from normal users and the attackers alike. Note that 10.25.129.5 was the IP address of the victim server and 10.31.133.235 was the IP address of the attacker. The first request packet from the attacker was successful as it was treated like a normal requesting packet. The subsequent ones were not successful as the server was too busy and could not respond. A screen shot of the packet capture is shown in Figure 7.4 (a and b). The I/O graph was filled with TCP errors, as shown in Figure 7.5.

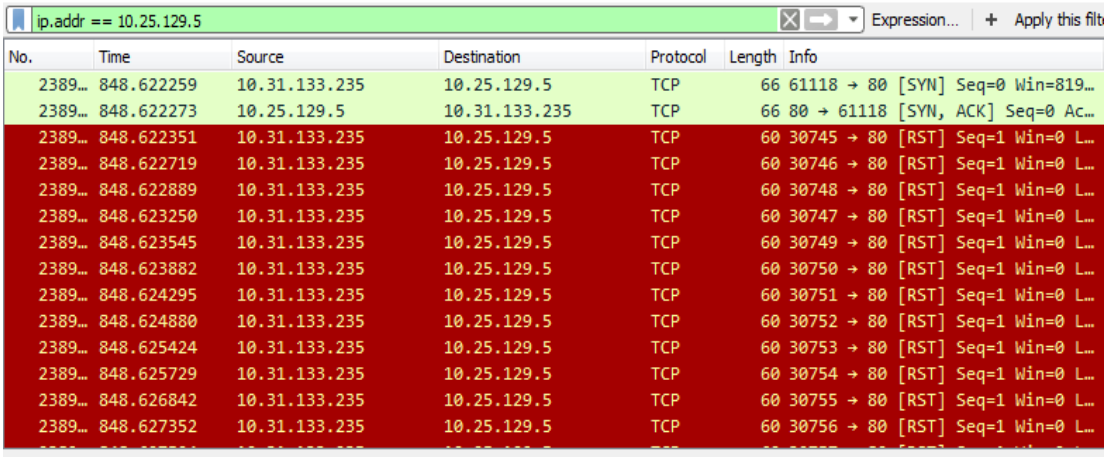
Finally, to test the connection, a 50 TCP probes test was sent within a few seconds to the victim machine during the attacking time. The reply time was in 9.6ms on average, which differs considerably from the first test as shown below:

```
Ping statistics for 10.25.129.5:80
     50 probes sent.
Approximate trip times in milliseconds:
     Minimum = 0.181ms, Maximum = 152.341ms, Average
= 9.586ms
```

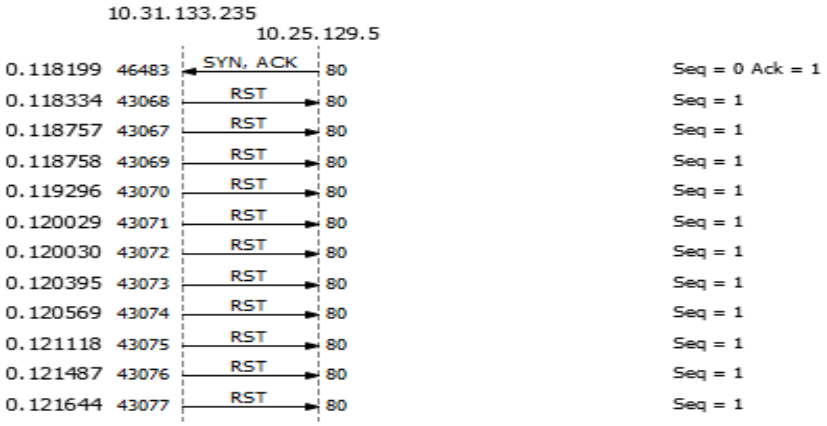
To sum up, the DDoS TCP flood attack affected the cloud server's performance within a short time, slowing down the response and even stopping service completely. TCP errors were also increased.

Therefore, efficient and effective detection and prevention techniques are in demand.

The next section presents the proposed CS_DDoS System.



(a) Captured packets



(b) TCP flags

Figure 7.4 Captured packets and TCP flags (abnormal)

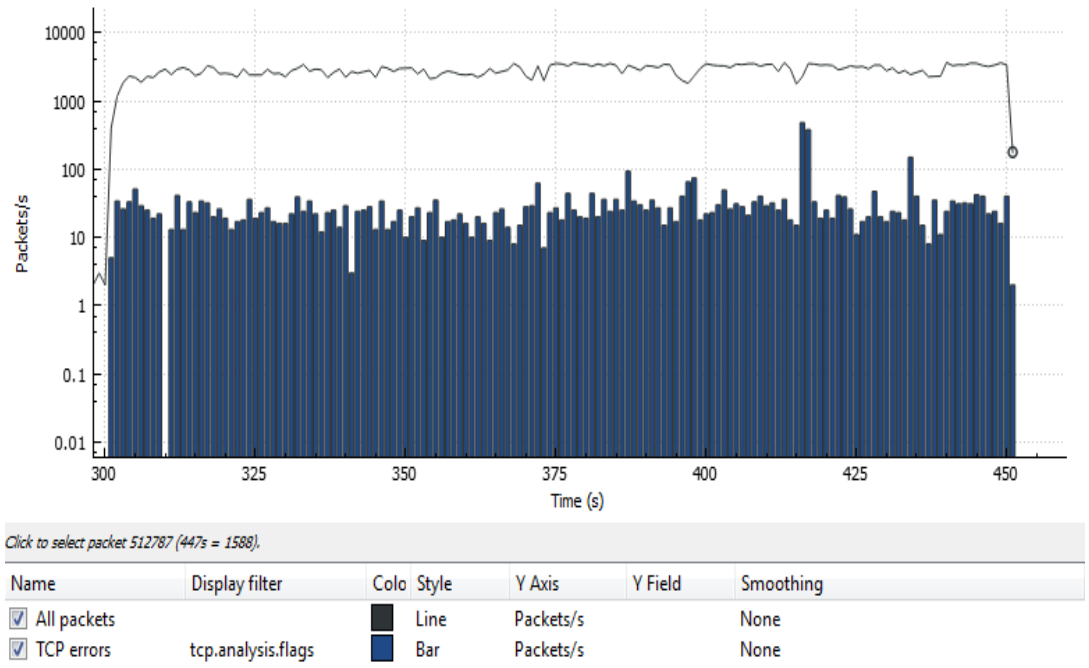


Figure 7.5 The I/O graph (with TCP errors)

7.1.4 The Proposed CS_DDoS System

This section presents the proposed CS_DDoS system which can prevent DDoS TCP flood attacks. First, assume that the IP addresses of the attackers are not spoofed. Examples of how to prevent IP spoofing can be found in (Wang et al., 2016 (A)). The proposed system includes two subsystems: detection sub-system and prevention sub-system, as shown in Figure 7.6.

7.1.4.1 Detection phase

During the detection phase the Detection Sub-system collects the incoming packets within a timeframe; let's say 60 seconds. The collected packets go through a blacklist check to test whether their sources were blacklisted as attackers to the cloud system or not. If the packet source was listed in the attacker blacklist, the detection system will send the packets directly to the Prevention Sub-system without further processing. If the packet source was not blacklisted, the incoming packet will be passed to the classifier to decide whether the packets are normal (coming from a client) or abnormal (coming from an attacker). A packet is considered an attacking one if the source requests connect to the same destination more frequently than an assumed threshold. The threshold can be manually adjusted by the system administrator to cater for the varying requirements of a particular network. If a packet is considered to be normal, the detection system will send it to its destination (the cloud service provider). Otherwise, the Detection Sub-system will send the packet to the Prevention Sub-system.

Four different classifiers have been used in the Detection Sub-System for the classification operation. The classifiers will be explained and evaluated in Section 7.1.5.

7.1.4.2 Prevention phase

While the packets reach the prevention system, they are considered as attacking packets by the Detection Sub-system. The Prevention Sub-system will first alert the system administrator of the attacks. Then, the Prevention Sub-system will add the attacking source address to the attacker blacklist used by the Detection Sub-system if it is not on the list. Finally, the attacker packet will be dropped. The overall architecture of the CS_DDoS system is shown in Figure 7.6.

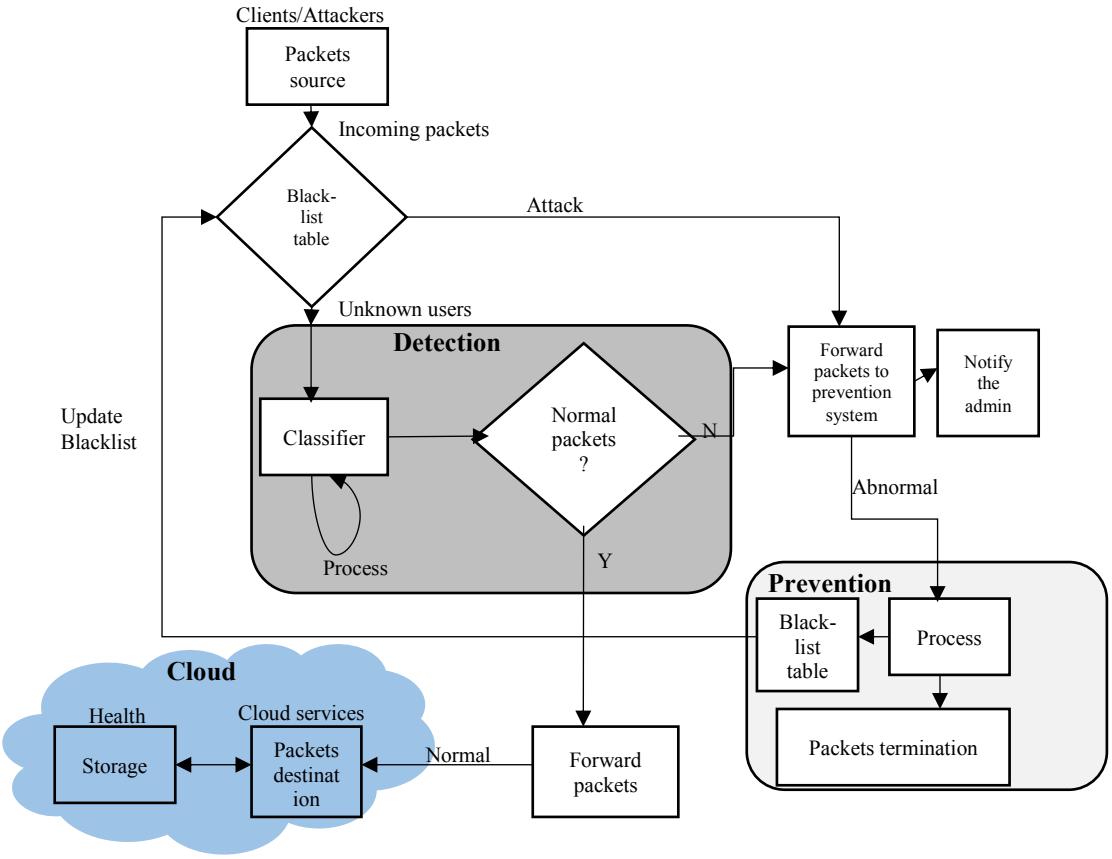


Figure 7.6 The overall architecture of the proposed CS_DDoS system

Algorithm 1 is used to determine whether these packets are normal or abnormal by counting the number of requests for a connection from an IP address and checking if it exceeds a predefined threshold within a certain timeframe. This algorithm is applied to the training data used for each classifier. As a result, each classifier used will predict the behaviour of the attackers according to Algorithm 1.

Algorithm 1: Pre-processing

1: *Load data*
2: *For I=1: n*
3: *P=data (I, 2)*
4: *P2= (I, 1)*
5: *K=100*
6: *For J=1: n*
7: *N=find (data (J, 1) ==P2) & (data (J, 2) ==P)*
8: *If N>=K*
9: *New_data (I, 1) =data (I, 1)*

10: *New_data (I, 2) =-1*

11: *Else*

12: *New_data (I, 1) =data (I, 1)*

13: *New_data (I, 2) =1*

14: *End*

where: **n** is the packets number

P is the destination IP address

P2 is the source IP address

N is the number of packets from the same source to the same destination within 60 seconds

K is the threshold for a packet to be considered as an attacking packet (we will test K with four different values K=100, K=200, K=300 and K=400, see Section 7.1.5.2.1.1)

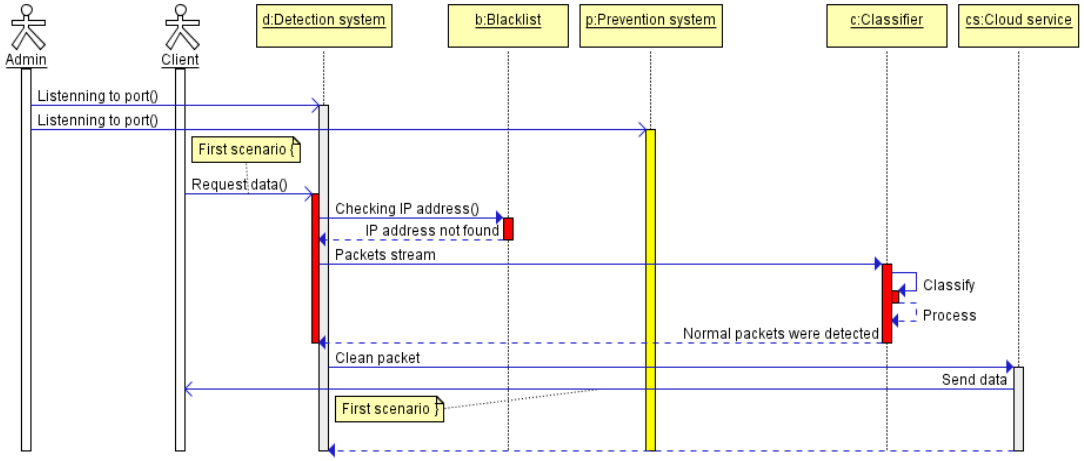
-1 indicates abnormal packets (blacklist array)

1 indicates normal packets

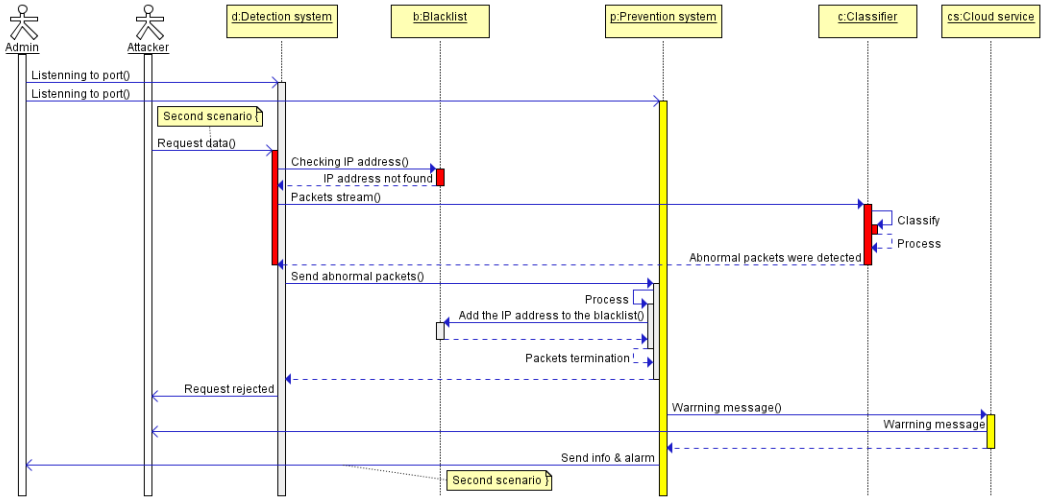
New_data () is a new entry list with tag "1" or "-1"

The proposed CS_DDoS system can be carried out in three possible scenarios. The first scenario is a normal service request packet. The requested service will be delivered as usual. The next scenario is when the source IP address is not blacklisted but the number of services requesting packets exceeds a predefined threshold within a certain timeframe. The packet in this scenario will be considered as a DDoS attack packet. The source address will be blacklisted and the packet will be dropped. The last scenario is when the source address of a packet is blacklisted and the packet is dropped without any further processing.

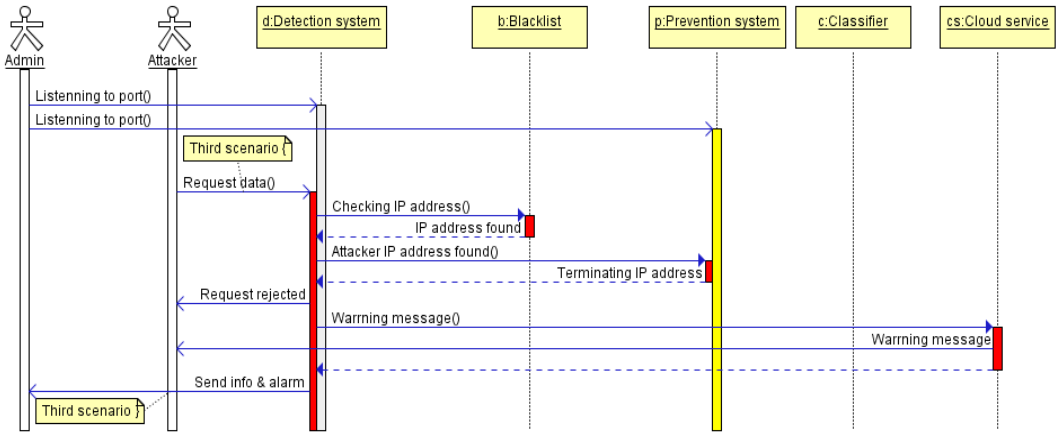
The three scenarios are illustrated in the following sequences diagram. The code used is shown in Appendix C. The resulting sequences diagrams are shown in Figures 6.7 (a, b, and c).



(a) First scenario (normal packets)



(b) Second scenario (store abnormal packets in the blacklist)



(c) Third scenario (abnormal packets that are already in the blacklist)

Figure 7.7 CS_DDoS Possible scenarios

The proposed CS_DDoS system can be used in any clouds, such as eHealth clouds, to ensure the security and availability of health records against the DDoS TCP flood attacks.

7.1.5 Experimental results

7.1.5.1 Classification Algorithms

This section briefly explains the four commonly used classification algorithms that used in the experiments. The classification algorithms are as follows:

7.1.5.1.1 LS-SVM

The LS-SVM is a powerful classifier in the field of pattern recognition for the detection of abnormalities from signals, images and time series signals. The LS-SVM is an efficient classifier for classifying two different sets of observations into their relevant classes. It is capable of handling high dimensional and non-linear data. In this work, the LS-SVM is employed to detect the illegal activities in network. The parameters of the LS-SVM are set during the training session to obtain high detection results (Suykens and Vandewalle, 1999).

7.1.5.1.2 Naïve Bayes

Naïve Bayes is a frequently used classifier and has a straightforward approach based on the application of Bayes' theorem (John and Langley, 1995). It is a simple approach based on the probabilistic knowledge that accurately predicts test instances. This algorithm assumes that the predictive attributes are conditionally independent and there are no hidden attributes which can affect the prediction process (John and Langley, 1995). Naïve Bayes classifier uses small training sets to provide decent performance, which most likely prevents overtraining issues.

7.1.5.1.3 K-nearest-neighbour

K-nearest-neighbour is one of the most straightforward learning algorithms. In this algorithm, the similarity function depends on distance measurements to compute the similarity between training members (Duda and Hart, 1973). The value of k is adjusted during the training session to assign each instance during the training to a correct class. The K-nearest-neighbour classifier is very sensitive to the data size and dimensionality that affect the feature space and homogeneous areas, which represent the distribution of various classes (Depeursinge et al., 2010).

7.1.5.1.4 Multilayer perceptron

The multilayer perceptron is a particular type of neural network based classifier (Lippmann, 1987; Madyastha and Aazhang, 1994). This classifier employs a multilayer feed-forward neural network with one or more layers of nodes between the input and output layers. These nodes at different layers are interconnected through the weighted networks. Using different training algorithms, the parameters (weights) of the networks are optimised. In this classifier, the data are put from input to output. Each feature is used as one of the inputs in the multilayer perceptron, and the outputs are the class categories. The multilayer perceptron can be linear when it is used with one layer of nodes. It can also be a nonlinear perceptron when it is applied with multilayers of nodes with several hidden layers (Duda and Hart, 1973).

7.1.5.2 Performance Evaluation and Validation

This section evaluates and validates the performance of the CS_DDoS system using classification performance measurements and K-Fold-Cross validation.

7.1.5.2.1 Performance evaluation

In this section the performance of the CS_DDoS method is evaluated using the four classifiers of the LS-SVM, Naïve Bayes, K-nearest-neighbour, and Multilayer perceptron. Different training data sizes (window sizes) and different thresholds are used in the experiments. Algorithm 1 is applied to the training data for all the classifiers.

The CS_DDoS system was evaluated in terms of accuracy, sensitivity (detection rate) and specificity (false alarm rate), as well as the descriptive statistic Kappa coefficient. Kappa coefficients are measures used to correlate between categorical variables frequently used as reliability or validity coefficients (Kraemer et al., 2002).

Accuracy represents the rate of the correctly identified results over the entire data used by the CS_DDoS. The correctly identified results can be the True Positives (TP) and the True Negatives (TN), while incorrectly identified results are the False Positives (FP) and the False Negatives (FN). The accuracy of the CS_DDoS system is measured by equation (7.1).

- True Positives (TP): Correctly identified abnormal packets in this research
- False Positives (FP): Incorrectly identified abnormal packets
- True Negatives (TN): Correctly identified normal packets

- False Negatives (FN): Incorrectly identified normal packets

$$CS\_DDoS_{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \times 100\% \quad (7.1)$$

Sensitivity represents the rate of the correctly identified abnormal packets over the entire positive results by the CS_DDoS. The sensitivity of the CS_DDoS system is measured by equation (7.2).

$$CS\_DDoS_{Sensitivity} = \frac{TP}{TP+FN} \times 100\% \quad (7.2)$$

Specificity represents the rate of the incorrectly identified abnormal packets over the entire negative results produced by the CS_DDoS. The specificity of the CS_DDoS system is measured by equation (7.3).

$$CS\_DDoS_{Specificity} = \frac{FP}{FP+TN} \times 100\% \quad (7.3)$$

Kappa coefficients metric will be used to evaluate the performance of the classifiers that use some metrics including true positive percentage and true negative percentage. The main formula of Kappa coefficients is given below:

$$CS_{DDoS_{\text{Kappa coefficients}}} = \frac{\frac{TP+TN}{N} - PRE}{1 - pre}$$

$$pre = \frac{TP+FN}{N}.\frac{TP+FP}{N} + \left(1 - \frac{TP+FN}{N}\right).\left(1 - \frac{TP+FP}{N}\right), \text{ and}$$

$$N = (TP + FP + TN + FN)$$

The proposed CS_DDoS system is evaluated under both single source attack and multiple source attack environments as shown below.

7.1.5.2.1.1 Evaluation under single source attack

Four test data sizes (n) of 1000 packets, 2000 packets, 5000 packets and 6000 packets, and four thresholds (K) of 100 requests, 200 requests, 300 requests, and 400 requests were randomly selected. Algorithm 1 was implemented using MATLAB R2015b and applied to the data according to the window size n, and tested according to the threshold K. Each classifier classified the data using four windows and four thresholds. Results are shown in Tables 6.1-6.5:

Table 7.1 Classification performance measurements (n=1000, and K=100)

		Detection results			
	Classifiers	Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	99.5%	95.3%	96%	0.91
2	Naïve Bayes	80%	92.3%	93%	0.82
3	K-nearest-neighbour	75%	93.5%	95%	0.74
4	Multilayer perceptron	88.3%	95.3%	97%	0.78

Table 7.2 Classification performance measurements (n=2000, and K=200)

		Detection results			
	Classifiers	Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94.6%	94%	96%	0.89
2	Naïve Bayes	82%	93%	94%	0.75
3	K-nearest-neighbour	80%	95%	93%	0.87
4	Multilayer perceptron	92%	97%	97%	0.65

Table 7.3 Classification performance measurements (n=5000, and K=300)

		Detection results			
	Classifiers	Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	96%	98%	97%	0.90
2	Naïve Bayes	96%	94%	92%	0.82
3	K-nearest-neighbour	82%	96%	94%	0.68
4	Multilayer perceptron	95%	99%	97%	0.75

Table 7.4 Classification performance measurements (n=6000, and K=400)

		Detection results			
	Classifiers	Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	98%	99%	98%	0.85
2	Naïve Bayes	95%	95%	96%	0.67
3	K-nearest-neighbour	85%	98%	97%	0.62
4	Multilayer perceptron	97%	99%	97%	0.58

Table 7.5 Classification performance average

		Average			
	Classifiers	Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	97%	97%	97%	0.8875
2	Naïve Bayes	88%	94%	94%	0.765
3	K-nearest-neighbour	81%	96%	95%	0.7275
4	Multilayer perceptron	93%	98%	97%	0.69

Tables 6.1 to 6.4 show the classification performances by the proposed CS_DDoS system with different data sizes and thresholds. The performance measurements are accuracy (correctly detected data over the entire dataset), sensitivity (correctly detected attacks, detection rate), specificity (incorrectly detected attacks, false alarm rate), and Kappa coefficient (stability rate).

According to Tables 6.1 to 6.4, the window sizes and the thresholds did not affect the results of each classifier significantly. There are only small differences between the tables. Tables 6.1 to 6.4 are summed up in Table 7.5.

Table 7.5 shows that the LS-SVM classifier has the highest accuracy average percentage (97%) and the highest Kappa coefficient (0.89). On the other hand, the K-nearest-neighbour classifier made the lowest accuracy percentage at about 81%, and the multilayer perceptron classifier has the lowest Kappa coefficient 0.69. To sum up, the proposed CS_DDoS system is more effective and stable in single source attacks when adopting the LS-SVM classifier regardless of the window size and the threshold.

7.1.5.2.1.2 Evaluation under multiple source attacks

To evaluate the performance under attacks from multiple sources, the same four window sizes (1000, 2000, 5000, and 6000) and the same four thresholds (100, 200, 300, and 400) were used. Algorithm 1 is also used. Results are shown in Tables 6.6-6.10:

Tables 6.6 to 6.9 show the classification accuracy results of the proposed CS_DDoS system when under multiple DDoS attacks. Tables 6.6 to 6.9 again show that the window sizes and thresholds did not significantly affect the results of each classifier.

Table 7.6 Classification performance measurements (n=6000, and K=400)

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	98%	93%	94%	0.91
2	Naïve Bayes	82%	91.3%	91%	0.82
3	K-nearest-neighbour	80%	91.5%	92%	0.74
4	Multilayer perceptron	83.3%	92.3%	95%	0.78

Table 7.7 Classification performance measurements (n=6000, and K=400)

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	93%	91%	94%	0.91
2	Naïve Bayes	85%	92%	95%	0.81
3	K-nearest-neighbour	79%	96%	92%	0.82
4	Multilayer perceptron	87%	95%	96%	0.71

Table 7.8 Classification performance measurements (n=6000, and K=400)

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94%	97%	95%	0.92
2	Naïve Bayes	95%	92%	94%	0.85
3	K-nearest-neighbour	88%	90%	92%	0.69
4	Multilayer perceptron	89%	97%	93%	0.81

Table 7.9 Classification performance measurements (n=6000, and K=400)

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	92%	97%	94%	0.87
2	Naïve Bayes	91%	93%	95%	0.65
3	K-nearest-neighbour	87%	91%	96%	0.69
4	Multilayer perceptron	94%	97%	94%	0.60

Table 7.10 Classification performance average

	Classifiers	Average			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94%	95%	94%	0.9025
2	Naïve Bayes	88%	92%	94%	0.7825
3	K-nearest-neighbour	84%	92%	93%	0.735
4	Multilayer perceptron	88%	95%	95%	0.725

LS-SVM is again the best performing classifier with accuracy percentage around 94%, and Kappa coefficient about 0.9. These experiments used different criteria to assess classifiers' performance including: Kappa coefficients, sensitivity, specificity and accuracy. All these criteria were used to identify the most effective classifier.

Based on the simulation results, the SVM gained the highest scores of Kappa coefficients, sensitivity, specificity and accuracy among the four classification algorithms. As a result, it was considered to be a robust classifier for detecting DDoS attacks.

To sum up, the proposed CS_DDoS system is also affective and stable when resisting multiple source attacks of up to 30 attackers as well as single source attacks when adopting the LS-SVM classifier regardless of the window size and threshold. Therefore, the proposed CS_DDoS system can be implemented in a big cloud project, such as a health cloud, as well as smaller projects like a private cloud for a medium size company. CS_DDoS can prevent the DDoS attacks with 94% accuracy and is highly stable (Kappa coefficient 0.9). CS_DDoS outperforms previous approaches as either the accuracy percentage of previous approaches are less than that which CS_DDoS can achieve, for example 91% in (Moradi and Zulkernine, 2004). Thus, the proposed CS_DDoS system is more effective.

To shed more light on the performance evaluation of the proposed CS_DDoS system, the simulation was repeated with different numbers of attackers (source IP) under similar conditions, and performance measurements were calculated.

Figure 7.8 shows the performance of CS_DDoS based on the number of attackers. There are slight fluctuations in the performance measurements of all four classifiers. Nevertheless, LS-SVM was still the best performer among the four.

The process complexity times of the four classification algorithms shown in Figure 7.9. While LS-SVM is only the second less time consuming, the fastest classifier, K-nearest-neighbour, has lower performance measurements and smaller Kappa coefficient compared to LS-SVM, therefore the LS-SVM is the most efficient and effective classifier to be adopted by CS_DDoS system to resist the DDoS TCP flood attacks.

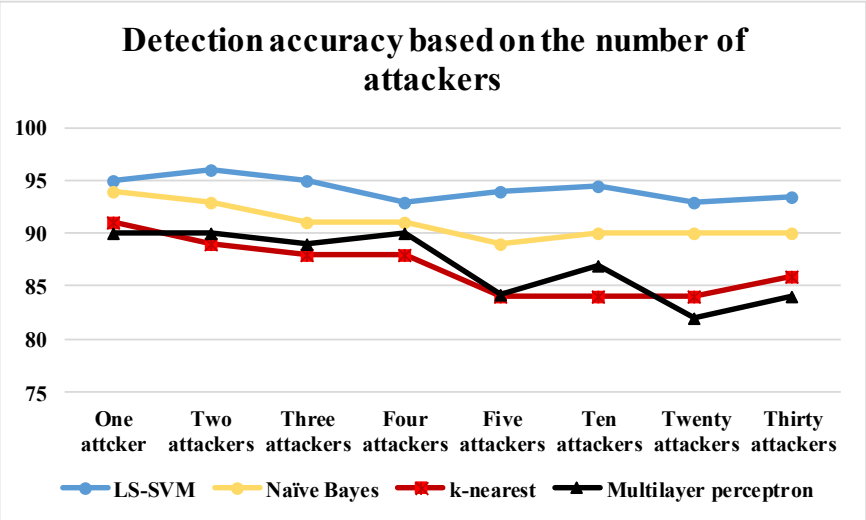


Figure 7.8 Multiple attacks detection accuracy

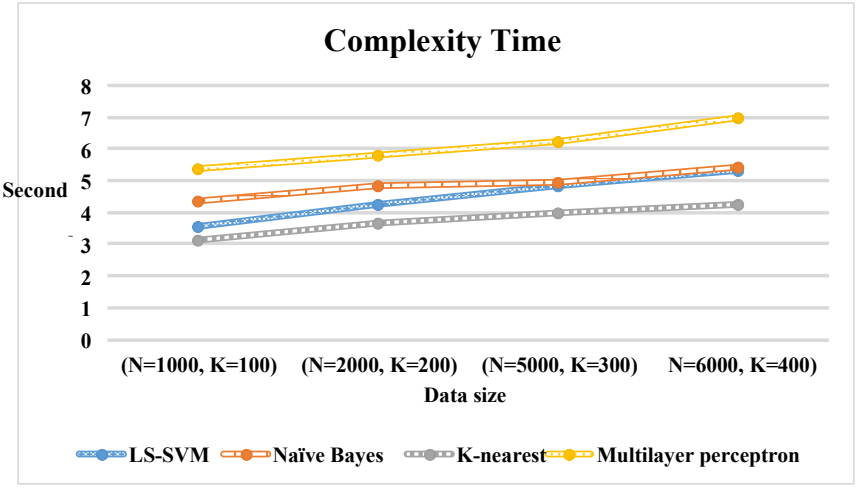


Figure 7.9 Complexity times

7.1.5.2.2 K-Fold-Cross validation

In K-Fold-Cross validation, the original sample is randomly partitioned into k equal sized subsamples, a single subsample is retrained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. Generally, it is utilised to validate the performance accuracy estimation in the practice of a predictive model (Allen, 1974; Geisser, 1975; McLachlan et al., 2005; Stone, 1974).

The K-Fold-Cross validation was adopted to conduct a performance comparison of the four predictive modelling algorithms used in CS_DDoS: LS-SVM, Naïve Bayes, K-nearest-neighbour, and Multilayer perceptron. These four algorithms were compared for their prediction results.

The dataset was divided into 6 equal sized chunks, k=6. As a validation for model testing, one of the 6 chunks was retained, and the rest of chunks (5 chunks) were considered as training data. Then the process of the 6-cross model was repeated 6 times; with each of the 6 chunks used as validation data for each occasion. The results are shown in Figure 7.10. The values of all folds are almost the same, which means that every single fold has approximately the same rate of the four classification algorithms.

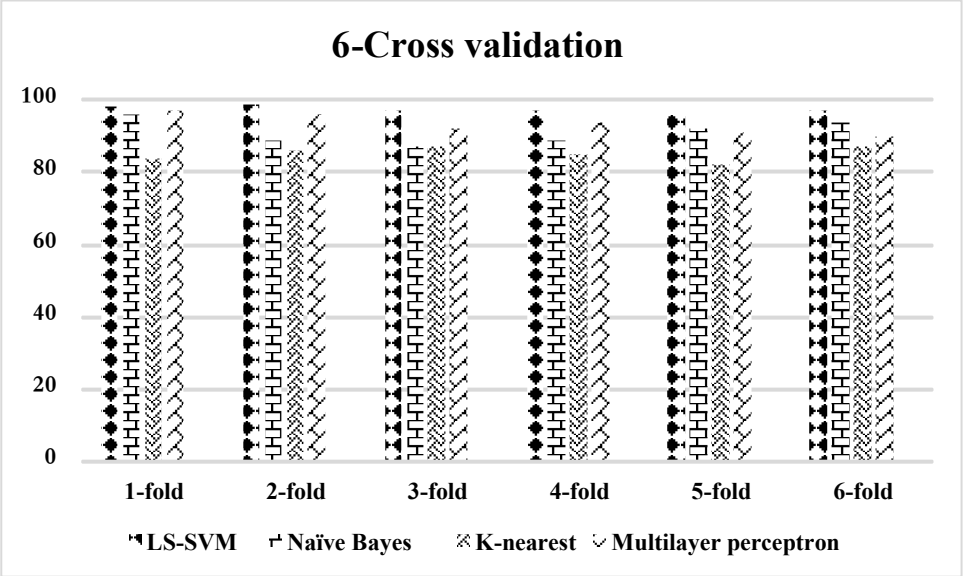


Figure 7.10 6-Fold-Cross validation diagram

Thus, the classification results are stable and accurate, as each algorithm has almost the same result in each fold.

7.2 An Energy Efficient TCP DoS Attacks Mitigation Method

7.2.1 Introducing DoS Attacks Energy Consumption

The importance of cloud computing has been increasing significantly within the last decade due to the ever increasing number of services and facilities that clouds can deliver. In addition, cloud computing services are in extremely high demand from organisations and individuals as a result of benefits the services can offer; such as better computing power, inexpensive operation costs, good performance, and high availability. On the other hand, cloud computing projects have some security issues that must be addressed (Sahi et al., 2016). One of the most concerning issues is Denial of Service (DoS) attacks (Nezhad et al., 2016). There are many types of DoS and distributed DoS attacks. Many of these are mentioned in (Shameli-Sendi et al., 2015). One type is the Transmission Control Protocol (TCP) DoS flood attack, which is discussed in this chapter. TCP DoS attacks can flood the victim's network with extremely large numbers of packets to shut the network down and prevent legitimate users from accessing the services, to exhaust the resources and cause overheating, and to consume more energy than expected. In 2013 there was an incident of overheating Microsoft's data centre which led to a 16 hour shutdown in Microsoft's cloud services such as SkyDrive, Hotmail and Outlook (Anwar et al., 2014). Therefore, developing

an efficient DoS mitigation method that can keep the services available 24/7 and reducing excessive energy consumption is an important research topic. This chapter proposes an energy-efficient TCP DoS attack mitigation method that can be implemented in the cloud environment. The proposed method is designed to keep the incoming packet number under the number of packets the server can handle. Thus, this method can improve the services for cloud projects by protecting them against the packet floods, as well as reducing cost and energy consumption.

The next section presents the energy efficient TCP DoS attacks mitigation method.

7.2.2 The Proposed Method

This section discusses the proposed DoS mitigation method and the cloud architecture used in the experiment. The cloud includes: (1) data storage, a single server point (*Server*); (2) several other points that represent the legitimate points (*Client*); (3) spurious points (*Attacker*); and (4) TCP packets ($TCP_{Packets}$). Assume that the *Attacker* uses a real, not faked or spoofed IP. Due to time constraints, the case of spoofed IP has been left for future studies. It is also assumed that $T_{Packets}$ is the rate of $TCP_{Packets,}$ which *Server* can handle within a time frame *Time*, and $L_{Packets}$ is the total rate of the legitimate $TCP_{Packets,}$ that the *Client* points send to the server. The following equation is a further assumption that the rate of the legitimate packets is at most half of the handling capacity of the *Server* to allow for the bursts of requests:

$$L_{Packets} = T_{Packets} / 2 \tag{7.4}$$

In addition, it is assumed that together with the legitimate packets, the *Attacker* is able to send spurious packets to make $TCP_{Packets}$ more than the *Server* can handle. The *Attacker* spurious packets (unauthentic packets) is denoted as $S_{Packets}$, such that:

$$L_{Packets} + S_{Packets} > T_{Packets} \tag{7.5}$$

The proposed method aims to keep the sum of $L_{Packets}$ and $S_{Packets}$ smaller than $T_{Packets}$ to ensure the availability of a cloud project, reduce energy consumption, and decrease the cost of the cloud maintenance:

$$L_{Packets} + S_{Packets} \leq T_{Packets} \tag{7.6}$$

In case of a normal traffic when no attack is made on the cloud,

$$S_{Packets} = 0 \tag{7.7}$$

From equation (7.5), $L_{Packets} = T_{Packets} / 2$, equation (7.6) is satisfied for the normal traffic scenario.

On the other hand, in the case of abnormal traffic when the attack is performed on the cloud, a method (∏) is proposed which will drop the $TCP_{Packets}$ of the most frequent requesting IP addresses in order to ensure that $L_{Packets}$ + $S_{Packets}$ does not exceed $T_{Packets}$. Most frequent requesting IP addresses can be filtered using any packet filtering method such as iptables (Diekmann et al., 2016).

For example, suppose that the server can handle 10 packets per *Time*, and there are six sources of {*attacker* | *Client*} initiating a connection with the *Server*. The number of packets that are sent by the sources are as follows: $source_1$=1 packet, $source_2$=2 packets, $source_3$=3 packets, $source_4$=4 packets, $source_5$=5 packets, and $source_6$=6 packets. Therefore, the number of packets that are sent from sources 1 to 6 is about 21 packets, while the server can handle a maximum of 10 packets within *Time*. In this case, ∏ will use a filter to determine which sources are the most frequent requesting IP addresses, and then ∏ will drop them. In the example, $source_5$ and $source_6$ have 5+6=11 packets, and the rest of the packets are 1+2+3+4=10 packets. Intuitively, $source_5$ and $source_6$ that have relatively higher request rates, are more likely the TCP DoS attacker sources than other sources with less frequent requesting rate. Even though there is a possibility that $source_5$ and $source_6$ could be *Client* sources, ∏ can keep the request rate below $T_{Packets}$, and the service is still available. Furthermore, using ∏ will save energy during a DoS attack as shown in the performance evaluation section below.

7.2.3 Performance Evaluation

The proposed method was tested using one *Server*, one *Attacker*, and eight legitimate *Client*s.

TCP DoS attacks were established from the *Attacker* (IP_2) point against the *Server* (IP_1) point alongside the legitimate TCP connection requests from *Clients* (IP_3 to P_{10}). According to Tables 6.11 and 6.12, the *Attacker* sent 47807 requests and received only 21130 responses, and the *Server* received 47893 requests and made 21219 responses.

Table 7.11 Overall packets statistics

		Sources	Destinations
N	Valid Packets	69235	69235
n	TCP Packets	68997	68997
	Missing	0	0

Table 7.12 Sources and destination statistics

IP addresses	Sources		Destinations	
	Number of packets	Percent	Number of packets	Percent
(IP_1) *Server*	21219	30.6	47893	69.2
(IP_2) *Attacker*	47807	69.1	21130	30.5
(IP_3) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_4) *Client*	< 60	< 0.08	< 60	< 0.08
(IP_5) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_6) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_7) *Client*	< 60	< 0.08	< 60	< 0.08
(IP_8) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_9) *Client*	< 60	< 0.08	< 60	< 0.08
(IP_{10}) *Client*	< 30	< 0.04	< 30	< 0.04
Total	69235	100.0	69235	100.0

Wireshark Network Analyser 2.0.0 (Wireshark, 2017) was used to capture the packets traversing the experimental network and no missing packets were discovered. As shown in Table 7.11, 99.7% of the traffic with 68997 packets were used in TCP protocol. In Tables 7.11 and 7.12, sources indicate that the source of a packet and destinations present the destination of a packet.

Let the probability of an accepted $TCP_{Packets}$ within the time frame of *Time* be ε_1. The number of all the incoming packets during *Time* is denoted as *n*. The number of $TCP_{Packets}$ which are expected to be sent to the *Server* during *Time* must not exceed $T_{Packets} \cdot Time$. Thus, every single $TCP_{Packets}$ has the following probability of being accepted by the *Server*:

$$\varepsilon_1 . n \leq T_{Packets} . Time \tag{7.8}$$

$$\therefore \varepsilon_1 \leq \frac{T_{Packets} . Time}{n} \tag{7.9}$$

In the case of normal traffic when no attack is performed against the cloud, all $TCP_{Packets}$ must be accepted by the *Server*. As a result, the *Server* accepts the $TCP_{Packets}$ with the following probability:

$$\varepsilon_1 = \begin{cases} 1, & if\ n \leq T_{Packets} . Time \\ \frac{T_{Packets} . Time}{n}, & if\ n > T_{Packets} . Time \end{cases} \tag{7.10}$$

As a result, this can be expressed as follows:

$$n = 68997, \quad T_{Packets} = 21219, \quad and\ Time = 3.2$$

$$\because n > T_{Packets} . Time$$

$$\therefore \varepsilon_1 = \frac{T_{Packets} . Time}{n}, \text{ from (6.10)}$$

$$\varepsilon_1 = \frac{21219 \cdot 3.2}{68997} \simeq 0.984 \ \blacksquare$$

Thus, the proposed method $\prod$ can process 98% of the accepted packets during an attack.

As mentioned above, that the $TCP_{Packets}$ of the most frequent requesting IP addresses will be dropped to ensure that $L_{Packets}$ + $S_{Packets}$ won't exceed $T_{Packets}$. However, this means that $L_{Packets}$ which represents legitimate users packets could be dropped as well, and not only $S_{Packets}$ has the potential to be dropped. Therefore, $\prod$ must be justified in terms of the energy consumption. Whether the benefits of $\prod$ are sufficient enough when compared with the loss of $L_{Packets}$ must also be investigated. If $\prod$ drops $L_{Packets}$ from the traffic, it costs energy of receiving (R_{energy}) these packets by the *Server*, denoted by $\prod_{cost}$. Whereas, if $\prod$ drops $S_{Packets}$ from the traffic, it saves energy by not processing the packets (P_{energy}), as well as saving the energy S_{energy} by not sending packets from a *Server* to *Client*, denoted by $\prod_{benefit}$. According to the authors in (Chen et al., 2002; Feeney and Nilsson, 2001), S_{energy} is more than R_{energy}. Consequently, the relationship between R_{energy} and P_{energy} + S_{energy} can be formulated as follows:

$$P_{energy} + S_{energy} > R_{energy} \quad (7.11)$$

$$\prod_{cost} = n \,.\, (1 - \varepsilon_1).\frac{L_{Packets}}{L_{Packets} + S_{Packets}} \,.\, \left(R_{energy}\right) \quad (7.12)$$

$$\prod_{benefit} = n \,.\, (1 - \varepsilon_1).\frac{S_{Packets}}{L_{Packets} + S_{Packets}} \,.\, (P_{energy} + S_{energy}) \quad (7.13)$$

$$\prod_{profit} = \prod_{benefit} - \prod_{cost} \quad (7.14)$$

where $\prod_{profit}$ is the net energy saved when $\prod$ is in action.

In the case of no attacks $n \leq T_{Packets} \,.\, Time$, from (6.10) the probability of packets acceptance will be $\varepsilon_1 = 1$, therefore the profit of $\prod$ will be calculated as follows:

$$\prod_{cost} = n \,.\, (1 - 1).\frac{L_{Packets}}{L_{Packets} + S_{Packets}} \,.\, \left(R_{energy}\right) = 0$$

$$\prod_{benefit} = n \,.\, (1 - 1).\frac{S_{Packets}}{L_{Packets} + S_{Packets}} \,.\, (P_{energy} + S_{energy}) = 0$$

$\therefore \prod_{profit} = \prod_{benefit} - \prod_{cost} = 0$, from (11)

On the other hand, in the case of attacks $n > T_{Packets} \,.\, Time$, from (6.10) the probability of packets acceptance will be $\varepsilon_1 = \frac{T_{Packets} \,.\, Time}{n}$, therefore the profit of $\prod$ will be calculated as follows:

$$\prod_{cost} = n \,.\, \left(1 - \frac{T_{Packets} \,.\, Time}{n}\right).\frac{L_{Packets}}{L_{Packets} + S_{Packets}} \,.\, \left(R_{energy}\right) \quad (7.15)$$

$$\therefore \prod_{benefit} = n \,.\, \left(1 - \frac{T_{Packets} \,.\, Time}{n}\right).\frac{S_{Packets}}{L_{Packets} + S_{Packets}} \cdot (P_{energy} + S_{energy})$$

$\because P_{energy} + S_{energy} > R_{energy}$, from (6.11)

$$\therefore \prod_{benefit} = n \,.\, \left(1 - \frac{T_{Packets} \,.\, Time}{n}\right).\frac{S_{Packets}}{L_{Packets} + S_{Packets}} \cdot (R_{energy}) \cdot Z \quad (7.16)$$

$where\ Z > 1$

$$\therefore \prod_{profit} = \frac{n \cdot (S_{energy} + R_{energy})}{L_{Packets} + S_{Packets}} \cdot \left(1 - \frac{T_{Packets} \,.\, Time}{n}\right) \cdot (Z - 1) \cdot R_{energy} \quad (7.17)$$

$$\because Z > 1$$

$$\therefore \prod_{profit} > 0 \ \blacksquare$$

As a result, dropping $TCP_{Packets}$ of the most frequently requesting IP addresses when under attack will always save more energy than not dropping.

To recap, this method handles the security problem in cloud computing and proposes a method to mitigate TCP DoS attacks by reducing excessive energy consumption via limiting the number of packets. Instead of system shutdown, the proposed method ensures the availability of service.

7.3 Chapter summary

This chapter has presented a new approach called CS_DDoS for detection and prevention of the DDoS TCP flood attacks based on classification to ensure security and availability of stored data, especially eHealth records in emergency cases. The results showed that using LS-SVM, the CS_DDoS system can identify attacks accurately; at about 97% accuracy with about 0.89 Kappa coefficient when under single attack and 94% accuracy with about 0.9 Kappa coefficient when under multiple attacks. The performance was validated using the K-Fold validation and shown to be stable and accurate. Thus, the proposed approach can efficiently improve health records' security, reduce bandwidth consumption and mitigate resources exhaustion.

The chapter also presented an energy efficient TCP based DoS attacks' mitigation method to enhance the security of the cloud and at the same time save energy. The method maintains service availability by controlling the number of server processed TCP packets to below the number of packets that the server can handle. This method reduces the energy consumption in the case of a DoS attack thereby reducing the chance of shutting down services due to such attacks.

Hypothesis H5 (*Classification techniques can detect DDoS attacks in eHealth clouds, and filtering techniques can reduce excessive energy consumption caused by DoS attacks*) has been proven in this chapter. The LS-SVM classifier successfully classified packets and detected potential DDoS attacks on eHealth clouds. Packets filtering was also proven to reduce excessive energy consumption caused by DoS attacks.

We will conclude this thesis and discuss future work in Chapter 8.

8

CHAPTER 8

CONCLUSIONS AND DIRECTIONS FOR FUTURE WORK

There is no doubt that personal health records require a high level of security and privacy. The proliferation and extensive use of eHealth clouds have highlighted serious security and privacy concerns related to sensitive health data, and practical and effective methods are therefore required to preserve the privacy and security of health data. This chapter summarises the methods and protocols introduced in this thesis, and a number of unsolved issues are discussed as possible future research directions.

8.1 Summary and Conclusions of the Thesis

This thesis consists of eight chapters. Chapter 1 is the introduction. Chapter 2 is an overview of cloud computing, in terms of its security and privacy and some necessary background knowledge. It briefly introduced information and knowledge related to the current research on eHealth cloud security, privacy and associated issues. The chapter then summarised the state-of-the-art security and privacy issues related to the cloud. Chapter 3 described some research design methodologies and the methodology adopted in this research project. The adopted Vaishnavi and Kuechler research design methodology and its five phases are described to provide an overall picture of the process used in this thesis. In Chapter 4, identified a gap in the security of key exchange between parties, and presented a new TPAKE protocol that securely distributes keys between users and protects systems against MITM attacks, among others. Next, to enhance the speed performance of encryption, the PBC mode was introduced in Chapter 5. In the PBC mode, blocks of cipher can be processed in parallel to ensure both high performance and security. Furthermore, in Chapter 6 with the

security and privacy for eHealth cloud in mind, the PBC mode is integrated with the TPAKE protocol to form a security-preserving approach and a privacy-preserving approach for eHealth clouds. In addition, Chapter 6 also presented a disaster recovery plan which ensures uninterrupted connectivity for users during disasters. Chapter 7 presented a classification-based security system that helps to detect and prevent DDoS TCP flood attacks. This chapter also introduced a method for mitigating DoS attacks in the cloud and reducing excessive energy consumption by limiting the number of packets. This method was evaluated with more than 14,900 attacks per second and the server could still provide services (no missing packets). Finally, Chapter 8 presents a summary and the findings of this study. Possible directions for future work are also discussed in this chapter.

Table 8.1 Research questions and hypotheses

	Research questions	Research hypotheses
1	Has a researcher in this area reviewed the major and relevant literature on cloud security and privacy?	Investigation and review of the relevant literature on eHealth cloud security and privacy would facilitate the researcher's decisions on research directions
2	How can the security of key exchange between parties be hardened, and different types of attacks prevented?	Developing a key exchange protocol based on the Computational Diffie-Hellman assumption should secure key distribution between parties and protect systems against multiple attacks
3	How can the performance of the adopted encryption algorithm be improved on top of security assurance?	A new parallel hash-based block cipher mode of operation would be able to improve the encryption process on top of security assurance
4	How can security and privacy be preserved in eHealth clouds? How can a client be enabled to connect to the system at any time, even during a disaster?	Integration of potential key exchange protocol and block cipher mode would preserve privacy and security on eHealth clouds. A disaster recovery plan with heartbeat signals will result in 24/7 services availability
5	How can DDoS TCP flood attacks be detected and prevented? How can the energy consumption caused by DoS attacks be reduced?	Classification techniques can detect DDoS attacks in eHealth clouds, and filtering techniques can reduce excessive energy consumption caused by DoS attacks.

Research questions identified and corresponding research hypotheses are listed side by side in table 8.1.

According to the experiments and analyses undertaken in Chapters 4 to 7, hypotheses H1 to H5 have been proven and research questions (RQs) RQ1 to RQ5 have been answered.

Hypothesis H1 has been proven and RQ1 has been answered in Chapter 2 through the investigation and review of the relevant literature on eHealth cloud security and privacy.

Hypothesis H2 has been proven and RQ2 has been answered in Chapter 4. TPAKE key exchange protocol has been developed based on the Computational Diffie-Hellman assumption. Analysis and tests showed that the TPAKE protocol is a secure key distribution protocol and can protect systems against multiple attacks.

In the Chapter 5 experiments, the PBC mode was shown to save 60% of the execution time when compared with the CBC mode on top of security assurance. Therefore, hypothesis H3 has been proven and RQ3 has been answered.

Hypothesis H4 has been proven and RQ4 has been answered in Chapter 6. The TPAKE protocol and the PBC mode have been integrated. According to the experiment in Chapter 6, this integration preserves the privacy and security of eHealth clouds. In addition, the presented DR plan will ensure the connectivity of users 24/7.

Hypothesis H5 has been proven and RQ5 has been answered in Chapter 7. The LS-SVM classifier successfully classified packets and detected potential DDoS attacks on eHealth clouds. Packets filtering was also shown to reduce excessive energy consumption caused by DoS attacks.

The following objectives were formulated to help the researcher to answer the research question:

1. To investigate and review the state of the art in privacy and security in eHealth clouds, in order to fully explore current research directions
2. To develop a TPAKE protocol based on the CDH assumption to securely distribute keys between parties, and to protect systems against multiple attacks

3. To introduce an efficient hash-based PBC mode of operation to increase the performance of the encryption process while maintaining an assurance of security
4. To integrate Objectives 2 and 3 into security and privacy-preservation approaches in eHealth clouds, and to ensure the connectivity of users during disasters
5. To provide a classification-based system for detecting and mitigating DDoS TCP flood attacks in eHealth cloud environments and an energy-efficient TCP DoS attack mitigation method.

To accomplish these objectives, the state of the art regarding security and privacy in eHealth clouds from five main perspectives: security and privacy, security controls, effective encryption, data security requirements and disaster recovery plans, was investigated and reviewed. This study therefore provides a clear overall picture of the development of eHealth to stakeholders, to facilitate better designs and decisions. In summary, this study collects, evaluates, and classifies state-of-the-art eHealth security and privacy schemes. It covers the most recent studies in this area, and discusses the drawbacks of the existing proposals, in order to help improve the security and privacy of eHealth clouds.

Next, the TPAKE protocol based on the CDH assumption, and a hash-based PBC mode of operation were developed which contribute to cloud security.

The 3PAKE protocol enables the negotiation of a session key between two clients through a trusted server over a public channel. Clients only have to present a single password to commence negotiating with another partner through a server. Most of the 3PAKE protocols proposed here use public keys to guarantee users' identities. However, sharing public keys may lead to various types of attacks, such as man-in-the-middle attacks, which allow an attacker to simply intercept and insert traffic into a network. A formal analysis of the TPAKE protocol in terms of its robustness is therefore provided. TPAKE is an enhanced three-party password-based authenticated key exchange protocol based on the CDH assumption and the random Oracle model. The proposed protocol performs better than other 3PAKE protocols, since it never shares clear, plain information through insecure channels. A security analysis using the random Oracle model shows that the

proposed protocol provides mutual authentication, a safe and secure session key, perfect forward secrecy and prevention against multiple attacks.

Block cipher encryption works on fixed-length blocks, usually of 128 bits. The blocks of data are transformed into encrypted data blocks of identical size, using a shared session key. A common feature of some modes of block cipher (such as the cipher block chaining mode (CBC), cipher feedback mode (CFB) and output feedback mode (OFB)) is sequential processing. The ciphering process of a block cannot begin until the processing of the preceding block is completed. This feature does not make full use of the processing power in multiple processor systems. Thus, a hash-based PBC mode of operation that considerably increases the speed of the encryption process since it processes data in parallel, is presented. In this mode, each cipher block makes use of the characteristics of the entire file (the hash value of the file) during encryption, rather than just the previous cipher block as in the CBC, to improve the randomness of the key stream. This mode offers high levels of security and better speed compared to the standard CBC mode. The PBC mode was shown to save 60% in terms of execution time when compared with the CBC mode. Furthermore, the hash value of the data file can be used to provide an integrity check, as well as encryption using AES128. As a result, the PBC mode has better performance in terms of speed while maintaining the confidentiality and security provided by the CBC mode.

Cloud computing has been introduced as an alternative storage and computing model within the health sector, as in other sectors, to handle large amounts of data. To reduce in-house storage, IT development and maintenance costs, many healthcare companies have moved their electronic data to the cloud. However, storing healthcare records on a third-party server may create serious storage, security and privacy issues, and numerous approaches have been proposed to preserve both security and privacy in cloud computing projects. Cryptographic-based approaches have been reported to be one of the best ways to ensure the security and privacy of healthcare data in the cloud. Nevertheless, the cryptographic-based approaches used to transfer health records safely remain vulnerable in terms of security and privacy and lack disaster recovery strategies.

This research study therefore aims to establish new approaches to hardening the security of EHRs and the privacy of PHRs in cloud projects, as well as ensuring the continuity of projects during disaster events. A privacy preservation

approach is proposed based on the TPAKE key exchange protocol that ensures the privacy of PHRs, and a security preservation approach is proposed based on the PBC that ensures the security of EHRs.

These two approaches are able to meet the most important requirements of every cryptosystem, such as authentication, non-repudiation, integrity and confidentiality. In addition, a disaster recovery plan is proposed to enable a client to connect to the system at any time, even during an ongoing disaster. A combination of these two approaches and the disaster recovery plan provide a private, secure and robust cloud computing environment for the health sector.

Even with the disaster recovery plan, and the privacy preserving and security preserving approaches in place, cloud projects are vulnerable to many types of attack such as DDoS TCP flood attacks. A new approach has been proposed, entitled CS_DDoS, for the detection and prevention of DDoS TCP flood attacks. It is based on packet classification and ensures the security and availability of stored data, and particularly eHealth records, in cases of emergency. In this approach, the incoming packets are classified to determine the behaviour of the source within a given timeframe, to discover whether the source is associated with a genuine client or an attacker. The results show that using LS-SVM, the CS_DDoS system can identify DDoS attacks with about 97% accuracy and a Kappa coefficient of about 0.89 when under single attack, and 94% accuracy with a Kappa coefficient of about 0.9 when under multiple attacks. The performance is validated using the k-fold cross-validation and is shown to be stable and accurate. Thus, the proposed approach can efficiently improve the security of health records, reduce bandwidth consumption and mitigate resource exhaustion.

Finally, an energy-efficient TCP-based DoS attack mitigation method to enhance the security of the cloud while saving energy was proposed. This method maintains the availability of the service by controlling the number of TCP packets to below the threshold number of packets that can be handled by the server. This method reduces energy consumption in the case of a DoS attack, thereby reducing the chance of service shut-down due to such attacks.

It can be concluded that this research study has established a new, successful suite of security protocols, algorithms and approaches for securing, authenticating and providing 24/7 availability of cloud services, such as eHealth cloud services. This security suite can enable cloud users to store and share their

data in a secure manner. The outcomes will help healthcare stakeholders' access efficient, secure and uninterrupted cloud services.

8.2 Future Work

This research uses cryptography and classification techniques to tackle security issues in cloud computing, based on an analytic reading of and critical reflection on the relevant literature. The security suite proposed in this thesis can deliver secure and robust practice in cloud security, and privacy. However, this work is far from perfect. To enable an additional improvement to the methods presented here, a few key issues are highlighted and discussed below.

The TPAKE protocol was built based on the CDH assumption. The proposed protocol is valid while this assumption holds. In the near future, with the rapid development of computer processing power, the CDH assumption may become invalid. The feasibility of this protocol is tied to the CDH assumption, and developing another protocol based on a stronger assumption will be the next challenge.

In addition, the PBC mode was tested in a virtual network to encrypt files of up to 5GB. Consideration of future testing in a real network and using a cloud environment with larger files to increase confidence in the adoption of this block cipher mode in the cloud and eHealth industries is necessary.

The proposed CS_DDoS system must be tested using a larger number of attacking machines; ideally more than 30 attacking machines. A patient usability test and an attacker penetration test are also priorities for future works.

Finally, the CS_DDoS system was developed based on another assumption, which is that IP addresses of the attackers are not faked or spoofed. Since real world attacks use spoofed IPs, the CS_DDoS system needs to be linked to an IP spoofing detection and prevention mechanism to provide maximum protection against flood attacks.

In summary, the proposed security suite can successfully secure cloud projects and efficiently ensure service availability, although it is far from perfect and more research work needs to be done in the future.

REFERENCES

Abbas, A & Khan, SU 2014, 'A review on the state-of-the-art privacy-preserving approaches in the e-health clouds', *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 4, pp. 1431-41.

Acharya, UR, Fujita, H, Adam, M, Lih, OS, Sudarshan, VK, Hong, TJ, Koh, JE, Hagiwara, Y, Chua, CK & Poo, CK 2017, 'Automated characterization and classification of coronary artery disease and myocardial infarction by decomposition of ECG signals: a comparative study', *Information Sciences*, vol. 377, pp. 17-29.

Adrian, D, Bhargavan, K, Durumeric, Z, Gaudry, P, Green, M, Halderman, JA, Heninger, N, Springall, D, Thomé, E & Valenta, L 2015, 'Imperfect forward secrecy: How Diffie-Hellman fails in practice', in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*: *Proceedings of the* ACM, pp. 5-17.

Ahmed, M, Sanjabi, B, Aldiaz, D, Rezaei, A & Omotunde, H 2012, 'Diffie-Hellman and Its Application in Security Protocols', *International Journal of Engineering Science and Innovative Technology (IJESIT)*, vol. 1, pp. 69-73.

Albanese, M, Battista, E & Jajodia, S 2016, 'Deceiving Attackers by Creating a Virtual Attack Surface', in *Cyber Deception*, Springer, pp. 169-201.

Allen, DM 1974, 'The relationship between variable selection and data agumentation and a method for prediction', *Technometrics*, vol. 16, no. 1, pp. 125-7.

Alomari, MA, Samsudin, K & Ramli, AR 2009, 'A parallel XTS encryption mode of operation', in *Research and Development (SCOReD), 2009 IEEE Student Conference on*: *proceedings of the* IEEE, pp. 172-5.

Al-Shaer, E & Gillani, SF 2016, 'Agile virtual infrastructure for cyber deception against stealthy ddos attacks', in *Cyber Deception*, Springer, pp. 235-59.

Amin, R & Biswas, G 2015, 'Cryptanalysis and design of a three-party authenticated key exchange protocol using smart card', *Arabian Journal for Science and Engineering*, vol. 40, no. 11, pp. 3135-49.

Amin, R, Islam, SH, Biswas, G, Khan, MK, Leng, L & Kumar, N 2016, 'Design of an anonymity-preserving three-factor authenticated key exchange protocol for wireless sensor networks', *Computer networks*, vol. 101, pp. 42-62.

Andreeva, E, Bogdanov, A, Luykx, A, Mennink, B, Tischhauser, E & Yasuda, K 2013, 'Parallelizable and authenticated online ciphers', in *Advances in Cryptology-ASIACRYPT 2013*, Springer, pp. 424-43.

ANSI, I 2003, 'TS 18308 health informatics-requirements for an electronic health record architecture', *ISO (Ed.)*.

Anwar, Z & Malik, AW 2014, 'Can a DDoS attack meltdown my data center? A simulation study and defense strategies', *IEEE Communications Letters*, vol. 18, no. 7, pp. 1175-8.

Attrapadung, N, Herranz, J, Laguillaumie, F, Libert, B, De Panafieu, E & Ràfols, C 2012, 'Attribute-based encryption schemes with constant-size ciphertexts', *Theoretical Computer Science*, vol. 422, pp. 15-38.

Ayres, PE, Sun, H, Chao, HJ & Lau, WC 2006, 'ALPi: A DDoS defense system for high-speed networks', *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 10, pp. 1864-76.

Balding, C 2012, 'What everyone ought to know about cloud security', *http://www. slideshare. net/craigbalding/what-everyone-oughtto-know-a bout-cloud-security*.

Beeputh, AK, Doomun, MR & Dookee, P 2010, 'Energy-Security Adaptation Scheme of Block Cipher Mode of Operations', in *Innovations and Advances in Computer Sciences and Engineering*, Springer, pp. 73-8.

Bellare, M & Rogaway, P 1993, 'Random oracles are practical: A paradigm for designing efficient protocols', in *Proceedings of the 1st ACM conference on Computer and communications security*: *proceedings of the* ACM, pp. 62-73.

Bellare, M & Rogaway, P 1995, 'Provably secure session key distribution: the three party case', in *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*: *proceedings of the* ACM, pp. 57-66.

Bellare, M, Pointcheval, D & Rogaway, P 2000, 'Authenticated key exchange secure against dictionary attacks', in *Advances in Cryptology—EUROCRYPT 2000*: *proceedings of* Springer, pp. 139-55.

Bellovin, SM & Merritt, M 1992, 'Encrypted key exchange: Password-based protocols secure against dictionary attacks', in *Research in Security and Privacy, 1992. Proceedings., 1992 IEEE Computer Society Symposium on*: *proceedings of the* IEEE, pp. 72-84.

Bertino, E, Shang, N & Wagstaff Jr, SS 2008, 'An efficient time-bound hierarchical key management scheme for secure broadcasting', *IEEE Transactions on dependable and secure computing*, vol. 5, no. 2, pp. 65-70.

Bhuyan, MH, Bhattacharyya, D & Kalita, JK 2015, 'An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection', *Pattern Recognition Letters*, vol. 51, pp. 1-7.

Bichsel, P, Camenisch, J, Dubovitskaya, M, Enderlein, RR, Krontiris, I, Lehmann, A, Neven, G, Nielsen, JD, Paquin, C & Preiss, F-S 2013, 'H2. 2–ABC4Trust architecture for developers', *ABC4Trust heartbeat H*, vol. 2, p. 2.

Bichsel, P, Camenisch, J, Dubovitskaya, M, Enderlein, R, Krenn, S, Krontiris, I, Lehmann, A, Neven, G, Nielsen, JD & Paquin, C 2014, 'D2. 2 Architecture for attribute-based credential technologies-final version', *ABC4TRUST project deliverable*.

Biswas, S, Akhter, T, Kaiser, M & Mamun, S 2014, 'Cloud based healthcare application architecture and electronic medical record mining: An integrated approach to improve healthcare system', in *Computer and Information Technology (ICCIT), 2014 17th International Conference on* IEEE, pp. 286-91.

Braga, R, Mota, E & Passito, A 2010, 'Lightweight DDoS flooding attack detection using NOX/OpenFlow', in *Local Computer Networks (LCN), 2010 IEEE 35th Conference on* IEEE, pp. 408-15.

Bresson, E, Chevassut, O & Pointcheval, D 2007, 'Provably secure authenticated group Diffie-Hellman key exchange', *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 3, p. 10.

Burrows, M, Abadi, M & Needham, RM 1989, 'A logic of authentication', in *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* The Royal Society, pp. 233-71.

Camenisch, J, Krontiris, I, Lehmann, A, Neven, G, Paquin, C & Rannenberg, K 2012, 'H2. 1—ABC4Trust Architecture for Developers', *Heartbeat*, vol. 2, p. 1.

Canetti, R, Goldreich, O & Halevi, S 2004, 'The random oracle methodology, revisited', *Journal of the ACM (JACM)*, vol. 51, no. 4, pp. 557-94.

Carlson, FR 2014, 'Security analysis of cloud computing', *arXiv preprint arXiv:1404.6849*.

Carroll, M, Van Der Merwe, A & Kotze, P 2011, 'Secure cloud computing: Benefits, risks and controls', in *proceedings of the Information Security South Africa (ISSA), 2011* IEEE, pp. 1-9.

Cash, D, Kiltz, E & Shoup, V 2009, 'The twin Diffie–Hellman problem and applications', *Journal of Cryptology*, vol. 22, no. 4, pp. 470-504.

Castiglione, A, Pizzolante, R, De Santis, A, Carpentieri, B, Castiglione, A & Palmieri, F 2015, 'Cloud-based adaptive compression and secure management services for 3D healthcare data', *Future Generation Computer Systems*, vol. 43, pp. 120-34.

Chakraborty, D & Sarkar, P 2008, 'HCH: A new tweakable enciphering scheme using the hash-counter-hash approach', *Information Theory, IEEE Transactions on*, vol. 54, no. 4, pp. 1683-99.

Chang, V 2015, 'Towards a Big Data system disaster recovery in a Private Cloud', *Ad Hoc Networks*, vol. 35, pp. 65-82.

Chase, M 2007, 'Multi-authority attribute based encryption', in *Theory of Cryptography Conference* Springer, pp. 515-34.

Chen, B, Jamieson, K, Balakrishnan, H & Morris, R 2002, 'Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks', *Wireless networks*, vol. 8, no. 5, pp. 481-94.

Chen, C-L, Chen, Y-Y, Lee, C-C & Wu, C-H 2014, 'Design and analysis of a secure and effective emergency system for mountaineering events', *The Journal of Supercomputing*, vol. 70, no. 1, pp. 54-74.

Chen, D & Zhao, H 2012, 'Data security and privacy protection issues in cloud computing', in *2012 International Conference on*: *proceedings of the Computer Science and Electronics Engineering (ICCSEE),* IEEE, pp. 647-51.

Chen, T-S, Liu, C-H, Chen, T-L, Chen, C-S, Bau, J-G & Lin, T-C 2012, 'Secure dynamic access control scheme of PHR in cloud computing', *Journal of medical systems*, vol. 36, no. 6, pp. 4005-20.

Chevalier, Y, Küsters, R, Rusinowitch, M & Turuani, M 2008, 'Complexity results for security protocols with Diffie-Hellman exponentiation and commuting public key encryption', *ACM Transactions on Computational Logic (TOCL)*, vol. 9, no. 4, p. 24.

Chmura Kraemer, H, Periyakoil, VS & Noda, A 2002, 'Kappa coefficients in medical research', *Statistics in medicine*, vol. 21, no. 14, pp. 2109-29.

Choi, C, Choi, J & Kim, P 2014, 'Ontology-based access control model for security policy reasoning in cloud computing', *The Journal of Supercomputing*, vol. 67, no. 3, pp. 711-22.

Chonka, A, Xiang, Y, Zhou, W & Bonti, A 2011, 'Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks', *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1097-107.

Chung, H-R & Ku, W-C 2008, 'Three weaknesses in a simple three-party key exchange protocol', *Information Sciences*, vol. 178, no. 1, pp. 220-9.

Cisco Systems. 2018. Cisco 2018, 'Annual Cybersecurity Report', Technical Report.

Creswell, J 2014, 'Research Design: Qualitative, Quantitative, and Mixed Methods Approaches', *(4th edition)*, London: Sage Publications.

Creswell, JW & Creswell, JD 2017, 'Research design: Qualitative, quantitative, and mixed methods approaches', *(5th edition),* London: Sage Publications.

Dansereau, RM, Jin, S & Goubran, RA 2006, 'Reducing Packet Loss in CBC Secured VoIP using Interleaved Encryption', in *Electrical and Computer Engineering, 2006. CCECE'06. Canadian Conference on*: *proceedings of the* IEEE, pp. 1320-4.

Darmohray, T & Oliver, R 2000, *hot spares" for dos attacks,; login*, The Magazine of Usenix and SAGE.

Deebak, BD, Muthaiah, R, Thenmozhi, K & Swaminathan, P 2015, 'Evaluating Three Party Authentication and Key Agreement Protocols Using IP Multimedia Server–Client Systems', *Wireless Personal Communications*, vol. 81, no. 1, pp. 77-99.

Depeursinge, A, Iavindrasana, J, Hidki, A, Cohen, G, Geissbuhler, A, Platon, A, Poletti, P-A & Müller, H 2010, 'Comparative performance analysis of state-of-the-art classification algorithms applied to lung tissue categorization', *Journal of digital imaging*, vol. 23, no. 1, pp. 18-30.

Diekmann, C, Michaelis, J, Haslbeck, M & Carle, G 2016, 'Verified iptables firewall analysis', in *IFIP Networking Conference (IFIP Networking) and Workshops, 2016* IEEE, pp. 252-60.

Diffie, W & Hellman, M 1976, 'New directions in cryptography', *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644-54.

Dou, W, Chen, Q & Chen, J 2013, 'A confidence-based filtering method for DDoS attack defense in cloud environment', *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1838-50.

Dou, W, Zhang, X, Liu, J & Chen, J 2015, 'HireSome-II: Towards privacy-aware cross-cloud service composition for big data applications', *IEEE transactions on parallel and distributed systems*, vol. 26, no. 2, pp. 455-66.

Duda, R & Hart, P 1973, 'Pattern classification and scene analysis'. Wiley, New York.

Dworkin, M 2001, *Recommendation for block cipher modes of operation. methods and techniques*, DTIC Document.

Dworkin, M 2007, *Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC*.

Dworkin, M 2010, 'Recommendation for block cipher modes of operation: three variants of ciphertext stealing for CBC mode'. (A)

Dworkin, M 2010, 'Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on', *Storage Devices ", NIST Special Publication*, vol. 800. (B)

E-CrimeCongress.E-CrimeSurvey 2009, /http://www.e-crimecongress.org/ecrime2009/ documents/e-CrimeSurvey2009_AKJ_KPMG(1).pdf

Edgar, TW & Manz, DO 2017, *Research Methods for Cyber Security*, United States, Syngress.

Ehrsam, WF, Meyer, CH, Smith, JL & Tuchman, WL 1978, *Message verification and transmission error detection by block chaining*, Google Patents.

El-Gazzar, R, Hustad, E & Olsen, DH 2016, 'Understanding cloud computing adoption issues: A Delphi study approach', *Journal of Systems and Software*, vol. 118, pp. 64-84.

Elminaam, DSA, Abdual-Kader, HM & Hadhoud, MM 2010, 'Evaluating the performance of symmetric encryption algorithms', *IJ Network Security*, vol. 10, no. 3, pp. 216-22.

Fabian, B, Ermakova, T & Junghanns, P 2015, 'Collaborative and secure sharing of healthcare data in multi-clouds', *Information Systems*, vol. 48, pp. 132-50.

Farash, MS & Attari, MA 2013, 'An enhanced authenticated key agreement for session initiation protocol', *Information Technology and Control*, vol. 42, no. 4, pp. 333-42.

Farash, MS & Attari, MA 2014, 'An efficient and provably secure three-party password-based authenticated key exchange protocol based on Chebyshev chaotic maps', *Nonlinear Dynamics*, vol. 77, no. 1-2, pp. 399-411. (A)

Farash, MS & Attari, MA 2014, 'An enhanced and secure three-party password-based authenticated key exchange protocol without using server's public-keys and symmetric cryptosystems', *Information Technology and Control*, vol. 43, no. 2, pp. 143-50. (B)

Fazackerley, S, McAvoy, SM & Lawrence, R 2012, 'Gpu accelerated aes-cbc for database applications', in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*: *proceedings of the* ACM, pp. 873-8.

Feeney, LM & Nilsson, M 2001, 'Investigating the energy consumption of a wireless network interface in an ad hoc networking environment', in *the INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings.* IEEE, pp. 1548-57.

Felici, M & Pearson, S 2015, 'Accountability for data governance in the cloud', in *Accountability and Security in the Cloud*, Springer, pp. 3-42.

Feng, Y, Guo, R, Wang, D & Zhang, B 2009, 'Research on the active DDoS filtering algorithm based on IP flow', in *Natural Computation, 2009. ICNC'09. Fifth International Conference on* IEEE, pp. 628-32.

Fernández-Alemán, JL, Señor, IC, Lozoya, PÁO & Toval, A 2013, 'Security and privacy in electronic health records: A systematic literature review', *Journal of biomedical informatics*, vol. 46, no. 3, pp. 541-62.

Fernando, R, Ranchal, R, An, B, Othman, LB & Bhargava, B 2016, 'Consumer Oriented Privacy Preserving Access Control for Electronic Health Records in the Cloud', in *Computing (CLOUD), 2016 IEEE 9th International Conference on* IEEE, pp. 608-15.

Forouzan, BA & Mukhopadhyay, D 2011, *Cryptography And Network Security (Sie)*, McGraw-Hill Education.

Fuchsbauer, GJ 2006, 'An Introduction to Probabilistic Encryption', *Osječki matematički list*, vol. 6, no. 1, pp. 37-44.

Fujisaki, E, Okamoto, T, Pointcheval, D & Stern, J 2004, 'RSA-OAEP is secure under the RSA assumption', *Journal of Cryptology*, vol. 17, no. 2, pp. 81-104.

Gampala, V, Inuganti, S & Muppidi, S 2012, 'Data security in cloud computing with elliptic curve cryptography', *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 2, no. 3, pp. 138-41.

Geisser, S 1975, 'The predictive sample reuse method with applications', *Journal of the American Statistical Association*, vol. 70, no. 350, pp. 320-8.

Giniat, EJ 2011, 'Cloud computing: innovating the business of health care: healthcare finance executives can lead their organizations in exploring ways to use cloud computing for operational success', *Healthcare Financial Management*, vol. 65, no. 5, pp. 130-2.

Girma, A, Abayomi, K & Garuba, M 2016, 'The design data flow architecture and methodologies for a newly researched comprehensive hybrid model for the detection of DDoS attacks on cloud computing environment', in *Proc. Inf. Technol. Generat. 13th Int. Conf. Inf. Technol.* pp. 377-87.

Goldwasser, S & Micali, S 1984, 'Probabilistic encryption', *Journal of computer and system sciences*, vol. 28, no. 2, pp. 270-99.

Gonzalez, N, Miers, C, Redigolo, F, Simplicio, M, Carvalho, T, Näslund, M & Pourzandi, M 2012, 'A quantitative analysis of current security concerns and solutions for cloud computing', *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 1, no. 1, p. 11.

Gonzalez, N, Miers, C, Redigolo, F, Simplicio, M, Carvalho, T, Näslund, M & Pourzandi, M 2012, 'A quantitative analysis of current security concerns and solutions for cloud computing', *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 1, no. 1, p. 11.

González-Martínez, JA, Bote-Lorenzo, ML, Gómez-Sánchez, E & Cano-Parra, R 2015, 'Cloud computing and education: A state-of-the-art survey', *Computers & Education*, vol. 80, pp. 132-51.

Gope, P & Hwang, T 2016, 'BSN-Care: A secure IoT-based modern healthcare system using body sensor network', *IEEE Sensors Journal*, vol. 16, no. 5, pp. 1368-76.

Goyal, V, Pandey, O, Sahai, A & Waters, B 2006, 'Attribute-based encryption for fine-grained access control of encrypted data', in *Proceedings of the 13th ACM conference on Computer and communications security* Acm, pp. 89-98.

Gregg, DG, Kulkarni, UR & Vinzé, AS 2001, 'Understanding the philosophical underpinnings of software engineering research in information systems', *Information Systems Frontiers*, vol. 3, no. 2, pp. 169-83.

Grizzle, K, Wahlstroem, E, Mortimore, C & Hunt, P 2015, 'System for cross-domain identity management: Core schema', *System*.

Gu, Y, Wang, D & Liu, C 2014, 'DR-Cloud: Multi-cloud based disaster recovery service', *Tsinghua Science and Technology*, vol. 19, no. 1, pp. 13-23.

Guo, C, Zhuang, R, Jie, Y, Ren, Y, Wu, T & Choo, K-KR 2016, 'Fine-grained database field search using attribute-based encryption for e-healthcare clouds', *Journal of medical systems*, vol. 40, no. 11, p. 235.

Guo, H, Li, Z, Mu, Y & Zhang, X 2008, 'Cryptanalysis of simple three-party key exchange protocol', *Computers & Security*, vol. 27, no. 1, pp. 16-21.

Guo, H, Li, Z, Mu, Y, Zhang, F, Wu, C & Teng, J 2011, 'An efficient dynamic authenticated key exchange protocol with selectable identities', *Computers & Mathematics with Applications*, vol. 61, no. 9, pp. 2518-27.

Haghighat, M, Zonouz, S & Abdel-Mottaleb, M 2015, 'CloudID: Trustworthy cloud-based and cross-enterprise biometric identification', *Expert Systems with Applications*, vol. 42, no. 21, pp. 7905-16.

Håkansson, A 2013, 'Portal of research methods and methodologies for research projects and degree projects', in *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*: The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), p. 1.

Hameed, AA, Karlik, B & Salman, MS 2016, 'Back-propagation algorithm with variable adaptive momentum', *Knowledge-Based Systems*, vol. 114, pp. 79-87.

Haufe, K, Dzombeta, S & Brandis, K 2014, 'Proposal for a security management in cloud computing for health care', *The Scientific World Journal*, vol. 2014.

He, D, Chan, S, Zhang, Y & Yang, H 2014, 'Lightweight and confidential data discovery and dissemination for wireless body area networks', *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 2, pp. 440-8.

Hevner, AR, March, ST, Park, J & Ram, S 2004, 'Design science in information systems research', *Management Information Systems Quarterly*, vol. 28, no. 1, p. 6.

Hosseini, SM, Karimi, H & Jahan, MV 2014, 'Generating pseudo-random numbers by combining two systems with complex behaviors', *Journal of Information Security and Applications*, vol. 19, no. 2, pp. 149-62.

Hsu, I & Cheng, FQ 2015, 'SAaaS: a cloud computing service model using semantic-based agent', *Expert Systems*, vol. 32, no. 1, pp. 77-93.

Hu, J-X, Chen, C-L, Fan, C-L & Wang, K-h 2017, 'An Intelligent and Secure Health Monitoring Scheme Using IoT Sensor Based on Cloud Computing', *Journal of Sensors*, vol. 2017.

Huang, HF 2009, 'A simple three-party password-based key exchange protocol', *International Journal of communication systems*, vol. 22, no. 7, pp. 857-62.

Huang, L-C, Chu, H-C, Lien, C-Y, Hsiao, C-H & Kao, T 2009, 'Privacy preservation and information security protection for patients' portable electronic health records', *Computers in biology and medicine*, vol. 39, no. 9, pp. 743-50.

Hussain, I & Ashraf, I 2014, 'Security Issues in Cloud Computing-A Review', *International Journal of Advanced Networking and Applications*, vol. 6, no. 2, p. 2240.

Ibrahem, MK 2012, 'Modification of Diffie-Hellman key exchange algorithm for Zero knowledge proof', in *Future Communication Networks (ICFCN), 2012 International Conference on*: *proceedings of the* IEEE, pp. 147-52.

Ibraimi, L, Asim, M & Petković, M 2009, 'Secure management of personal health records by applying attribute-based encryption', in *the Wearable Micro and Nano Technologies for Personalized Health (pHealth), 2009 6th International Workshop on* IEEE, pp. 71-4.

Identity Theft Resource Centre (ITRC) 2015, https://www.idtheftcenter.org.

IEEE Std 1619 2008, 'The XTS-AES tweakable block cipher', *Institute of Electrical and Electronics Engineers, Inc.*

Islam, SH 2015, 'Design and analysis of a three party password-based authenticated key exchange protocol using extended chaotic maps', *Information Sciences*, vol. 312, pp. 104-30.

Jaber, AN & Zolkipli, MFB 2013, 'Use of cryptography in cloud computing', in *2013 IEEE International Conference on*: *proceedings of the Control System, Computing and Engineering (ICCSCE),* IEEE, pp. 179-84.

Jafari, M, Safavi-Naini, R & Sheppard, NP 2011, 'A rights management approach to protection of privacy in a cloud of electronic health records', in *Proceedings of the 11th annual ACM workshop on Digital rights management* ACM, pp. 23-30.

Jalili, R, Imani-Mehr, F, Amini, M & Shahriari, H 2005, 'Detection of distributed denial of service attacks using statistical pre-processor and unsupervised neural networks', *Information Security Practice and Experience*, pp. 192-203.

Jiang, Q, Khan, MK, Lu, X, Ma, J & He, D 2016, 'A privacy preserving three-factor authentication protocol for e-Health clouds', *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3826-49. (A)

Jiang, Q, Lian, X, Yang, C, Ma, J, Tian, Y & Yang, Y 2016, 'A bilinear pairing based anonymous authentication scheme in wireless body area networks for mHealth', *Journal of medical systems*, vol. 40, no. 11, p. 231. (B)

Jiang, Q, Ma, J, Li, G & Ma, Z 2013, 'An improved password-based remote user authentication protocol without smart cards', *Information Technology and Control*, vol. 42, no. 2, pp. 113-23.

Jing, S-Y, Ali, S, She, K & Zhong, Y 2013, 'State-of-the-art research study for green cloud computing', *The Journal of Supercomputing*, pp. 1-24.

John, GH & Langley, P 1995, 'Estimating continuous distributions in Bayesian classifiers', in *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence* Morgan Kaufmann Publishers Inc., pp. 338-45.

Kaelber, DC, Jha, AK, Johnston, D, Middleton, B & Bates, DW 2008, 'A research agenda for personal health records (PHRs)', *Journal of the American Medical Informatics Association*, vol. 15, no. 6, pp. 729-36.

Kahani, N, Elgazzar, K & Cordy, JR 2016, 'Authentication and Access Control in e-Health Systems in the Cloud', in *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on* IEEE, pp. 13-23.

Kahate, A 2013, *Cryptography and network security*, Tata McGraw-Hill Education.

Kamara, S & Lauter, K 2010, 'Cryptographic cloud storage', in *Financial Cryptography and Data Security*, Springer, Berlin Heidelberg, pp. 136-49.

Kapoor, B, Pandya, P & Sherif, JS 2011, 'Cryptography: A security pillar of privacy, integrity and authenticity of data communication', *Kybernetes*, vol. 40, no. 9/10, pp. 1422-39.

Katz, J & Vaikuntanathan, V 2013, 'Round-optimal password-based authenticated key exchange', *Journal of Cryptology*, vol. 26, no. 4, pp. 714-43.

Keromytis, AD, Misra, V & Rubenstein, D 2004, 'SOS: An architecture for mitigating DDoS attacks', *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 176-88.

Khader, AS & Lai, D 2015, 'Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol', in *Telecommunications (ICT), 2015 22nd International Conference on*: *proceedings of the* IEEE, pp. 204-8.

Khalil, I, Khreishah, A & Azeem, M 2014, 'Consolidated Identity Management System for secure mobile cloud computing', *Computer networks*, vol. 65, pp. 99-110.

Khan, AR 2012, 'Access control in cloud computing environment', *ARPN Journal of Engineering and Applied Sciences*, vol. 7, no. 5, pp. 613-5.

Khan, MK & Kumari, S 2013, 'An improved biometrics-based remote user authentication scheme with user anonymity', *BioMed research international*, vol. 2013.

Khanna, S, Venkatesh, SS, Fatemieh, O, Khan, F & Gunter, CA 2012, 'Adaptive selective verification: An efficient adaptive countermeasure to thwart dos attacks', *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 3, pp. 715-28.

Kocabas, O & Soyata, T 2015, 'Utilizing homomorphic encryption to implement secure and private medical cloud computing', in *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on* IEEE, pp. 540-7.

Kumar, CK, Jose, GJA, Sajeev, C & Suyambulingom, C 2012, 'Safety measures against man-in-the-middle attack in key exchange', *Asia Research Publishing Network (ARPN), Journal of Engineering and Applied Sciences*, vol. 7, no. 2.

Kumar, N, Gupta, P, Sahu, M & Rizvi, M 2013, 'Boolean Algebra based effective and efficient asymmetric key cryptography algorithm: BAC algorithm', in *2013 International Multi-Conference on*: *proceedings of the Automation, Computing, Communication, Control and Compressed Sensing (iMac4s),* IEEE, pp. 250-4.

Kumar, S & Gomez, O 2010, 'Denial of Service due to direct and Indirect ARP storm attacks in LAN environment', *Journal of Information Security*, vol. 1, no. 02, p. 88.

Kumarage, H, Khalil, I, Alabdulatif, A, Tari, Z & Yi, X 2016, 'Secure data analytics for cloud-integrated internet of things applications', *IEEE Cloud Computing*, vol. 3, no. 2, pp. 46-56.

Kushida, KE, Murray, J & Zysman, J 2015, 'Cloud computing: from scarcity to abundance', *Journal of Industry, Competition and Trade*, vol. 15, no. 1, pp. 5-19.

Kuzmanovic, A & Knightly, EW 2003, 'Low-rate TCP-targeted denial of service attacks: the shrew vs. the mice and elephants', in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications* ACM, pp. 75-86.

Laplante, PA, Zhang, J & Voas, J 2008, 'What's in a Name? Distinguishing between SaaS and SOA', *It Professional*, vol. 10, no. 3.

Lee, C-C, Li, C-T & Hsu, C-W 2013, 'A three-party password-based authenticated key exchange protocol with user anonymity using extended chaotic maps', *Nonlinear Dynamics*, vol. 73, no. 1-2, pp. 125-32.

Lee, K, Kim, J, Kwon, KH, Han, Y & Kim, S 2008, 'DDoS attack detection method using cluster analysis', *Expert Systems with Applications*, vol. 34, no. 3, pp. 1659-65.

Lehr, W 2015, '3 Reliability and the Internet Cloud', *Regulating the Cloud: Policy for Computing Infrastructure*, p. 87.

Lek, K & Rajapakse, N 2012, *Cryptography: Protocols, Design and Applications*, Nova Science Publishers, Inc.

Lewko, AB & Waters, B 2012, 'New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques', in *CRYPTO*: *proceedings of the CRYPTO* Springer, pp. 180-98.

Li N 2010, 'editor Research on Diffie-Hellman key exchange protocol', *Computer Engineering and Technology (ICCET), 2010 2nd International Conference of the* IEEE.

Li, C-T, Lee, C-W & Shen, J-J 2015, 'A secure three-party authenticated key exchange protocol based on extended chaotic maps in cloud storage service', in *Information Networking (ICOIN), 2015 International Conference on*: *proceedings of the* IEEE, pp. 31-6.

Li, H, Yang, Y, Luan, TH, Liang, X, Zhou, L & Shen, XS 2016, 'Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data', *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 3, pp. 312-25. (A)

Li, J, Lin, X, Zhang, Y & Han, J 2017, 'KSF-OABE: outsourced attribute-based encryption with keyword search function for cloud storage', *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 715-25.

Li, M, Yu, S, Zheng, Y, Ren, K & Lou, W 2013, 'Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption', *IEEE transactions on parallel and distributed systems*, vol. 24, no. 1, pp. 131-43.

Li, Y, Gai, K, Ming, Z, Zhao, H & Qiu, M 2016, 'Intercrossed access controls for secure financial services on multimedia big data in cloud systems', *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 12, no. 4s, p. 67. (B)

Liang, K & Susilo, W 2015, 'Searchable attribute-based mechanism with efficient data sharing for secure cloud storage', *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1981-92.

Liang, K, Au, MH, Liu, JK, Susilo, W, Wong, DS, Yang, G, Yu, Y & Yang, A 2015, 'A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing', *Future Generation Computer Systems*, vol. 52, pp. 95-108.

Lipmaa, H, Rogaway, P & Wagner, D 2000, 'CTR-mode encryption', in *First NIST Workshop on Modes of Operation*: *proceedings of NIST*.

Lipmaa, H, Rogaway, P & Wagner, D 2010, 'Comments to NIST concerning AES Modes of Operations: CTR-mode encryption, September 2000', *available at the website of http://www. cs. ucdavis. edu/rogaway/papers/ctr. pdf*.

Lippmann, R 1987, 'An introduction to computing with neural nets', *IEEE Assp magazine*, vol. 4, no. 2, pp. 4-22.

Liskov, M & Minematsu, K 2008, 'Comments on xts-aes', *Comments to NIST, available from their web page*.

Liu, C, Yang, C, Zhang, X & Chen, J 2015, 'External integrity verification for outsourced big data in cloud and IoT: A big picture', *Future Generation Computer Systems*, vol. 49, pp. 58-67. (A)

Liu, D, Dai, Y, Luan, T & Yu, S 2015, 'Personalized search over encrypted data with efficient and secure updates in mobile clouds', *IEEE Transactions on Emerging Topics in Computing*. (B)

Liu, J, Huang, X & Liu, JK 2015, 'Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption', *Future Generation Computer Systems*, vol. 52, pp. 67-76. (C)

Liu, Z, Weng, J, Li, J, Yang, J, Fu, C & Jia, C 2016, 'Cloud-based electronic health record system supporting fuzzy keyword search', *Soft Computing*, vol. 20, no. 8, pp. 3243-55.

Lu, K, Wu, D, Fan, J, Todorovic, S & Nucci, A 2007, 'Robust and efficient detection of DDoS attacks for large-scale internet', *Computer Networks*, vol. 51, no. 18, pp. 5036-56. (A)

Lu, R & Cao, Z 2007, 'Simple three-party key exchange protocol', *Computers & Security*, vol. 26, no. 1, pp. 94-7.

Lu, W-J, Yamada, Y & Sakuma, J 2015, 'Privacy-preserving genome-wide association studies on cloud environment using fully homomorphic encryption', *BMC medical informatics and decision making*, vol. 15, no. 5, p. S1.

Ma, X 2012, 'Security concerns in cloud computing', in *2012 Fourth International Conference on*: *proceedings of the Computational and Information Sciences (ICCIS),* IEEE, pp. 1069-72.

Madyastha, RK & Aazhang, B 1994, 'An algorithm for training multilayer perceptrons for data classification and function interpolation', *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 41, no. 12, pp. 866-75.

Mansoori, B, Rosipko, B, Erhard, KK & Sunshine, JL 2014, 'Design and implementation of disaster recovery and business continuity solution for radiology PACS', *Journal of digital imaging*, vol. 27, no. 1, pp. 19-25.

Maral, V, Kale, S, Balharpure, K, Bhakkad, S & Hendre, P 2016, 'Homomorphic Encryption for Secure Data Mining in Cloud', *International Journal of Engineering Science*, vol. 4533.

March, ST & Smith, GF 1995, 'Design and natural science research on information technology', *Decision support systems*, vol. 15, no. 4, pp. 251-66.

Mason, M 2018, 'Overcoming the risks of privileged user abuse in Salesforce', *Network Security*, vol. 2018, no. 8, pp. 6-8.

McGrew, D & Viega, J 2004, 'The Galois/counter mode of operation (GCM)', *Submission to NIST. http://csrc. nist. gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-spec. pdf*.

McGrew, DA & Fluhrer, SR 2004, 'The Extended Codebook (XCB) Mode of Operation', *IACR Cryptology ePrint Archive*, vol. 2004, p. 278.

McLachlan, G, Do, K-A & Ambroise, C 2005, *Analyzing microarray gene expression data*, vol. 422, John Wiley & Sons.

Meenakshi, M 2012, 'An overview on cloud computing technology', *International Journal of Advances in Computing and Information Technology*.

Mell, P & Grance, T 2009, 'The NIST definition of cloud computing. National Institute of Standards and Technology (NIST)', *Information Technology Laboratory.[Online]. Available: http://csrc. nist. gov/groups/SNS/cloud-computing/index. html*, vol. 53, p. 50.

Mell, P & Grance, T 2011, 'The NIST definition of cloud computing'. National Institute of Standards and Technology (NIST)', *Information Technology Laboratory.*

Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf.

Mihaita, A-E, Dobre, C, Pop, F, Mavromoustakis, CX & Mastorakis, G 2017, 'Secure Opportunistic Vehicle-to-Vehicle Communication', in *Advances in Mobile Cloud Computing and Big Data in the 5G Era*, Springer, pp. 229-68.

Moore, D, Shannon, C, Brown, DJ, Voelker, GM & Savage, S 2006, 'Inferring internet denial-of-service activity', *ACM Transactions on Computer Systems (TOCS)*, vol. 24, no. 2, pp. 115-39.

Moradi, M & Zulkernine, M 2004, 'A neural network based system for intrusion detection and classification of attacks', in *Proceedings of the IEEE International Conference on Advances in Intelligent Systems-Theory and Applications* pp. 15-8.

Mxoli, A, Gerber, M & Mostert-Phipps, N 2014, 'Information security risk measures for Cloud-based Personal Health Records', in *Information Society (i-Society), 2014 International Conference on* IEEE, pp. 187-93.

Nam, J, Lee, Y, Kim, S & Won, D 2007, 'Security weakness in a three-party pairing-based protocol for password authenticated key exchange', *Information Sciences*, vol. 177, no. 6, pp. 1364-75.

Nedelcu, B, Stefanet, M-E, Tamasescu, I-F, Tintoiu, S-E & Vezeanu, A 2015, 'Cloud Computing and its Challenges and Benefits in the Bank System', *Database Systems Journal*, vol. 5, no. 1, pp. 45-58.

Nezhad, SMT, Nazari, M & Gharavol, EA 2016, 'A novel DoS and DDoS attacks detection algorithm using ARIMA time series model and chaotic system in computer networks', *IEEE Communications Letters*, vol. 20, no. 4, pp. 700-3.

Nose, P 2011, 'Security weaknesses of authenticated key agreement protocols', *Information Processing Letters*, vol. 111, no. 14, pp. 687-96.

Nunamaker Jr, JF, Chen, M & Purdin, TD 1991, 'Systems development in information systems research', *Journal of management information systems*, vol. 7, no. 3, pp. 89-106.

Nuñez, D & Agudo, I 2014, 'BlindIdM: A privacy-preserving approach for identity management as a service', *International Journal of Information Security*, vol. 13, no. 2, pp. 199-215.

Paar, C & Pelzl, J 2009, *Understanding cryptography: a textbook for students and practitioners*, Springer Science & Business Media.

Page, A, Kocabas, O, Ames, S, Venkitasubramaniam, M & Soyata, T 2014, 'Cloud-based secure health monitoring: Optimising fully-homomorphic encryption for streaming algorithms', in *Globecom Workshops (GC Wkshps), 2014* IEEE, pp. 48-52.

Page, A, Kocabas, O, Soyata, T, Aktas, M & Couderc, JP 2015, 'Cloud-Based Privacy-Preserving Remote ECG Monitoring and Surveillance', *Annals of Noninvasive Electrocardiology*, vol. 20, no. 4, pp. 328-37.

Pasupuleti, SK, Ramalingam, S & Buyya, R 2016, 'An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing', *Journal of Network and Computer Applications*, vol. 64, pp. 12-22.

Peffers, K, Tuunanen, T, Rothenberger, MA & Chatterjee, S 2008, 'A design science research methodology for information systems research', *Journal of management information systems*, vol. 24, no. 3, pp. 45-77.

Pletea, D, Sedghi, S, Veeningen, M & Petkovic, M 2015, 'Secure distributed key generation in attribute based encryption systems', in *Internet Technology and Secured Transactions (ICITST), 2015 10th International Conference for*: *proceedings of* IEEE, pp. 103-7.

Purao, S 2013, 'Truth or dare: The ontology question in design science research', *Journal of Database Management (JDM)*, vol. 24, no. 3, pp. 51-66.

Purao, S. (2013). "Truth or Dare: The Ontology Question in Design Science Research." Journal of Database Management 24(3): 51-66

Qian, H, Li, J, Zhang, Y & Han, J 2015, 'Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation', *International Journal of Information Security*, vol. 14, no. 6, pp. 487-97.

Qin, B, Deng, H, Wu, Q, Domingo-Ferrer, J, Naccache, D & Zhou, Y 2015, 'Flexible attribute-based encryption applicable to secure e-healthcare records', *International Journal of Information Security*, vol. 14, no. 6, pp. 499-511.

Rahulamathavan, Y, Veluru, S, Han, J, Li, F, Rajarajan, M & Lu, R 2016, 'User Collusion Avoidance Scheme for Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption', *IEEE Transactions on computers*, vol. 65, no. 9, pp. 2939-46.

Rajamani, T, Sevugan, P & Purushotham, S 2016, 'An Investigation on the techniques used for encryption and authentication for data security in cloud computing', *IIOAB Journal*, vol. 7, no. 5, pp. 126-38.

Rannenberg, K, Camenisch, J & Sabouri, A 2015, 'Attribute-based credentials for trust', *Identity in the Information Society, Springer*.

Rao, YS 2017, 'A secure and efficient Ciphertext-Policy Attribute-Based Signcryption for Personal Health Records sharing in cloud computing', *Future Generation Computer Systems*, vol. 67, pp. 133-51.

Regola, N & Chawla, NV 2013, 'Storing and using health data in a virtual private cloud', *Journal of medical Internet research*, vol. 15, no. 3.

Ritchey, K, Paez, M, McGregor, V & Sendra, M 2013, 'Global Privacy and Data Security Developments-2013', *Bus. Law.*, vol. 69, p. 245.

Roberts, SL 2017, 'Examining Data Breaches in Healthcare', Utica College.

Rocha, F, Abreu, S & Correia, M 2011, 'The final frontier: Confidentiality and privacy in the cloud', *Computer*, vol. 44, no. 9, pp. 44-50.

Rodrigues, JJ, De La Torre, I, Fernández, G & López-Coronado, M 2013, 'Analysis of the security and privacy requirements of cloud-based electronic health records systems', *Journal of medical Internet research*, vol. 15, no. 8.

Rogaway, P 2004, 'Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC', in *Advances in Cryptology-ASIACRYPT 2004*, Springer, pp. 16-31.

Rogaway, P 2011, 'Evaluation of some blockcipher modes of operation', *Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan*.

Rosado, DG, Gómez, R, Mellado, D & Fernández-Medina, E 2012, 'Security analysis in the migration to cloud environments', *Future Internet*, vol. 4, no. 2, pp. 469-87.

Ruj, S, Stojmenovic, M & Nayak, A 2014, 'Decentralized access control with anonymous authentication of data stored in clouds', *IEEE transactions on parallel and distributed systems*, vol. 25, no. 2, pp. 384-94.

Sahai, A & Waters, B 2005, 'Fuzzy identity-based encryption', in *Eurocrypt*: Springer, pp. 457-73.

Sahi, A, Lai, D & Li, Y 2015, 'Parallel Encryption Mode for Probabilistic Scheme to Secure Data in the Cloud', in *10th International Conference on Information Technology and Applications (ICITA)* Sydney.

Sahi, A, Lai, D & Li, Y 2016, 'Security and privacy preserving approaches in eHealth clouds with disaster recovery plan', *Computers in biology and medicine*, vol. 78, pp. 1-8.

Sahi, A, Lai, D & Li, Y 2017, 'An Energy Efficient TCP DoS Attacks Mitigation Method in Cloud Computing', in *The First MoHESR and HCED Iraqi Scholars Conference in Australasia 2017*, Swinburne University of Technology, pp. 289-94. (B)

Sahi, A, Lai, D, Li, Y & Diykh, M 2017, 'An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment', *IEEE Access*, vol. 5, pp. 6036-48. (A)

Sajid, A & Abbas, H 2016, 'Data privacy in cloud-assisted healthcare systems: state of the art and future challenges', *Journal of medical systems*, vol. 40, no. 6, p. 155.

Salah, K, Elbadawi, K & Boutaba, R 2012, 'Performance modeling and analysis of network firewalls', *IEEE Transactions on network and service management*, vol. 9, no. 1, pp. 12-21.

Sánchez, R, Almenares, F, Arias, P, Díaz-Sánchez, D & Marín, A 2012, 'Enhancing privacy and dynamic federation in IdM for consumer cloud computing', *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1.

Sen, J 2013, 'Security and privacy issues in cloud computing', *Architectures and Protocols for Secure Information Technology Infrastructures*, pp. 1-45.

Shameli-Sendi, A, Pourzandi, M, Fekih-Ahmed, M & Cheriet, M 2015, 'Taxonomy of distributed denial of service mitigation approaches for cloud computing', *Journal of Network and Computer Applications*, vol. 58, pp. 165-79.

Sharma, G & Kalra, S 2016, 'Identity based secure authentication scheme based on quantum key distribution for cloud computing', *Peer-to-Peer Networking and Applications*, pp. 1-15.

Shehata, K, Hussien, H & Hamdy, N 2003, 'Design and implementation of a universal communication security unit on an FPGA', in *Circuits and Systems, 2003 IEEE 46th Midwest Symposium on*: *proceedings of the* IEEE, pp. 855-9.

Snedaker, S 2013, *Business continuity and disaster recovery planning for IT professionals*, Newnes.

Somani, G, Gaur, MS, Sanghi, D & Conti, M 2016, 'DDoS attacks in cloud computing: collateral damage to non-targets', *Computer Networks*, vol. 109, pp. 157-71.

Son, J, Kim, J-D, Na, H-S & Baik, D-K 2016, 'Dynamic access control model for privacy preserving personalized healthcare in cloud environment', *Technology and Health Care*, vol. 24, no. s1, pp. S123-S9.

Song, W, Wang, B, Wang, Q, Peng, Z, Lou, W & Cui, Y 2017, 'A privacy-preserved full-text retrieval algorithm over encrypted data for cloud storage applications', *Journal of Parallel and Distributed Computing*, vol. 99, pp. 14-27.

Sosinsky, B 2010, *Cloud computing bible*, vol. 762, John Wiley & Sons.

Stallings, W 2010, 'NIST block cipher modes of operation for confidentiality', *Cryptologia*, vol. 34, no. 2, pp. 163-75.

Stamp, M 2011, *Information security: principles and practice*, John Wiley & Sons.

Stone, M 1974, 'Cross-validatory choice and assessment of statistical predictions', *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 111-47.

Su, J-S, Cao, D, Wang, X-F, Sun, Y-P & Hu, Q-L 2011, 'Attribute based encryption schemes', *Journal of Software*, vol. 22, no. 6, pp. 1299-315.

Subashini, S & Kavitha, V 2011, 'A survey on security issues in service delivery models of cloud computing', *Journal of network and computer applications*, vol. 34, no. 1, pp. 1-11.

Sugumaran, M, Murugan, BB & Kamalraj, D 2014, 'An architecture for data security in cloud computing', in *2014 World Congress on*: *proceedings of the Computing and Communication Technologies (WCCCT),* IEEE, pp. 252-5.

Suykens, JA & Vandewalle, J 1999, 'Least squares support vector machine classifiers', *Neural processing letters*, vol. 9, no. 3, pp. 293-300.

System for cross-domain identity management, 2017, <http://www.simplecloud.info/>.

Talbot, D 2010, 'Security in the Ether', *Technology Review*, vol. 113, no. 1, pp. 36-42.

Tang, H, Liu, X & Jiang, L 2013, 'A Robust and Efficient Timestamp-based Remote User Authentication Scheme with Smart Card Lost Attack Resistance', *IJ Network Security*, vol. 15, no. 6, pp. 446-54.

Tang, J, Cui, Y, Li, Q, Ren, K, Liu, J & Buyya, R 2016, 'Ensuring security and privacy preservation for cloud data services', *ACM Computing Surveys (CSUR)*, vol. 49, no. 1, p. 13.

Taravat, A, Del Frate, F, Cornaro, C & Vergari, S 2015, 'Neural networks and support vector machine algorithms for automatic cloud classification of whole-sky ground-

based images', *IEEE Geoscience and remote sensing letters*, vol. 12, no. 3, pp. 666-70.

Tarhuni, MA, Ng, SH, Samsudin, A & Ng, WP 2003, 'Enhanced counter mode', in *Communications, 2003. APCC 2003. The 9th Asia-Pacific Conference on*: *proceedings of the* IEEE, pp. 701-5.

Thilakanathan, D, Chen, S, Nepal, S, Calvo, R & Alem, L 2014, 'A platform for secure monitoring and sharing of generic health data in the Cloud', *Future Generation Computer Systems*, vol. 35, pp. 102-13. (A)

Thilakanathan, D, Zhao, Y, Chen, S, Nepal, S, Calvo, RA & Pardo, A 2014, 'Protecting and analysing health care data on cloud', in *Advanced Cloud and Big Data (CBD), 2014 Second International Conference on*: *proceedings of the* IEEE, pp. 143-9. (B)

Toma, C 2009, 'Security Issues of the Digital Certificates within Public Key Infrastructures', *Informatica Economica*, vol. 13, no. 1, p. 16.

Tong, Y, Sun, J, Chow, SS & Li, P 2014, 'Cloud-assisted mobile-access of health data with privacy and auditability', *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 2, pp. 419-29.

Tran, DH, Nguyen, H-L, Zha, W & Ng, WK 2011, 'Towards security in sharing data on cloud-based social networks', in *proceedings of the Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on* IEEE, pp. 1-5.

Vaishnavi, V & Kuechler, W 2004, 'Design research in information systems'.

Van Gorp, P, Comuzzi, M, Jahnen, A, Kaymak, U & Middleton, B 2014, 'An open platform for personal health record apps with platform-level privacy protection', *Computers in biology and medicine*, vol. 51, pp. 14-23.

Van Tilborg, HC & Jajodia, S 2014, *Encyclopedia of cryptography and security*, Springer Science & Business Media.

Vuyyuru, M, Annapurna, P, Babu, KG & Ratnam, A 2012, 'An overview of cloud computing technology', *International Journal of Soft Computing and Engineering*, vol. 2, no. 3, p. 244.

Wang, B, Zheng, Y, Lou, W & Hou, YT 2015, 'DDoS attack protection in the era of cloud computing and software-defined networking', *Computer Networks*, vol. 81, pp. 308-19. (D)

Wang, C, Liu, X & Li, W 2012, 'Implementing a personal health record cloud platform using ciphertext-policy attribute-based encryption', in *proceedings of the Intelligent*

Networking and Collaborative Systems (INCoS), 2012 4th International Conference on IEEE, pp. 8-14.

Wang, C, Xu, X, Shi, D & Fang, J 2015, 'Privacy-preserving Cloud-based Personal Health Record System Using Attribute-based Encryption and Anonymous Multi-ReceiverIdentity-based Encryption', *Informatica*, vol. 39, no. 4. (A)

Wang, P, Lin, H-T & Wang, T-S 2016, 'An improved ant colony system algorithm for solving the IP traceback problem', *Information Sciences*, vol. 326, pp. 172-87. (A)

Wang, S, Zhou, J, Liu, JK, Yu, J, Chen, J & Xie, W 2016, 'An efficient file hierarchy attribute-based encryption scheme in cloud computing', *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1265-77. (B)

Wang, W, Chen, L & Zhang, Q 2015, 'Outsourcing high-dimensional healthcare data to cloud with personalized privacy preservation', *Computer networks*, vol. 88, pp. 136-48. (B)

Wang, X & Reiter, MK 2010, 'Using web-referral architectures to mitigate denial-of-service threats', *IEEE Transactions on dependable and secure computing*, vol. 7, no. 2, pp. 203-16.

Wang, XA, Ma, J, Xhafa, F, Zhang, M & Luo, X 2017, 'Cost-effective secure E-health cloud system using identity based cryptographic techniques', *Future Generation Computer Systems*, vol. 67, pp. 242-54.

Wang, Z, Graham, J, Ajam, N & Jiang, H 2011, 'Design and optimization of hybrid MD5-blowfish encryption on GPUs', in *Proceedings of 2011 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), Las Vegas, Nevada, USA*.

Wang, Z, Huang, D, Zhu, Y, Li, B & Chung, C-J 2015, 'Efficient attribute-based comparable data access control', *IEEE Transactions on computers*, vol. 64, no. 12, pp. 3430-43. (C)

Wei, S 2006, 'On generalization of Geffe's generator', *IJCSNS International Journal of Computer Science and Network Security*, vol. 6, no. 8A, pp. 161-5.

Wei, W, Chen, F, Xia, Y & Jin, G 2013, 'A rank correlation based detection against distributed reflection DoS attacks', *IEEE Communications Letters*, vol. 17, no. 1, pp. 173-5.

Wen, H-A, Lee, T-F & Hwang, T 2005, 'Provably secure three-party password-based authenticated key exchange protocol using Weil pairing', *IEE Proceedings-Communications*, vol. 152, no. 2, pp. 138-43.

Wen, W, Wang, L & Pan, J 2016, 'Unified security model of authenticated key exchange with specific adversarial capabilities', *IET Information Security*, vol. 10, no. 1, pp. 8-17.

Williamson, K & Johanson, G 2017, 'Research Methods: Information, Systems, and Contexts', United States, Chandos Publishing.

Wireshark Network Analyser 2017, https://www.wireshark.org/

Wood, T, Cecchet, E, Ramakrishnan, KK, Shenoy, PJ, van der Merwe, JE & Venkataramani, A 2010, 'Disaster Recovery as a Cloud Service: Economic Benefits & Deployment Challenges', *HotCloud*, vol. 10, pp. 8-15.

Wu, F, Xu, L, Kumari, S & Li, X 2015, 'A novel and provably secure biometrics-based three-factor remote authentication scheme for mobile client–server networks', *Computers & Electrical Engineering*, vol. 45, pp. 274-85.

Wu, L, Zhang, Y, Li, L & Shen, J 2016, 'Efficient and anonymous authentication scheme for wireless body area networks', *Journal of medical systems*, vol. 40, no. 6, p. 134. (A)

Wu, R 2012, *Secure sharing of electronic medical records in cloud computing*, Arizona State University.

Wu, S, Pu, Q, Wang, S & He, D 2012, 'Cryptanalysis of a communication-efficient three-party password authenticated key exchange protocol', *Information Sciences*, vol. 215, pp. 83-96.

Wu, Y, Lu, X, Su, J & Chen, P 2016, 'An efficient searchable encryption against keyword guessing attacks for sharable electronic medical records in cloud-based system', *Journal of medical systems*, vol. 40, no. 12, p. 258. (B)

Xhafa, F, Feng, J, Zhang, Y, Chen, X & Li, J 2015, 'Privacy-aware attribute-based PHR sharing with user accountability in cloud computing', *The Journal of Supercomputing*, vol. 71, no. 5, pp. 1607-19. (A)

Xhafa, F, Li, J, Zhao, G, Li, J, Chen, X & Wong, DS 2015, 'Designing cloud-based electronic health record system with attribute-based encryption', *Multimedia Tools and Applications*, vol. 74, no. 10, pp. 3441-58. (B)

Xia, M, Lu, W, Yang, J, Ma, Y, Yao, W & Zheng, Z 2015, 'A hybrid method based on extreme learning machine and k-nearest neighbor for cloud classification of ground-based visible cloud image', *Neurocomputing*, vol. 160, pp. 238-49.

Xia, Z, Wang, X, Sun, X & Wang, Q 2016, 'A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data', *IEEE transactions on parallel and distributed systems*, vol. 27, no. 2, pp. 340-52.

Xie, Q, Wong, DS, Wang, G, Tan, X, Chen, K & Fang, L 2017, 'Provably secure dynamic id-based anonymous two-factor authenticated key exchange protocol with extended security model', *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1382-92.

Xiong, J, Yao, Z, Ma, J, Liu, X, Li, Q & Ma, J 2014, 'PRIAM: Privacy Preserving Identity and Access Management Scheme in Cloud', *KSII Transactions on Internet & Information Systems*, vol. 8, no. 1, p. 23.

Xu, J, Wen, Q, Li, W & Jin, Z 2016, 'Circuit ciphertext-policy attribute-based hybrid encryption with verifiable delegation in cloud computing', *IEEE transactions on parallel and distributed systems*, vol. 27, no. 1, pp. 119-29.

Xuan, Y, Shin, I, Thai, MT & Znati, T 2010, 'Detecting application denial-of-service attacks: A group-testing-based approach', *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, no. 8, pp. 1203-16.

Yan, Q, Yu, FR, Gong, Q & Li, J 2016, 'Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges', *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 602-22.

Yang, J-J, Li, J-Q & Niu, Y 2015, 'A hybrid solution for privacy preserving medical data sharing in the cloud environment', *Future Generation Computer Systems*, vol. 43, pp. 74-86.

Yang, K & Jia, X 2014, 'Expressive, efficient, and revocable data access control for multi-authority cloud storage', *IEEE transactions on parallel and distributed systems*, vol. 25, no. 7, pp. 1735-44.

Yang, Y & Ma, M 2016, 'Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds', *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 746-59.

Yang, Y 2015, 'Attribute-based data retrieval with semantic keyword search for e-health cloud', *Journal of Cloud Computing*, vol. 4, no. 1, p. 10.

Yang, Y, Deng, RH & Bao, F 2006, 'A practical password-based two-server authentication and key exchange system', *IEEE Transactions on dependable and secure computing*, vol. 3, no. 2, pp. 105-14.

Yang, Y, Zhu, H, Lu, H, Weng, J, Zhang, Y & Choo, K-KR 2016, 'Cloud based data sharing with fine-grained proxy re-encryption', *Pervasive and Mobile Computing*, vol. 28, pp. 122-34.

Yantao, Z & Jianfeng, M 2010, 'A highly secure identity-based authenticated key-exchange protocol for satellite communication', *Journal of Communications and Networks*, vol. 12, no. 6, pp. 592-9.

Yao, AC-C & Zhao, Y 2014, 'Privacy-preserving authenticated key-exchange over Internet', *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 1, pp. 125-40.

Yeh, H-L, Chen, T-H, Hu, K-J & Shih, W-K 2013, 'Robust elliptic curve cryptography-based three factor user authentication providing privacy of biometric data', *IET Information Security*, vol. 7, no. 3, pp. 247-52.

Yeh, Y-S, Huang, T-Y & Lin, H-Y 2009, 'Structural Binary CBC Encryption Mode', *Journal of information science and engineering*, vol. 25, no. 3, pp. 937-44.

Yi, X, Ling, S & Wang, H 2013, 'Efficient two-server password-only authenticated key exchange', *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 9, pp. 1773-82.

Yoon, EJ & Yoo, KY 2011, 'Cryptanalysis of a simple three-party password-based key exchange protocol', *International Journal of communication systems*, vol. 24, no. 4, pp. 532-42.

Younis, YA, Kifayat, K & Merabti, M 2014, 'An access control model for cloud computing', *Journal of Information Security and Applications*, vol. 19, no. 1, pp. 45-60.

Yu, J, Lee, H, Kim, M-S & Park, D 2008, 'Traffic flooding attack detection with SNMP MIB using SVM', *Computer Communications*, vol. 31, no. 17, pp. 4212-9.

Yu, S, Wang, C, Ren, K & Lou, W 2010, 'Achieving secure, scalable, and fine-grained data access control in cloud computing', in *Infocom, 2010 proceedings IEEE*, pp. 1-9.

Yu, Y, Miyaji, A, Au, MH & Susilo, W 2017, *Cloud computing security and privacy: Standards and regulations*, Elsevier, 0920-5489.

Yüksel, B, Küpçü, A & Özkasap, Ö 2017, 'Research issues for privacy and security of electronic health services', *Future Generation Computer Systems*, vol. 68, pp. 1-13.

Zapata, BC, Niñirola, AH, Idri, A, Fernández-Alemán, JL & Toval, A 2014, 'Mobile PHRs compliance with Android and iOS usability guidelines', *Journal of medical systems*, vol. 38, no. 8, p. 81.

Zargar, ST, Joshi, J & Tipper, D 2013, 'A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks', *IEEE communications surveys & tutorials*, vol. 15, no. 4, pp. 2046-69.

Zhang, R, Liu, L & Xue, R 2014, 'Role-based and time-bound access and management of EHR data', *Security and Communication Networks*, vol. 7, no. 6, pp. 994-1015.

Zhao, F, Li, C & Liu, CF 2014, 'A cloud computing security solution based on fully homomorphic encryption', in *Advanced Communication Technology (ICACT), 2014 16th International Conference on* IEEE, pp. 485-8.

Zhen, Y 2011, 'Privacy-preserving personal health record system using attribute-based encryption', Worcester Polytechnic Institute.

Zissis, D & Lekkas, D 2012, 'Addressing cloud computing security issues', *Future Generation Computer Systems*, vol. 28, no. 3, pp. 583-92.

Zukarnain, ZA 2014, 'Increase Throughput of CCM Security Mode Using MKP', *Applied Mathematics*, vol. 5, pp. 581-92.

APPENDICES

Appendix A

The practical aspect of Chapter 4 were implemented as two main web pages to simulate this project. Each of the simulation pages has a C# code, HTML code, and a screen shot. This was implemented using Visual Studio 2010.

A.1 First web page codes

A.1.1 C# code:

```
using System;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;
using System.Numerics;


public partial class _Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

        TextBox1.Focus();
        TextBox1.MaxLength = 8;
        TextBox2.Text = "";
        TextBox3.Text = "";
        TextBox4.Text = "";
        TextBox5.Text = "";
        TextBox6.Text = "";
        TextBox9.Text = "";
        TextBox10.Text = "";
        TextBox13.Text = "";
        TextBox14.Text = "";
        TextBox17.Text = "";
        TextBox18.Text = "";
        TextBox19.Text = "";
        TextBox26.Text = "";

    }


    protected void Button1_Click1(object sender, EventArgs e)
    {
        Image2.Visible = true;

        int s = 256;
```

```
 string result = string.Empty;
 foreach (char ch in TextBox1.Text)
 {
     if (ch > 255)
     {
         Server.Transfer("Error page.aspx");
     }
     else
     {
       TextBox3.Text += Convert.ToString((int)ch,
       2).PadLeft(8, '0');
         TextBox2.Text += Convert.ToByte(ch);
         TextBox2.Text += " ";
     }
 }

 ///TextBox4.Text =
TextBox2.Text.Substring(startIndex, length);
 TextBox4.Text = TextBox3.Text.Substring(0, 20);
 TextBox5.Text = TextBox3.Text.Substring(20, 21);
 TextBox6.Text = TextBox3.Text.Substring(41, 23);


 ///LFSR 1... start here
 string lfsr1 = TextBox4.Text;

 for (int i = 0; i < s; i++)
 {
     string s10 = lfsr1.Substring(9, 1);
     string s_last = lfsr1.Substring(0, 1);

     if (s10 == "0")
         if (s_last == "0")
         {
             TextBox8.Text = "0";
         }

         else
         {
             TextBox8.Text = "1";
         }
     if (s10 == "1")
         if (s_last == "1")
         {
             TextBox8.Text = "0";
         }
         else
         {
             TextBox8.Text = "1";
         }

     TextBox7.Text = "";
     TextBox7.Text = TextBox8.Text;
     TextBox7.Text += lfsr1.Substring(0, 19);
     TextBox9.Text += lfsr1.Substring(19, 1);
     lfsr1 = TextBox7.Text;

 }
 ///LFSR 1... finished here


 ///LFSR 2... start here
 string lfsr2 = TextBox5.Text;

 for (int j = 0; j < s; j++)
 {
     string s10_2 = lfsr2.Substring(10, 1);
     string s_last_2 = lfsr2.Substring(0, 1);
```

```
if (s10_2 == "0")
    if (s_last_2 == "0")
    {
        TextBox12.Text = "0";
    }
    else
    {
        TextBox12.Text = "1";
    }
if (s10_2 == "1")
    if (s_last_2 == "1")
    {
        TextBox12.Text = "0";
    }
    else
    {
        TextBox12.Text = "1";
    }

TextBox11.Text = "";
TextBox11.Text = TextBox12.Text;
TextBox11.Text += lfsr2.Substring(0, 20);
TextBox10.Text += lfsr2.Substring(20, 1);
lfsr2 = TextBox11.Text;


string not_str = TextBox10.Text.Substring(j, 1);
if (not_str == "0")
{
    TextBox13.Text += "1";
}
else
{
    TextBox13.Text += "0";
}
```

```
}


///LFSR 2... finished here



///LFSR 3... start here
string lfsr3 = TextBox6.Text;

for (int k = 0; k < s; k++)
{
    string s10_3 = lfsr3.Substring(12, 1);
    string s_last_3 = lfsr3.Substring(0, 1);

    if (s10_3 == "0")
        if (s_last_3 == "0")
        {
            TextBox16.Text = "0";
        }
        else
        {
            TextBox16.Text = "1";
        }
    if (s10_3 == "1")
        if (s_last_3 == "1")
        {
            TextBox16.Text = "0";
        }
        else
        {
            TextBox16.Text = "1";
        }
```

```
    TextBox15.Text = "";
    TextBox15.Text = TextBox16.Text;
    TextBox15.Text += lfsr3.Substring(0, 22);
    TextBox14.Text += lfsr3.Substring(22, 1);
    lfsr3 = TextBox15.Text;

}
///LFSR 3... finished here


///K1 AND K2... start here

for (int l = 0; l < s; l++)
{
    string k1 = TextBox9.Text.Substring(l, 1);
    string k2 = TextBox10.Text.Substring(l, 1);

    if (k1 == "1")
        if (k2 == "1")
        {
            TextBox17.Text += "1";
        }
        else
        {
            TextBox17.Text += "0";
        }
    if (k1 == "0")
    {
        TextBox17.Text += "0";
    }

}

///K1 AND K2... finished here
///
```

```
///Not K2 AND K3... start here


for (int z = 0; z < s; z++)
{
    string nk2 = TextBox13.Text.Substring(z, 1);
    string k3 = TextBox14.Text.Substring(z, 1);

    if (nk2 == "1")
        if (k3 == "1")
        {
            TextBox18.Text += "1";
        }
        else
        {
            TextBox18.Text += "0";
        }
    if (nk2 == "0")
    {
        TextBox18.Text += "0";
    }

}
///Not K2 AND K3... finished here



///S... start here

for (int x = 0; x < s; x++)
{
    string and1 = TextBox17.Text.Substring(x, 1);
    string and2 = TextBox18.Text.Substring(x, 1);

    if (and1 == "0")
```

```
        if (and2 == "0")
        {
            TextBox19.Text += "0";
        }
        else
        {
            TextBox19.Text += "1";
        }
    if (and1 == "1")
        if (and2 == "1")
        {
            TextBox19.Text += "0";
        }
        else
        {
            TextBox19.Text += "1";
        }
}


///S... finished here


///Statistics results... start here
double i0 = 0, i1 = 0, i00 = 0, i11 = 0, i01 = 0,
i10 = 0, n = TextBox19.Text.Length;
for (int y = 0; y < TextBox19.Text.Length; y++)
{
    string s0 = TextBox19.Text.Substring(y, 1);
    if (s0 == "0")
    {
        i0 = i0 + 1; ;
    }
    string s1 = TextBox19.Text.Substring(y, 1);
    if (s1 == "1")
    {
        i1 = i1 + 1; ;
    }
}
for (int yy = 0; yy < TextBox19.Text.Length - 1; yy++)
{
    string s00 = TextBox19.Text.Substring(yy, 2);
    if (s00 == "00")
    {
        i00 = i00 + 1; ;
    }
    string s11 = TextBox19.Text.Substring(yy, 2);
    if (s11 == "11")
    {
        i11 = i11 + 1; ;
    }
    string s01 = TextBox19.Text.Substring(yy, 2);
    if (s01 == "01")
    {
        i01 = i01 + 1; ;
    }
    string s10 = TextBox19.Text.Substring(yy, 2);
    if (s10 == "10")
    {
        i10 = i10 + 1; ;
    }

}
    TextBox20.Text = "0=" + i0 + ", " + "1=" + i1 +
  ", " + "00=" + i00 + ", " + "11=" + i11 + ", " +
  "01=" + i01 + ", " + "10=" + i10 + ", " + "n=" +
  n;

///Statistics results... finished here
```

```
///Frequency test
double x1 = Math.Pow((i0 - i1), 2) / n;
TextBox22.Text = Convert.ToString(x1);
if (x1 < 3.8415)
{
    Label22.Text = "Frequency test succeeded";
}
else
{
    Label22.Text = "Frequency test failed";
}

///Serial test
 double x2 = (4 / (n - 1)) * (Math.Pow(i00, 2) +
 Math.Pow(i01, 2) + Math.Pow(i10, 2) + Math.Pow(i11,
 2)) - (2 / n) * (Math.Pow(i0, 2) + Math.Pow(i1, 2))
 + 1;
TextBox23.Text = Convert.ToString(x2);
if (x1 < 5.9915)
{
    Label23.Text = "Serial test succeeded";
}
else
{
    Label23.Text = "Serial test failed";
}

/// Poker test



 double m = 3, kk = 85, c = 0, c0 = 0, c1 = 0, c2 =
 0, c3 = 0, c4 = 0, c5 = 0, c6 = 0, c7 = 0;
for (int ff = 0; ff < kk; ff++)
{
    string blocks = TextBox19.Text.Substring(ff * 3, 3);


    if (blocks == "000")
    {
        c0 = c0 + 1;
    }
    if (blocks == "001")
    {
        c1 = c1 + 1;
    }
    if (blocks == "010")
    {
        c2 = c2 + 1;
    }
    if (blocks == "011")
    {
        c3 = c3 + 1;
    }
    if (blocks == "100")
    {
        c4 = c4 + 1;
    }
    if (blocks == "101")
    {
        c5 = c5 + 1;
    }
    if (blocks == "110")
    {
     c6 = c6 + 1;
    }
    if (blocks == "111")
    {
```

```
    c7 = c7 + 1;
}

/* for (int f = 0; f < m; f++)
 {
     string f1 = blocks.Substring(f, 1);
     if (f1 == "1")
     {
         c++;
     }
 }
 if (c == 0)
     c0 = c0 + 1;
 if (c == 1)
     c1 = c1 + 1;
 if (c == 2)
     c2 = c2 + 1;
 if (c == 3)
     c3 = c3 + 1;
 if (c == 4)
     c4 = c4 + 1;
 if (c == 5)
     c5 = c5 + 1;
 if (c == 6)
     c6 = c6 + 1;
 if (c == 7)
     c7 = c7 + 1;
 if (c == 8)
     c8 = c8 + 1;
 if (c == 9)
     c9 = c9 + 1;
 if (c == 10)
     c10 = c10 + 1;

 c = 0;
```

```
     */
}
// TextBox6.Text = Convert.ToString(c0) + ", "+
Convert.ToString(c1) + ", "+Convert.ToString(c2) +
", "+Convert.ToString(c3) + ",
"+Convert.ToString(c4) + ", "+Convert.ToString(c5);
double x3 = Math.Pow(2, m) / kk * (Math.Pow(c0, 2) +
Math.Pow(c1, 2) + Math.Pow(c2, 2) + Math.Pow(c3, 2)
+ Math.Pow(c4, 2) + Math.Pow(c5, 2) + Math.Pow(c6,
2) + Math.Pow(c7, 2)) - kk;
        TextBox24.Text = Convert.ToString(x3);

if (x3 < 14.0671)
{
    Label24.Text = "Poker test succeeded";
}
else
{
    Label24.Text = "Poker test failed";
}
/*
/// Autocorrelation test
double ad = 0;
string si, sid;
int succeeded = 0, failed = 0;
for (int d = 1; d < n/2; d++)
{
    si = TextBox19.Text;
   string pading = TextBox19.Text.PadRight(s + d, '0');

    sid = pading.Substring(d, s);
    //TextBox7.Text = sid;

    for (int fff = 0; fff < n-d-1; fff++)
    {
```

```
            string fff1 = si.Substring(fff, 1);
            string fff2 = sid.Substring(fff, 1);
            if (fff1 == "1" & fff2 == "1")
            {
                c++;
            }
        }
        ad = c;
        c = 0;
        //TextBox6.Text += Convert.ToString(ad)+",";
         double mm = Math.Pow(i1, 2) * (n - d) /
         Math.Pow(n, 2);
        double ad_mm = ad - m;
        //double x4 = Math.Pow(ad_mm, 2) / mm;
       double x4 = 2 * (ad - (n - d) / 2) / Math.Sqrt(n - d);
        TextBox25.Text += x4 + "," + mm + "@@@@";
        if(1.96>x4)
        if (x4 > -1.96)
       // if (x4 < 3.841)
        {
            succeeded++;
            Label25.Text = "Autocorrelation test succeeded";
        }
        else
        {
            failed++;
            Label25.Text = "Autocorrelation test failed";
            d = s;
        }

        //Label25.Text = succeeded+","+failed+"Failed
        percentage:" + (failed * (100 / s)) + "%";
        int Percentage = 0;
        if (x1 < 3.8415){Percentage=Percentage+25;}
            if (x1 < 3.8415){Percentage=Percentage+25;}
                        if (x3 <
       14.0671){Percentage=Percentage+25;}
                       if(1.96 > x4)
                                   if (x4 > -1.96) {
       Percentage = Percentage + 25; }
                       Label26.Text = "This password is " +
       Percentage + "% successful";

         }*/


         /// Convert Binary to decimal

         for (int dec = 0; dec < s / 8; dec++)
         {
             string dec1 = TextBox19.Text.Substring(dec * 8,
       8);

             int dec11 = Convert.ToInt32(dec1, 2);
             dec11 = dec11 % 10;
             TextBox26.Text += Convert.ToString(dec11);
         }

         //private key
         /*
         string half1 = TextBox26.Text.Substring(0, 16);
         long p1 = Convert.ToInt64(half1);
         p1 = p1 % 100000000;
         string p11 = Convert.ToString(p1);
         string half2 = TextBox26.Text.Substring(16, 16);
         long p2 = Convert.ToInt64(half2);
         p2 = p2 % 100000000;
         string p22 = Convert.ToString(p2);
         string p3 = p11 + p22;
         long p33 = Convert.ToInt64(p3);
         int privateKey =Convert.ToInt32(p33 % 100000000);
```

```
        TextBox27.Text = Convert.ToString(privateKey);
         */
        TextBox27.Text = "";
        for (int p1 = 0; p1 < 8; p1++)
        {
           string block = TextBox26.Text.Substring(p1 * 4, 1);
            // long p2 = Convert.ToInt64(block);
            TextBox27.Text += block;
        }

        int privateKey = Convert.ToInt32(TextBox27.Text);
        ///------------------------------------------------
----
        /*
         int publickey = 1;
        int mod1 = 1;
        for (int i = 1; i <= privateKey; i=i+i)
        {
            publickey = publickey * 5;
            publickey = publickey % 100000007;
            if (publickey != 0)
            {
                mod1 = mod1 * publickey;
            }
        }

       //int publickey =Convert.ToInt32(BigInteger.Pow(5,
       privateKey) % 100000007);
         */
         BigInteger t1 = 5;
         BigInteger t2 = Convert.ToInt64(privateKey);
         BigInteger t3 = 100000007;
        BigInteger publickey = BigInteger.ModPow(t1, t2, t3);
         TextBox28.Text = Convert.ToString(publickey);
            Image2.Visible = false;



    }


    protected void Button2_Click1(object sender, EventArgs e)
    {
        Session["field1"] = TextBox27.Text;
        Session["field2"] = TextBox28.Text;
        Response.Redirect("Default2.aspx");
    }

}
```

A.1.2 HTML code:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default.aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <style type="text/css">
        .style1
        {
            width: 100%;
        }
        .style3
```

```
        {
            line-height: 115%;
        }
    </style>
</head>
<body style="font-family: 'Times New Roman', Times, serif;
font-size: small">
    <form id="form1" runat="server">
    <div>

    </div>
    <table class="style1">
        <tr>
            <td>
                Password:</td>
            <td>
                <asp:TextBox ID="TextBox1" runat="server"
Width="500px">abcdefgh</asp:TextBox>
                <asp:Button ID="Button1" runat="server"
onclick="Button1_Click1"
                    Text="Calculate" />
                <asp:Button ID="Button2" runat="server"
Text="Go to the second party side"
                    onclick="Button2_Click1" />
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                Password in decimal:</td>
            <td>
                <asp:TextBox ID="TextBox2" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                Password in binary:</td>
            <td>
                <asp:TextBox ID="TextBox3" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                 </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                LFSR 1:</td>
            <td>
                <asp:TextBox ID="TextBox4" runat="server"
Width="500px"></asp:TextBox>
             20bits<br />
                <asp:TextBox ID="TextBox7" runat="server"
Width="500px"></asp:TextBox>
 Final state<br />
                <asp:TextBox ID="TextBox8" runat="server"
Width="500px"></asp:TextBox>
                <br />
```

```
                <asp:TextBox ID="TextBox9" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
             K1</td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                 </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                LFSR 2:</td>
            <td>
                <asp:TextBox ID="TextBox5" runat="server"
Width="500px"></asp:TextBox>
                 21bits<br />
                <asp:TextBox ID="TextBox10" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
             K2<br />
                <asp:TextBox ID="TextBox11" runat="server"
Width="500px"></asp:TextBox>
             Final state<br />
                <asp:TextBox ID="TextBox12" runat="server"
Width="500px"></asp:TextBox>
                <br />
                <asp:TextBox ID="TextBox13" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
             Not K2</td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                 </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                LFSR 3:</td>
            <td>
                <asp:TextBox ID="TextBox6" runat="server"
Width="500px"></asp:TextBox>
             23bits<br />
                <asp:TextBox ID="TextBox14" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
             K3<br />
                <asp:TextBox ID="TextBox15" runat="server"
Width="500px"></asp:TextBox>
             Final state<br />
                <asp:TextBox ID="TextBox16" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
```

```
            <td>
                 </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                K1 ^ K2:</td>
            <td>
                <asp:TextBox ID="TextBox17" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                Not K2 ^ K3:</td>
            <td>
                <asp:TextBox ID="TextBox18" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                S:</td>
            <td>
                <asp:TextBox ID="TextBox19" runat="server"
Width="500px" Height="100px"
                    TextMode="MultiLine"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                Statistics results:</td>
            <td>
                <asp:TextBox ID="TextBox20" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                S:</td>
            <td>
                <asp:TextBox ID="TextBox21" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td class="style3"
                style="mso-ascii-theme-font: major-bidi;
mso-fareast-font-family: Calibri; mso-fareast-theme-font:
minor-latin; mso-hansi-theme-font: major-bidi; mso-bidi-
theme-font: major-bidi; mso-ansi-language: EN-AU; mso-
fareast-language: EN-US; mso-bidi-language: AR-SA">
                Frequency test:</td>
            <td>
                <asp:TextBox ID="TextBox22" runat="server"
Width="500px"></asp:TextBox>
            </td>
```

```
            <td>
                 </td>
        </tr>
        <tr>
            <td class="style3"
                style="mso-ascii-theme-font: major-bidi;
mso-fareast-font-family: Calibri; mso-fareast-theme-font:
minor-latin; mso-hansi-theme-font: major-bidi; mso-bidi-
theme-font: major-bidi; mso-ansi-language: EN-AU; mso-
fareast-language: EN-US; mso-bidi-language: AR-SA">
                Serial test:</td>
            <td>
                <asp:TextBox ID="TextBox23" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td class="style3"
                style="mso-ascii-theme-font: major-bidi;
mso-fareast-font-family: Calibri; mso-fareast-theme-font:
minor-latin; mso-hansi-theme-font: major-bidi; mso-bidi-
theme-font: major-bidi; mso-ansi-language: EN-AU; mso-
fareast-language: EN-US; mso-bidi-language: AR-SA">
                Poker test:</td>
            <td>
                <asp:TextBox ID="TextBox24" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                <asp:TextBox ID="TextBox25" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                S in decimal:</td>
            <td>
                <asp:TextBox ID="TextBox26" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                First party private key:</td>
            <td>
                <asp:TextBox ID="TextBox27" runat="server"
Width="500px"></asp:TextBox>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                First party public key:</td>
            <td>
                <asp:TextBox ID="TextBox28" runat="server"
Width="500px"></asp:TextBox>
            </td>
```

```
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                <asp:Image ID="Image2" runat="server"
ImageUrl="~/Image2.gif" Visible="False" />
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                <asp:Label ID="Label22" runat="server"
Text="Label"></asp:Label>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                <asp:Label ID="Label23" runat="server"
Text="Label"></asp:Label>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                <asp:Label ID="Label24" runat="server"
Text="Label"></asp:Label>
            </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                 </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                 </td>
            <td>
                 </td>
        </tr>
        <tr>
            <td>
                 </td>
            <td>
                 </td>
            <td>
                 </td>
        </tr>
    </table>
    </form>
</body>
```

```
</html>
```

A.2 Second web page codes

A.2.1 C# code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Numerics;

public partial class Default2 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label1.Text = (string)(Session["field1"]);
        Label2.Text = (string)(Session["field2"]);
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Image1.Visible = true;

        //uint secretkey = 1;
        long publicvalue = Convert.ToInt64(Label2.Text);
        long basevalue = Convert.ToInt64(TextBox1.Text);

        long secretkey = basevalue;
        //Label4.Text =
Convert.ToString(secretkey*secretkey);
        for (long i = 0; i < publicvalue; i++)
        {
            secretkey = secretkey * basevalue;
            secretkey = secretkey % 100000007;
        }


        Label3.Text = Convert.ToString(secretkey);

        Image1.Visible = false;
    }
}
```

A.2.2 HTML code:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<script runat="server">


</script>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div style="text-align: center">
```

```
    </div>
    Alice private key is :<asp:Label ID="Label1"
runat="server" Text="Label"></asp:Label>
    <br />
    Alice public key is :<asp:Label ID="Label2"
runat="server" Text="Label"></asp:Label>
    <br />
    <br />
    Bob private key is:
    <asp:TextBox ID="TextBox1"
runat="server">6629794</asp:TextBox>
       
    <asp:Button ID="Button1" runat="server"
onclick="Button1_Click"
        Text="Calculate the shared key" />
    <br />
    <asp:Label ID="Label4" runat="server"></asp:Label>
    <br />
    <br />
    The shared key is :<asp:Label ID="Label3"
runat="server"></asp:Label>
    <p>
         </p>
    <p>
         </p>
    <p style="text-align: center">
        <asp:Image ID="Image1" runat="server"
ImageUrl="~/Image2.gif"
            style="text-align: center" Visible="False" />
    </p>
    <p>
         </p>
    <p>
         </p>
    <p style="text-align: center">
         </p>
    </form>
</body>
</html>
```

A.3 Screen shots

A.3.1 Screen shot 1

Alice private key is :85422243
Alice public key is :15071649

Bob private key is: [ ] [Calculate the shared key]

The shared key is :

Figure A.1 Screen shot 1

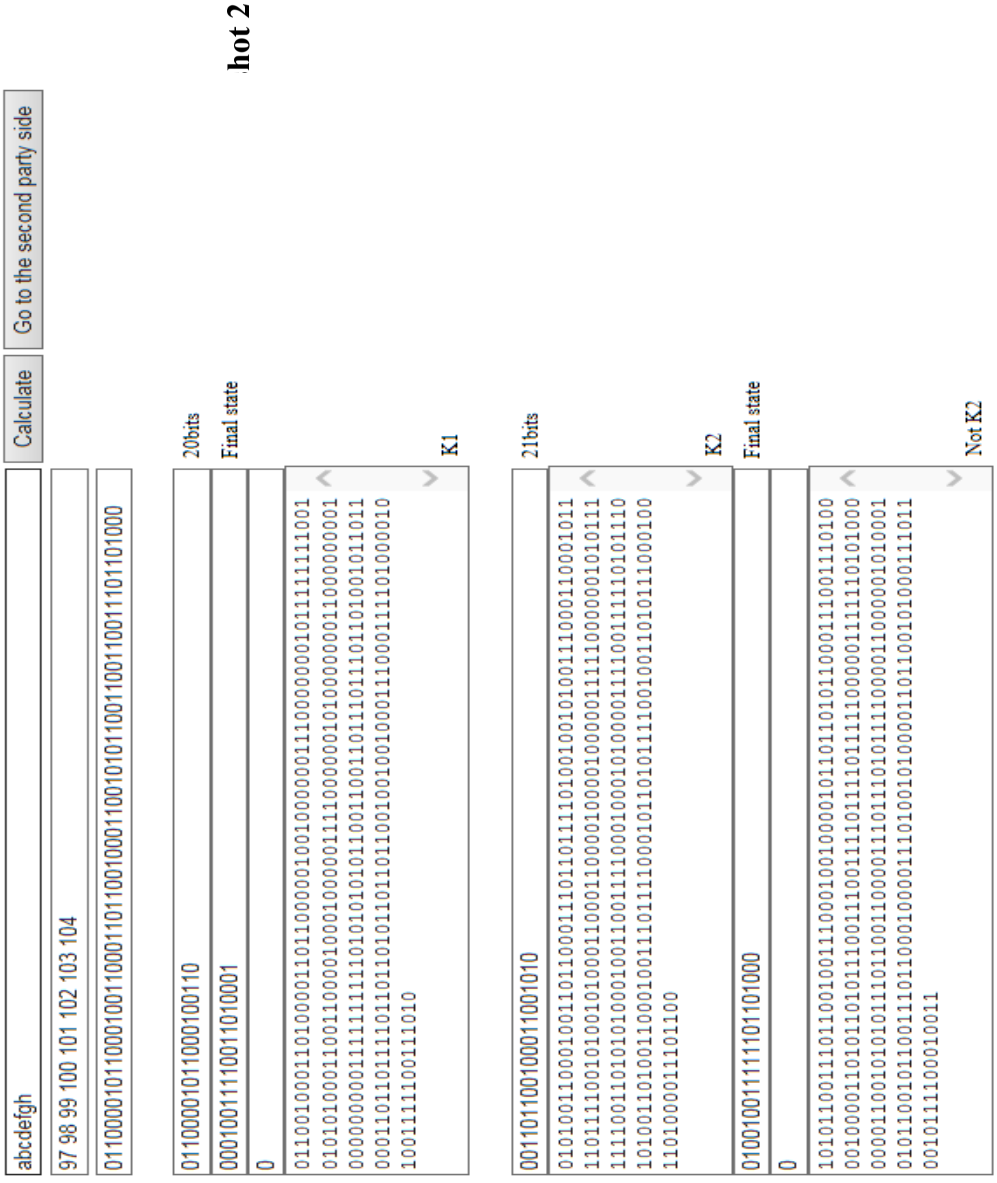


Figure A.3 Screen shot 2 - part 1

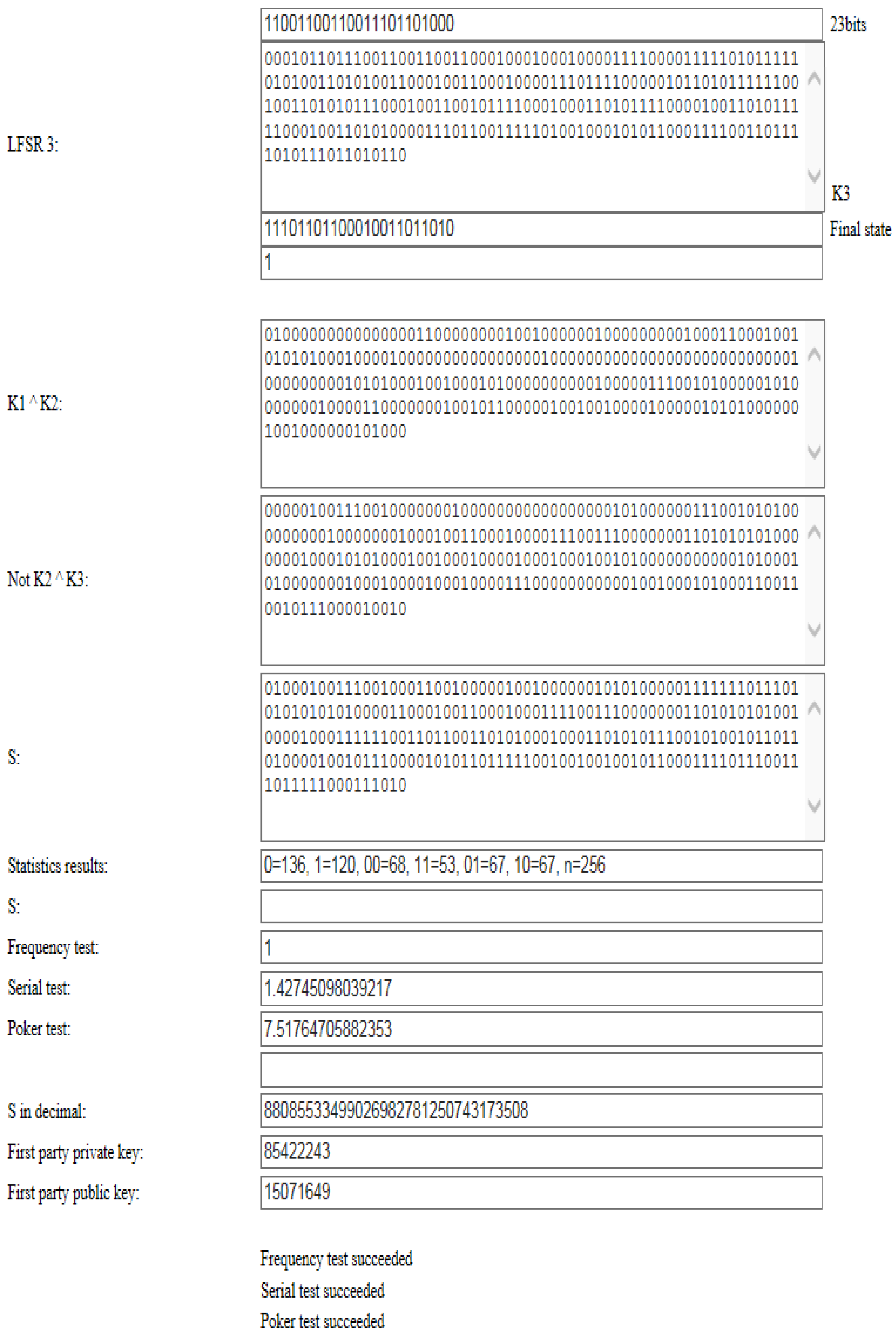


Figure A.3 Screen shot 2 - part 2

Appendix B

The practical aspects of Chapter 5 were implemented as a console application using Visual Studio 2010 and C# language. The PBC mode was implemented and tested it within this console application.

B.1 The PBC mode of operation C# code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Diagnostics;
using System.Threading;
using System.Threading.Tasks;
using System.Security.Cryptography;

namespace ConsoleApplicationPBC
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("CipherMode: Parallel Block
Cipher (PBC)");
            Stopwatch sw = Stopwatch.StartNew();
            ////////////////////////////////
            var t01 = Task.Factory.StartNew(() =>
            {
                Stopwatch sw0 = Stopwatch.StartNew();
                string pathSrc = "C:\\test.iso";
                String md5Result;
                StringBuilder sb = new StringBuilder();
                MD5 md5Hasher = MD5.Create();
                using (FileStream fs =
File.OpenRead(pathSrc))
                {
                    foreach (Byte b in
md5Hasher.ComputeHash(fs))

sb.Append(b.ToString("x2").ToLower());
                }
                md5Result = sb.ToString();
                string hash = md5Result;
                Console.WriteLine("Hash: {0}", hash);
                string password = hash.Substring(0, 1) +
hash.Substring(4, 1) + hash.Substring(8, 1) +
hash.Substring(12, 1) + hash.Substring(16, 1) +
hash.Substring(20, 1) + hash.Substring(24, 1) +
hash.Substring(28, 1);
                Console.WriteLine("Key:  {0}", password);
                sw0.Stop();
                Console.WriteLine("Hashing time: " +
sw0.Elapsed);
            });
```

```
            ////////////////////////////////////
            var t02 = Task.Factory.StartNew(() =>
            {
                Stopwatch sw1 = Stopwatch.StartNew();
                SplitFile("C:\\test.iso", "C:\\", 8);

//SplitFile("C:\\Users\\u1050771\\Desktop\\file\\test_encryp
ted.iso", "C:\\Users\\u1050771\\Desktop\\file", 8);
                sw1.Stop();
                Console.WriteLine("Splitting time: " +
sw1.Elapsed);
            });
            var tasklist12 = new List<Task> { t01, t02 };
            Task.WaitAll(tasklist12.ToArray());
            ////////////////////////////////////



            Stopwatch sw2 = Stopwatch.StartNew();
            var t1 = Task.Factory.StartNew(() =>
            {
                EncryptFile("C:\\test.iso.0000.part",
"C:\\test_encrypted.iso.0000.part", "aqeel",1);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0000.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0000
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0000.part");
            });
            var t2 = Task.Factory.StartNew(() =>
            {
                EncryptFile("C:\\test.iso.0001.part",
"C:\\test_encrypted.iso.0001.part", "aqeel",2);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");

//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0001.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0001
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0001.part");
            });
            var t3 = Task.Factory.StartNew(() =>
            {
                EncryptFile("C:\\test.iso.0002.part",
"C:\\test_encrypted.iso.0002.part", "aqeel",3);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0002.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0002
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0002.part");
            });
            var t4 = Task.Factory.StartNew(() =>
```

```
            {
                EncryptFile("C:\\test.iso.0003.part",
"C:\\test_encrypted.iso.0003.part", "aqeel",4);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0003.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0003
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0003.part");
            });
            var t5 = Task.Factory.StartNew(() =>
            {
                EncryptFile("C:\\test.iso.0004.part",
"C:\\test_encrypted.iso.0004.part", "aqeel",5);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0004.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0004
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0004.part");
            });
            var t6 = Task.Factory.StartNew(() =>
            {
                EncryptFile("C:\\test.iso.0005.part",
"C:\\test_encrypted.iso.0005.part", "aqeel",6);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0005.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0005
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0005.part");
            });
            var t7 = Task.Factory.StartNew(() =>
            {
                EncryptFile("C:\\test.iso.0006.part",
"C:\\test_encrypted.iso.0006.part", "aqeel",7);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0006.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0006
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0006.part");
            });
            var t8 = Task.Factory.StartNew(() =>
            {
```

```
                EncryptFile("C:\\test.iso.0007.part",
"C:\\test_encrypted.iso.0007.part", "aqeel",8);

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test.iso.
0000.part");


//DecryptFile("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0007.part",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0007
.part", "aqeel");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0007.part");
            });


            var tasklist = new List<Task> { t1, t2, t3, t4,
t5, t6, t7, t8 };
            Task.WaitAll(tasklist.ToArray());
            sw2.Stop();
            Console.WriteLine("Encrypting|Decrypting time: "
+ sw2.Elapsed);
            ////////////////////////////////////

            Stopwatch sw3 = Stopwatch.StartNew();



            var t9 = Task.Factory.StartNew(() =>
            {
                ConcatenateFiles("C:\\test_encrypted.iso",
"C:\\test_encrypted.iso.0000.part",

"C:\\test_encrypted.iso.0001.part",

"C:\\test_encrypted.iso.0002.part",

"C:\\test_encrypted.iso.0003.part",

"C:\\test_encrypted.iso.0004.part",

"C:\\test_encrypted.iso.0005.part",

"C:\\test_encrypted.iso.0006.part",

"C:\\test_encrypted.iso.0007.part");

File.Delete("C:\\test_encrypted.iso.0000.part");

File.Delete("C:\\test_encrypted.iso.0001.part");

File.Delete("C:\\test_encrypted.iso.0002.part");

File.Delete("C:\\test_encrypted.iso.0003.part");

File.Delete("C:\\test_encrypted.iso.0004.part");

File.Delete("C:\\test_encrypted.iso.0005.part");

File.Delete("C:\\test_encrypted.iso.0006.part");

File.Delete("C:\\test_encrypted.iso.0007.part");


//ConcatenateFiles("C:\\Users\\u1050771\\Desktop\\file\\test
_decrypted.iso",
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0000
.part",
```

```
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0001
.part",
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0002
.part",
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0003
.part",
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0004
.part",
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0005
.part",
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0006
.part",
                //
"C:\\Users\\u1050771\\Desktop\\file\\test_decrypted.iso.0007
.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0000.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0001.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0002.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0003.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0004.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0005.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0006.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_decr
ypted.iso.0007.part");


            });
            var t10 = Task.Factory.StartNew(() =>
            {
                File.Delete("C:\\test.iso.0000.part");
                File.Delete("C:\\test.iso.0001.part");
                File.Delete("C:\\test.iso.0002.part");
                File.Delete("C:\\test.iso.0003.part");
                File.Delete("C:\\test.iso.0004.part");
                File.Delete("C:\\test.iso.0005.part");
                File.Delete("C:\\test.iso.0006.part");
                File.Delete("C:\\test.iso.0007.part");


//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0000.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0001.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0002.part");
```

```
        {
            // Store the file in a byte array
            Byte[] byteSource =
System.IO.File.ReadAllBytes("C:\\test.iso");
            // Get file info
            FileInfo fiSource = new
FileInfo("C:\\test.iso");
            // Calculate the size of each part
            int partSize =
(int)Math.Ceiling((double)(fiSource.Length / OutputFiles));
            // The offset at which to start reading from the
source file
            int fileOffset = 0;

            // Stores the name of each file part
            string currPartPath;
            // The file stream that will hold each file part
            FileStream fsPart;
            // Stores the remaining byte length to write to
other files
            int sizeRemaining = (int)fiSource.Length;

            // Loop through as many times we need to create
the partial files
            for (int i = 0; i < OutputFiles; i++)
            //Parallel.For(0, OutputFiles, i =>
            {
                // Store the path of the new part
                currPartPath = FolderOutputPath + "\\" +
fiSource.Name + "." + String.Format(@"{0:D4}", i) + ".part";
                // A filestream for the path
                if (!File.Exists(currPartPath))
                {

//File.Create(currPartPath).Dispose();//;)
```

```
//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0003.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0004.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0005.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0006.part");

//File.Delete("C:\\Users\\u1050771\\Desktop\\file\\test_encr
ypted.iso.0007.part");

            });

            var tasklist9 = new List<Task> { t9, t10 };
            Task.WaitAll(tasklist9.ToArray());
            sw3.Stop();
            Console.WriteLine("Joining time: " +
sw3.Elapsed);
            ////////////////////////////////////

            sw.Stop();
            Console.WriteLine("Total time: " + sw.Elapsed);
            Console.WriteLine("Processes completed.. Press
any ket to exit...");
            Console.ReadKey();
            ////////////////////////////////////
        }

        private static void SplitFile(string FileInputPath,
string FolderOutputPath, int OutputFiles)
```

```
                    fsPart = new FileStream(currPartPath,
FileMode.CreateNew);
                    // Calculate the remaining size of the
whole file
                    sizeRemaining = (int)fiSource.Length -
(i * partSize);
                    // The size of the last part file might
differ because a file doesn't always split equally
                    if (i == (OutputFiles - 1))
                    //if (sizeRemaining < partSize)
                    {
                        partSize = sizeRemaining;
                    }
                    // Write the byte chunk to the part file
                    fsPart.Write(byteSource, fileOffset,
partSize);
                    // Close the file stream
                    fsPart.Close();
                    // Set the new offset
                    fileOffset += partSize;
                }
            }
        }

        //public static void SplitFile(string SourceFile,
string Split, int nNoofFiles)
        //{

        //    try
        //    {
        //        FileStream fs = new FileStream(SourceFile,
FileMode.Open, FileAccess.Read);
        //        int SizeofEachFile =
(int)Math.Ceiling(Convert.ToDouble((double)fs.Length /
nNoofFiles));//
        //        for (int i = 0; i < nNoofFiles; i++)
        //        {
        //            string baseFileName =
Path.GetFileNameWithoutExtension(SourceFile);
        //            string Extension =
Path.GetExtension(SourceFile);
        //            FileStream outputFile = new
FileStream(Path.GetDirectoryName(SourceFile) + "\\" +
baseFileName +
        //                Extension + "." +
i.ToString().PadLeft(4, Convert.ToChar("0")) + ".part",
FileMode.Create, FileAccess.Write);
        //            string mergeFolder =
Path.GetDirectoryName(SourceFile);
        //            int bytesRead = 0;
        //            byte[] buffer = new
byte[SizeofEachFile];
        //            if ((bytesRead = fs.Read(buffer, 0,
SizeofEachFile)) > 0)
        //            {
        //                outputFile.Write(buffer, 0,
bytesRead);
        //                //outp.Write(buffer, 0,
BytesRead);
        //                string packet = baseFileName + "."
+ i.ToString().PadLeft(3, Convert.ToChar("0")) +
Extension.ToString();
        //                //Packets.Add(packet);
        //            }
        //            outputFile.Close();
        //        }
        //        fs.Close();
        //    }
        //    catch (Exception Ex)
        //    {
```

```
        //        throw new ArgumentException(Ex.Message);
        //    }


        //}

        static void ConcatenateFiles(string outputFile,
params string[] inputFiles)
        {
            using (Stream output =
File.OpenWrite(outputFile))
            {
                foreach (string inputFile in inputFiles)
                {
                    using (Stream input =
File.OpenRead(inputFile))
                    {
                        input.CopyTo(output);
                    }
                }
            }
        }

        private static void EncryptFile(string inputFile,
string outputFile, string password, int i)
        {

            try
            {
                //string password = @"myKey123"; // Your Key
Here
                UnicodeEncoding UE = new UnicodeEncoding();
                byte[] key = UE.GetBytes(password);

                string cryptFile = outputFile;
                FileStream fsCrypt = new
FileStream(cryptFile, FileMode.Create);

                RijndaelManaged RMCrypto = new
RijndaelManaged();
                RMCrypto.Mode = CipherMode.ECB;
                //RMCrypto.Padding = PaddingMode.Zeros;
                CryptoStream cs = new CryptoStream(fsCrypt,
RMCrypto.CreateEncryptor(key, key), CryptoStreamMode.Write);

                FileStream fsIn = new FileStream(inputFile,
FileMode.Open);

                int data;
                while ((data = fsIn.ReadByte()) != -1)
                    cs.WriteByte((byte)data);


                fsIn.Close();
                cs.Close();
                fsCrypt.Close();
                Console.WriteLine("Encryption success on
Server "+i, "Error");
            }
            catch
            {
                Console.WriteLine("Encryption failed!",
"Error");
            }
        }

        private static void DecryptFile(string inputFile,
string outputFile, string password)
        {
            try
```

```
            {
                //string password = @"myKey123"; // Your Key
Here
                UnicodeEncoding UE = new UnicodeEncoding();
                byte[] key = UE.GetBytes(password);

                FileStream fsCrypt = new
FileStream(inputFile, FileMode.Open);

                RijndaelManaged RMCrypto = new
RijndaelManaged();
                RMCrypto.Mode = CipherMode.ECB;
                //RMCrypto.Padding = PaddingMode.Zeros;

                CryptoStream cs = new CryptoStream(fsCrypt,
                    RMCrypto.CreateDecryptor(key, key),
                    CryptoStreamMode.Read);

                FileStream fsOut = new
FileStream(outputFile, FileMode.Create);

                int data;
                while ((data = cs.ReadByte()) != -1)
                    fsOut.WriteByte((byte)data);

                fsOut.Close();
                cs.Close();
                fsCrypt.Close();
                Console.WriteLine("Decryption success!",
"Error");
            }
            catch
            {
                Console.WriteLine("Decryption failed!",
"Error");
            }
        }}}
```

Appendix C

C.1 Sequence diagrams generation codes

Sequence diagrams shown in Chapter 7 were drawn using Quick Sequence Diagram Editor 4.2, and using the following generation codes.

Table C.1 Sequence diagrams generation codes

First scenario	Second scenario	Third scenario
Admin:Actor Client:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listenning to port() Admin:p.Listenning to port() +1 Client First scenario { +1 (1) Client:d.Request data() d:IP address not found=b.Checking IP address() d:Normal packets were detected=c.Packets stream c:Process=c.Classify c:stop d:cs.Clean packet (2)cs:Client.Send data +2 d First scenario } +2	Admin:Actor Attacker:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listenning to port() Admin:p.Listenning to port() +3 Attacker Second scenario { +3 (3)Attacker:d.Request data() d:IP address not found=b.Checking IP address() d:Abnormal packets were detected=c.Packets stream() c:Process=c.Classify c:stop d:p.Send abnormal packets() p:Packets termination=p.Process p:b.Add the IP address to the blacklist() d:Attacker.Request rejected p:cs.Warrning message() cs:Attacker.Warrning message (4)p:Admin.Send info & alarm +4 b Second scenario } +4	Admin:Actor Attacker:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listenning to port() Admin:p.Listenning to port() +5 Attacker Third scenario { +5 (5)Attacker:d.Request data() d:IP address found=b.Checking IP address() d:Terminating IP address=p.Attacker IP address found() d:Attacker.Request rejected d:cs.Warrning message() cs:Attacker.Warrning message (6)d:Admin.Send info & alarm +6 Admin Third scenario } +6

Appendix D

Appendix D includes a copy of the following related publications:

1. Sahi, A, Lai, D, Li, Y & Diykh, M 2017, 'An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment', *IEEE Access*, vol. 5, pp. 6036-48. **(Q1 journal, Impact Factor: 3.244, published)**
2. Sahi, A, Lai, D & Li, Y, 'A review of the state of the art in privacy and security in the eHealth cloud', *Computer Science Review*, **(Q1 journal, Impact Factor: 7.63, Submitted)**
3. Sahi, A, Lai, D & Li, Y 2016, 'Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan', *Computers in biology and medicine*, vol. 78, pp. 1-8. **(Q2 journal, Impact Factor: 2.115, published)**
4. Sahi, A, Lai, D & Li, Y, 'Three-Party Password-Based Authenticated Key Exchange Protocol Based on the Computational Diffie-Hellman Assumption', *International Journal of Communication Networks and Distributed Systems*, **(Q3 journal, Impact Factor: 0.75, published)**
5. Sahi, A & Lai, D 2015, 'Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol', in *22nd International Conference on Telecommunications (ICT),*: *proceedings of the* IEEE, pp. 204-8. **(IEEE conference in Sydney, published)**
6. Sahi, A, Lai, D & Li, Y, 'An efficient hash-based parallel block cipher mode of operation', published in *the 4th International Conference on Computer and Communication Systems (ICCCS2018), Japan.* **(IEEE conference in Japan, published)**
7. Sahi, A, Lai, D & Li, Y 2015, 'Parallel Encryption Mode for Probabilistic Scheme to Secure Data in the Cloud', in *10th International Conference on Information Technology and Applications (ICITA)* Sydney. **(IEEE conference in Sydney, published)**
8. Sahi, A, Lai, D & Li, Y 2017, 'An Energy Efficient TCP DoS Attacks Mitigation Method in Cloud Computing', in *The First MoHESR and HCED Iraqi Scholars Conference in Australasia 2017*, Swinburne University of Technology, pp. 289-94. **(A conference at Swinburne University of Technology, published)**

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 16, 2017, accepted March 19, 2017, date of publication April 6, 2017, date of current version May 17, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2688460

An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment

AQEEL SAHI[1,2], DAVID LAI[2], YAN LI[2], (Member, IEEE), AND MOHAMMED DIYKH[1,2]

[1]Thi-Qar University, Nasiriya 64001, Iraq
[2]School of Agricultural, Computational and Environmental Sciences, University of Southern Queensland, Toowoomba, QLD 4350, Australia

Corresponding author: A. Sahi (akeel_sahy@yahoo.co.uk)

ABSTRACT Although the number of cloud projects has dramatically increased over the last few years, ensuring the availability and security of project data, services, and resources is still a crucial and challenging research issue. Distributed denial of service (DDoS) attacks are the second most prevalent cybercrime attacks after information theft. DDoS TCP flood attacks can exhaust the cloud's resources, consume most of its bandwidth, and damage an entire cloud project within a short period of time. The timely detection and prevention of such attacks in cloud projects are therefore vital, especially for eHealth clouds. In this paper, we present a new classifier system for detecting and preventing DDoS TCP flood attacks (CS_DDoS) in public clouds. The proposed CS_DDoS system offers a solution to securing stored records by classifying the incoming packets and making a decision based on the classification results. During the detection phase, the CS_DDOS identifies and determines whether a packet is normal or originates from an attacker. During the prevention phase, packets, which are classified as malicious, will be denied to access the cloud service and the source IP will be blacklisted. The performance of the CS_DDoS system is compared using the different classifiers of the least squares support vector machine (LS-SVM), naïve Bayes, K-nearest, and multilayer perceptron. The results show that CS_DDoS yields the best performance when the LS-SVM classifier is adopted. It can detect DDoS TCP flood attacks with about 97% accuracy and with a Kappa coefficient of 0.89 when under attack from a single source, and 94% accuracy with a Kappa coefficient of 0.9 when under attack from multiple attackers. Finally, the results are discussed in terms of accuracy and time complexity, and validated using a K-fold cross-validation model.

INDEX TERMS Classification, cloud computing, DDoS attacks, LS-SVM.

I. INTRODUCTION

Distributed denial of service (DDoS) TCP flood attacks are DoS attacks in which attackers flood a victim machine with packets in order to exhaust its resources or consume bandwidth [1]. As the attack may be distributed over multiple machines, it will be very hard to differentiate authentic users from attackers. In fact, a DDoS flood attack is not only a widespread attack; it is the second most common cybercrime attack to cause financial losses [2] according to the United States Federal Bureau of Investigation (FBI).

The use of cloud computing is quickly increasing in many sectors, and especially in the health sector, as a result of its vital features, such as availability and on-demand services [3]. Most people think of cloud computing as virtual network which can offer flexible and accessible on-demand services [4]. However, the author in [5] pointed out that cloud computing involves much more than this, which has led researchers to re-consider its security more seriously. In addition, as mentioned in an electronic cybercrime study published by KPMG in collaboration with eCrime Congress in 2009, most of the cloud's virtual clients are under threat, and these threats increase as time passes [6].

There are many procedures [7] which can be adopted to mitigate the DDoS flood attacks, such as classifications [8], [9], encryption techniques [10]–[12]. As DDoS flood attacks can be implemented in many forms, the form of these attacks cannot be foreseen. Therefore, our new proposed classifier system for the detection and prevention of DDoS TCP flood attacks (CS_DDoS) is classification based, and can identify these attacks data regardless of the form in which they arrive at the cloud system. Classification can be defined as a common procedure for classifying, distinguishing and

differentiating multiple objects. Different classifiers, such as least squares support vector machine (LS-SVM), naïve Bayes, K-nearest and multilayer perceptron [13], [14] are used in this study to perform the classification process.

This paper is organized as follows: in Section 2, we review related work, and Section 3 introduces the simulation platform, with and without DDoS TCP flood attacks. Section 4 presents our proposed CS_DDoS system, and its performance is evaluated and validated in Section 5. Finally, we conclude this study and discuss future work in Section 6.

II. RELATED WORK

Many detection and prevention methods for mitigating DDoS flood attacks have been reported in the last few years [15].

The rank correlation-based detection (RCD) scheme was proposed by Wei et al. in [16]. The authors of the RCD claimed that their scheme could distinguish whether the incoming requests were from genuine users or from attackers. In [17], the ALPi algorithm was introduced, which decreased difficulties in packet flows and improved functionality by extending the concept of packet scoring. The ALPi therefore raises the detection accuracy percentage and attack recognition. Another DDoS attack prevention architecture, known as secure overlay services (SOS), was presented in [18]. The SOS architecture is a combination of three parts: secure overlay tunneling, routing via consistent hashing, and filtering. The authors claimed that the SOS can successfully decrease the probability of these attacks using filtering close to the secure edge and randomness close to the front edge.

Moreover, Wang and Reiter proposed the web referral architecture for privileged service (WRAPS) [19]. The WRAPS adopted the structure of a web graph to resist DDoS flood attacks, and requires authentic users to be authenticated using a referral hyperlink from a trusted site. Another approach was introduced to detect application DoS attacks on backend servers called the group testing-based approach [20]. The authors extended the existing group testing approach by reallocating users' requests to several servers. Markov Chain probability theory was adopted by Salah et al. when proposing an analytical queuing approach which examines the performance of firewalls under DDoS attacks [21].

In addition, Dou et al. [22] presented a confidence-based filtering (CBF) scheme for cloud projects. In the CBF, packets of information from authentic users is gathered during non-attack periods to extract features, which can generate an information profile of these non-attack periods. With this profile, the CBF scheme will be endorsed using a packet-scoring calculation during attacks to make a decision on whether to remove these packets or not. Another approach to detecting flood attacks, the fast lightweight detection approach, was presented by Yu et al. [23]. This approach utilized SNMP-MIB (simple network management protocol-management information base) statistical data as an alternative to raw data, as well as a SVM classifier for attack classification. Lee et al. [24] introduced a practical DDoS detection scheme based on DDoS architecture. In this scheme, they selected variables based on particular features that were extracted from a DDoS architecture. A cloud trace back (CTB) method was proposed in [25]. The authors of the CTB claimed that their method could identify the sources of the attacks. They also proposed a cloud Protector (CP), which made use of a back-propagation classifier in order to detect such attacks.

Furthermore, a new framework was presented by Lu et al. in [26]. This framework was able to effectively identify compromised packets. It analyzed these packets at the router end using a perimeter-based DDoS prevention system. Wang et al. introduced a graphics-based DDoS attack prevention and detection scheme, which was able to work with the data shift issue [7]. This scheme works by prevention, using network monitoring and a precise response with an elastic control structure. In [27], an adaptive selective verification (ASV) system was proposed. The ASV does not rely on network assumptions, and utilizes bandwidth efficiently. Another approach was presented based on five features (average number of packets per flow, percentage of correlative flow, one-direction generating speed, ports generating speed, and percentage of abnormal packets) combined with a Bloom filter [28]. In this approach, only users on the whitelist are allowed to reach their destinations; this whitelist is generated to include legitimate users only. However, this approach was implemented on the switches side (i.e. in hardware), which makes any future amendments or updates challenging [29].

While many mechanisms have been proposed to detect and prevent DDoS flood attacks, most of these do not provide high accuracy and are not efficient or fast detection and prevention techniques [30]. Furthermore, many of the DDoS attack protection mechanisms described here face scalability issues due to the fact that networks are becoming larger and faster; in addition, industrial deployment needs to be considered [17].

Therefore, cloud computing needs an efficient DDoS mitigation approach that can offer fast and accurate detection while remaining scalable. The proposed CS_DDoS was designed with all of these factors in mind.

III. DDoS TCP FLOOD ATTACKS

DDoS attacks can be established in two different ways: either directly and/or indirectly [31], [32]. Direct attacks target a weakness in the system of the victim machines and damage the machines directly. On the other hand, indirect attacks do not target victim machines directly; they prey on other elements with which the victim machines are associated and hinder their work [33]. In the following discussion, the TCP flood attack is used; this is an indirect attack, as it consumes most of the network's resources, meaning that they are not readily available to other users.

A TCP flood attack was carried out using software on a virtual cloud network; Wireshark Network Analyzer 2.0.0 [34] was used to capture and analyze traffic both before and during the attack.

A. BEFORE THE ATTACK

The network was simulated as shown in Figure 1.

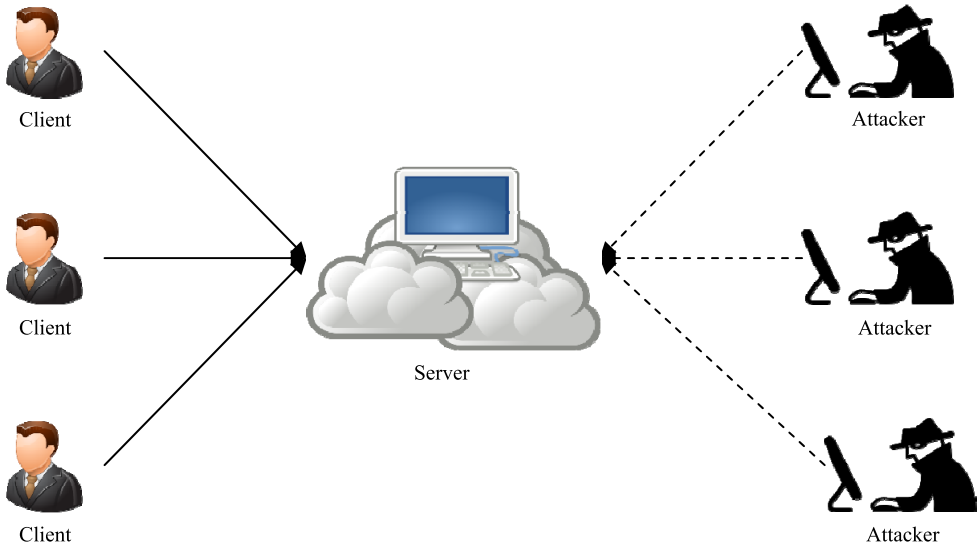


FIGURE 1. Test network architecture.

Firstly, using TCP Ping, we sent 50 TCP test probes (pings) to a server (server machine 10.25.129.5:80). The reply took 1.3 ms on average, as shown below:

```
Ping statistics for 10.25.129.5:80
  50 probes sent.
Approximate trip times in milliseconds:
  Minimum = 0.25 ms, Maximum = 26.065 ms,
  Average = 1.323 ms
```

The TCP protocol uses several flags to manage the state of a connection in the packet header [35]. We focused on two of these, which are used in establishing TCP connections:

- SYN (Synchronize) which represents the initiation of a connection; and
- ACK (Acknowledge) which represents data received.

We monitored the traffic of the 50 probes at the server machine using Wireshark, by capturing the packets that were associated with the server using the filter "ip.addr = = 10.25.129.5". As the traffic was normal, the server machine replied to all requested packets according to the TCP protocol, as shown in Figure 2 (a and b).

In addition, the I/O graph was stable. All packets were answered and almost no TCP errors occurred. Note that the number of requesting packets was approximately less than 10 per second, as shown in Figure 3.

B. DURING THE ATTACK

An attack was launched using a software program which performed a DDoS TCP flood attack on a particular server. Once the DDoS TCP flood attack commenced on the victim machine in the cloud, the arriving packets were much more numerous than the server could handle. Consequently, the server could not respond to all the requesting packets from either normal users or the attackers. Note that 10.25.129.5 was the IP address of the victim server and 10.31.133.235 was the IP address of the attacker. The first request packet from the attacker was successful, as it was treated like a normal requesting packet. The subsequent ones were not successful, as the server was too busy and could not respond. A screen shot of the packet capture is shown in Figure 4 (a and b).

Finally, we sent 50 TCP test probes within a few seconds to the victim machine during the attack period to test the connection. The reply time was 9.6 ms on average, which differs considerably from the first test as shown below:

```
Ping statistics for 10.25.129.5:80
  50 probes sent.
Approximate trip times in milliseconds:
  Minimum = 0.181 ms, Maximum = 152.341 ms,
Average = 9.586 ms
```

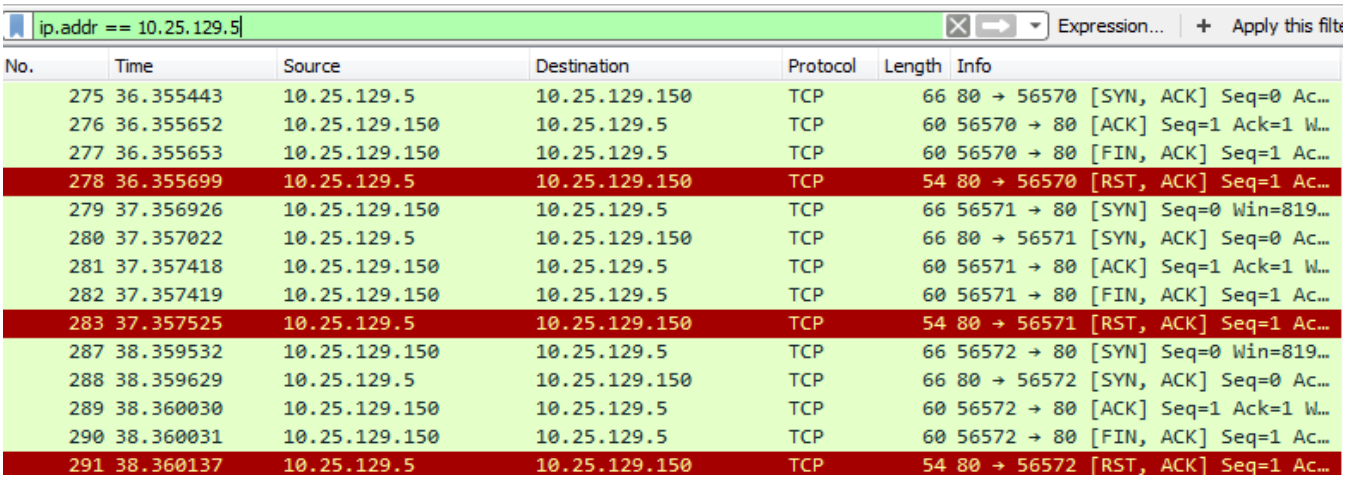
To sum up, the DDoS TCP flood attack can affect the cloud server's performance within a short time, slowing down the response, and can even stop the service completely. TCP errors will also be increased, as shown in Figure 5. Therefore, an efficient and effective detection and prevention technique is required.

IV. THE PROPOSED CS_DDoS SYSTEM

In this section we present the proposed CS_DDoS system, which can prevent DDoS TCP flood attacks. Firstly, it was assumed that the IP addresses of the attackers are not spoofed. Examples of how to prevent IP spoofing can be found in [36]. Our proposed system includes two sub-systems: the detection sub-system and prevention sub-system, as shown in Figure 6.

A. DETECTION PHASE

During the detection phase, the detection sub-system collects the incoming packets within a time frame, for example 60 seconds. The collected packets are subjected to a blacklist check to test whether their sources are blacklisted as attackers of the cloud system. If the packet source is listed in the attacker blacklist, the detection system will send the packets directly to the prevention sub-system without further processing.



ip.addr == 10.25.129.5 — Expression... — Apply this filt

No.	Time	Source	Destination	Protocol	Length	Info
275	36.355443	10.25.129.5	10.25.129.150	TCP	66	80 → 56570 [SYN, ACK] Seq=0 Ac…
276	36.355652	10.25.129.150	10.25.129.5	TCP	60	56570 → 80 [ACK] Seq=1 Ack=1 W…
277	36.355653	10.25.129.150	10.25.129.5	TCP	60	56570 → 80 [FIN, ACK] Seq=1 Ac…
278	36.355699	10.25.129.5	10.25.129.150	TCP	54	80 → 56570 [RST, ACK] Seq=1 Ac…
279	37.356926	10.25.129.150	10.25.129.5	TCP	66	56571 → 80 [SYN] Seq=0 Win=819…
280	37.357022	10.25.129.5	10.25.129.150	TCP	66	80 → 56571 [SYN, ACK] Seq=0 Ac…
281	37.357418	10.25.129.150	10.25.129.5	TCP	60	56571 → 80 [ACK] Seq=1 Ack=1 W…
282	37.357419	10.25.129.150	10.25.129.5	TCP	60	56571 → 80 [FIN, ACK] Seq=1 Ac…
283	37.357525	10.25.129.5	10.25.129.150	TCP	54	80 → 56571 [RST, ACK] Seq=1 Ac…
287	38.359532	10.25.129.150	10.25.129.5	TCP	66	56572 → 80 [SYN] Seq=0 Win=819…
288	38.359629	10.25.129.5	10.25.129.150	TCP	66	80 → 56572 [SYN, ACK] Seq=0 Ac…
289	38.360030	10.25.129.150	10.25.129.5	TCP	60	56572 → 80 [ACK] Seq=1 Ack=1 W…
290	38.360031	10.25.129.150	10.25.129.5	TCP	60	56572 → 80 [FIN, ACK] Seq=1 Ac…
291	38.360137	10.25.129.5	10.25.129.150	TCP	54	80 → 56572 [RST, ACK] Seq=1 Ac…

(a)

(b)

FIGURE 2. Captured packets and TCP flags (normal). (a) Captured packets. (b) TCP flags.

If the packet source is not blacklisted, the incoming packet will be passed to the classifier to decide whether the packets are normal (originating from a client) or abnormal (originating from an attacker). A packet is considered to be an attacking one if the source requests connections to the same destination more frequently than an assumed threshold. The threshold can be manually adjusted by the system administrator to cater for the varying requirements of a particular network. If a packet is considered to be normal, the detection system will send it to its destination (the cloud service provider). Otherwise, the detection sub-system will send the packet to the prevention sub-system.

Four different classifiers are used in the detection sub-system for the classification operation. The classifiers used are explained and evaluated in Section 5.

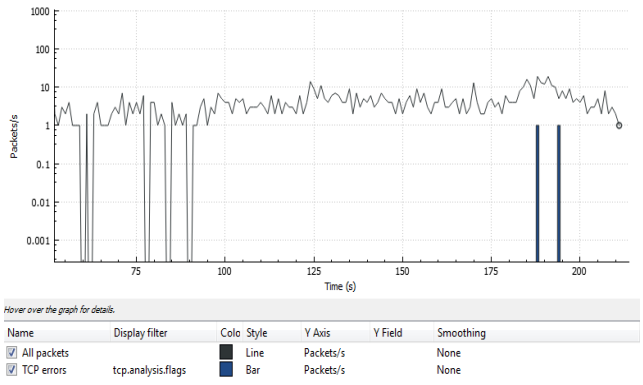
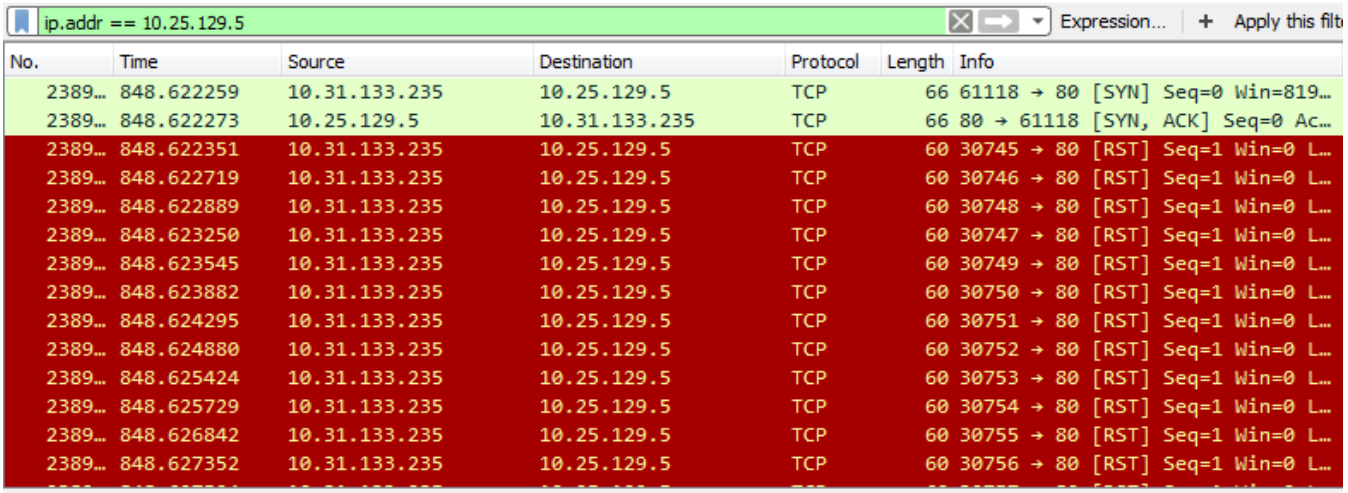


FIGURE 3. The I/O graph (no TCP errors).

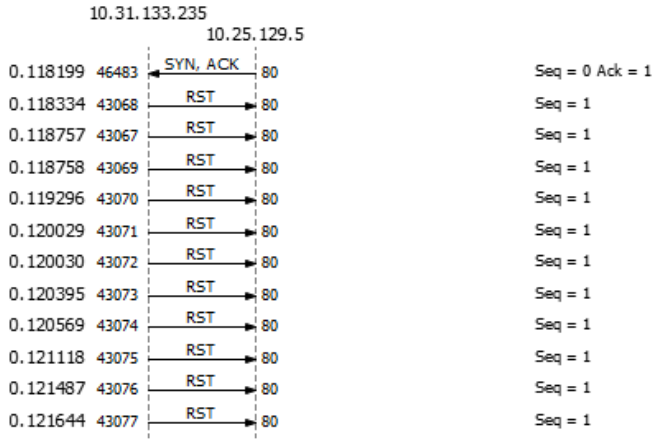
B. PREVENTION PHASE

When the packets reach the prevention system, they are considered to be attacking packets by the detection sub-system. The prevention sub-system first alerts the system administrator of the attacks. Then, the prevention sub-system will add the attacking source address to the attacker blacklist used by the detection sub-system, if it is not already on the list. Finally, the attacking packet will be dropped. The overall architecture of the CS_DDoS system is shown in Figure 6.

Algorithm 1 is used to determine whether these packets are normal or abnormal by counting the number of requests for a connection from an IP address and checking whether it



(a)



(b)

FIGURE 4. Captured packets and TCP flags (abnormal). (a) Captured packets. (b) TCP flags.

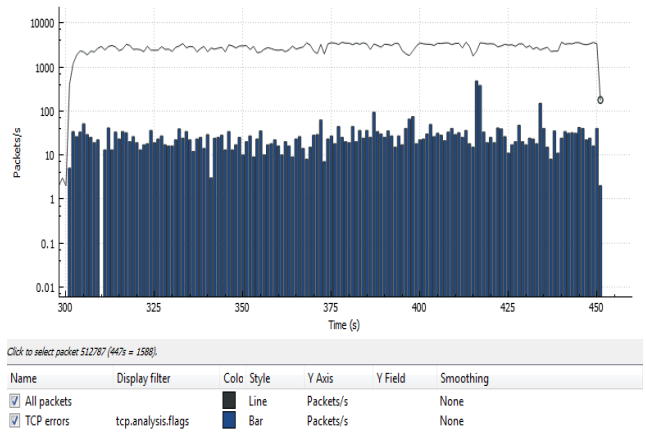


FIGURE 5. The I/O graph (with TCP errors).

exceeds a predefined threshold within a certain time frame. This algorithm is applied to the training data used for each classifier. As a result, each classifier used will predict the behavior of the attackers according to Algorithm 1.

Algorithm 1 Pre-Processing

```
Load data
For I=1: n
    P=data (I, 2)
    P2= (I, 1)
    For J=1: n
        N=find (data (J, 1) ==P2) & (data (J, 2) ==P)
        If N>=K
            New_data (I, 1) =data (I, 1)
            New_data (I, 2) =−1
        Else
            New_data (I, 1) =data (I, 1)
            New_data (I, 2) =1
End
```

where:
n is the number of packets
P is the destination IP address

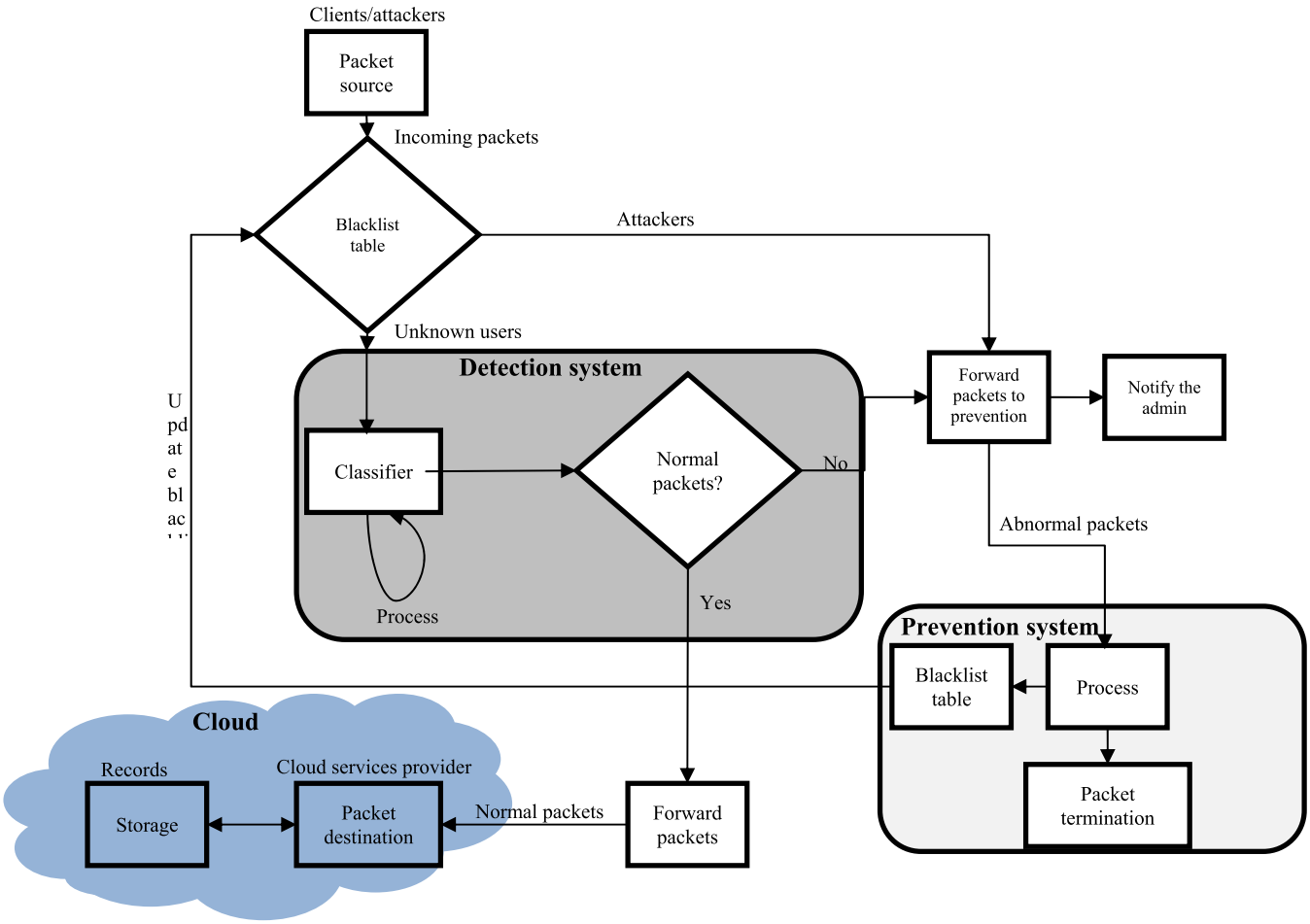


FIGURE 6. The overall architecture of the proposed CS_DDoS system.

P2 is the source IP address

N is the number of packets from the same source to the same destination within 60 seconds

K is the threshold for a packet to be considered an attacking packet

−1 indicates abnormal packets (blacklist array)

1 indicates normal packets

New_data () is a new entry list with tag "1" or "−1"

The proposed CS_DDoS system can be implemented in three possible scenarios. The first scenario is a normal service request packet. The requested service will be delivered as usual. The next scenario is when the source IP address is not blacklisted but the number of service requesting packets exceeds a predefined threshold within a certain time frame. The packet in this scenario will be considered a DDoS attack packet. The source address will be blacklisted and the packet will be dropped. The last scenario is when the source address of a packet is blacklisted and the packet is dropped without any further processing.

The three scenarios are illustrated using Quick Sequence Diagram Editor 4.2 [37]. The code used is shown in Table 11. The resulting sequence diagrams are shown in Figure 7 (a, b and c).

In case of flash crowd scenario, all packets must wait in a queue to be served sequentially.

The proposed CS_DDoS system can be used in any type of cloud, such as eHealth clouds, to ensure the security and availability of health records against DDoS TCP flood attacks.

V. EXPERIMENTAL RESULTS

A. CLASSIFICATION ALGORITHMS

In this section, we briefly explain the four commonly used classification algorithms used in our experiments. The classification algorithms are as follows:

1) LS-SVM

The LS-SVM is a powerful classifier in the field of pattern recognition for the detection of abnormalities from signals, images and time series signals. The LS-SVM is an efficient method of classifying two different sets of observations into their relevant classes. It is capable of handling high

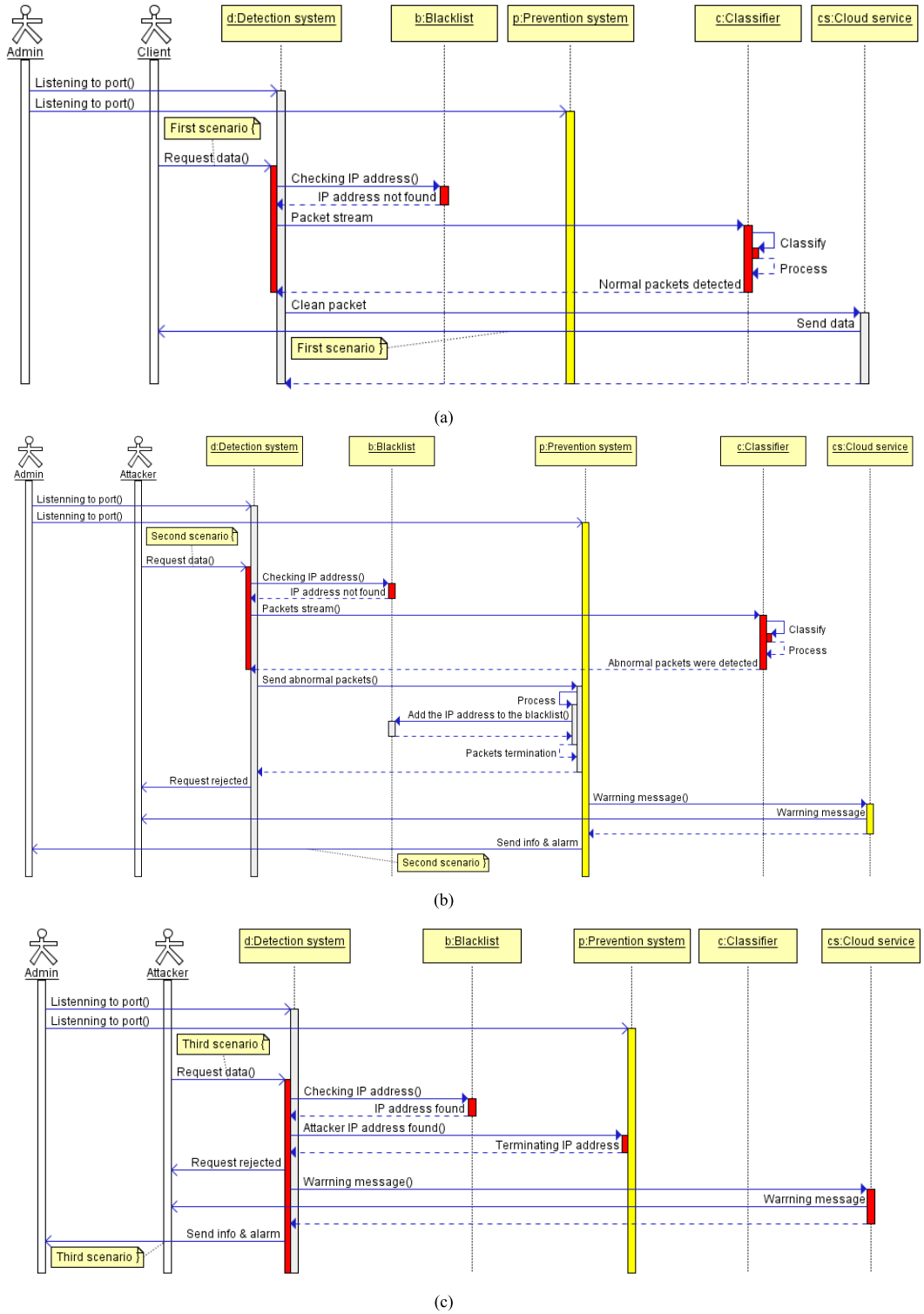


FIGURE 7. CS_DDoS possible scenarios. (a) First scenario (normal packets). (b) Second scenario (store abnormal packets in the blacklist). (c) Third scenario (abnormal packets already in the blacklist).

dimensional and non-linear data. In this work, the LS-SVM is employed to detect illegal activities in a network. The parameters of the LS-SVM are set during the training session to obtain a high proportion of detected results [38].

2) Naïve Bayes

Naïve Bayes is a frequently used classifier and has a straightforward approach based on the application of Bayes' theorem [39]. It is a simple approach which relies on proba-

TABLE 1. Classification performance measurements (n=1000 and K=100).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	99.5%	95.3%	96%	0.91
2	Naïve Bayes	80%	92.3%	93%	0.82
3	K-nearest	75%	93.5%	95%	0.74
4	Multilayer perceptron	88.3%	95.3%	97%	0.78

TABLE 2. Classification performance measurements (n=2000 and K=200).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94.6%	94%	96%	0.89
2	Naïve Bayes	82%	93%	94%	0.75
3	K-nearest	80%	95%	93%	0.87
4	Multilayer perceptron	92%	97%	97%	0.65

TABLE 3. Classification performance measurements (n=5000 and K=300).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	96%	98%	97%	0.90
2	Naïve Bayes	96%	94%	92%	0.82
3	K-nearest	82%	96%	94%	0.68
4	Multilayer perceptron	95%	99%	97%	0.75

TABLE 4. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	98%	99%	98%	0.85
2	Naïve Bayes	95%	95%	96%	0.67
3	K-nearest	85%	98%	97%	0.62
4	Multilayer perceptron	97%	99%	97%	0.58

bilistic knowledge to accurately predict test instances. This algorithm assumes that predictive attributes are conditionally independent and that there are no hidden attributes which can affect the prediction process [39]. The naïve Bayes classifier uses small training sets to provide relatively good performance, which generally overcomes any overtraining issues.

3) K-NEAREST

K-nearest is one of the most straightforward learning algorithms. In this algorithm, the similarity function relies on distance measurements to compute the similarity between training members [40]. The value of k is adjusted during the training session to assign each instance during training to the correct class. The k-nearest classifier is very sensitive to data size and dimensionality, and this affects the feature space and homogeneous areas, which represent the distribution of various classes [41].

4) MULTILAYER PERCEPTRON

The multilayer perceptron is a particular type of neural network-based classifier [42], [43]. This classifier employs a multilayer feed-forward neural network with one or more layers of nodes between the inputs and output layers. These nodes at different layers are interconnected through weighted networks. Using different training algorithms, the parameters (weights) of the networks are optimized. In this classifier, the data are transferred from input to output. Each feature is used as an input in the multilayer perceptron, and the outputs are the class categories. The multilayer perceptron may be linear, when it is used with a single layer of nodes. It can also be a nonlinear perceptron, when it is applied using multiple layers of nodes with several hidden layers [40].

B. PERFORMANCE EVALUATION AND VALIDATION

In this section, the performance of the CS_DDoS system is evaluated and validated using classification performance measurements and K-fold cross-validation.

1) PERFORMANCE EVALUATION

In this section, the performance of the CS_DDoS method is evaluated using the four classifiers of the LS-SVM, naïve Bayes, k-nearest, and multilayer perceptron. Various training data sizes (window sizes) and thresholds are used in the experiments. Algorithm 1 is applied to the training data for all the classifiers.

The CS_DDoS system was evaluated in terms of accuracy, sensitivity (detection rate) and specificity (false alarm

TABLE 5. Classification performance average.

	Classifiers	Average			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	97%	97%	97%	0.8875
2	Naïve Bayes	88%	94%	94%	0.765
3	K-nearest	81%	96%	95%	0.7275
4	Multilayer perceptron	93%	98%	97%	0.69

TABLE 6. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	98%	93%	94%	0.91
2	Naïve Bayes	82%	91.3%	91%	0.82
3	K-nearest	80%	91.5%	92%	0.74
4	Multilayer perceptron	83.3%	92.3%	95%	0.78

TABLE 7. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	93%	91%	94%	0.91
2	Naïve Bayes	85%	92%	95%	0.81
3	K-nearest	79%	96%	92%	0.82
4	Multilayer perceptron	87%	95%	96%	0.71

TABLE 8. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94%	97%	95%	0.92
2	Naïve Bayes	95%	92%	94%	0.85
3	K-nearest	88%	90%	92%	0.69
4	Multilayer perceptron	89%	97%	93%	0.81

TABLE 9. Classification performance measurements (n=6000 and K=400).

	Classifiers	Detection results			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	92%	97%	94%	0.87
2	Naïve Bayes	91%	93%	95%	0.65
3	K-nearest	87%	91%	96%	0.69
4	Multilayer perceptron	94%	97%	94%	0.60

TABLE 10. Classification performance average.

	Classifiers	Average			
		Accuracy	Sensitivity	Specificity	Kappa coefficient
1	LS-SVM	94%	95%	94%	0.9025
2	Naïve Bayes	88%	92%	94%	0.7825
3	K-nearest	84%	92%	93%	0.735
4	Multilayer perceptron	88%	95%	95%	0.725

rate), as well as the descriptive statistic Kappa coefficient. Kappa coefficients are procedures used to connect between categorical variables, and are frequently used as consistency or legitimacy coefficients [44].

The accuracy represents the rate of correctly identified results over the entire data used by the CS_DDoS. or true negatives (TN), while incorrectly identified results are false positives (FP) and false negatives (FN). The accuracy of the CS_DDoS system is measured by Equation (1).

- True positives (TP): correctly identified abnormal packets in this research.
- False positives (FP): incorrectly identified abnormal packets.

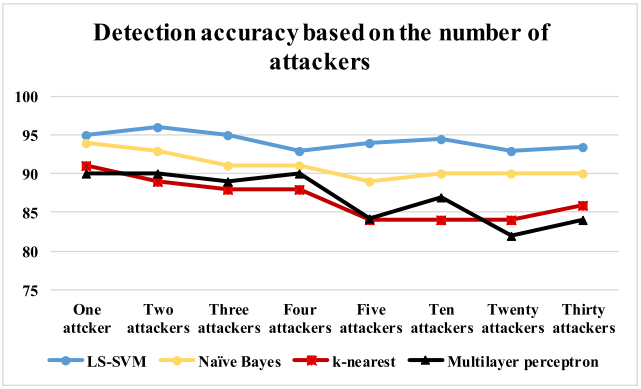


FIGURE 8. Detection accuracy for multiple attacks.

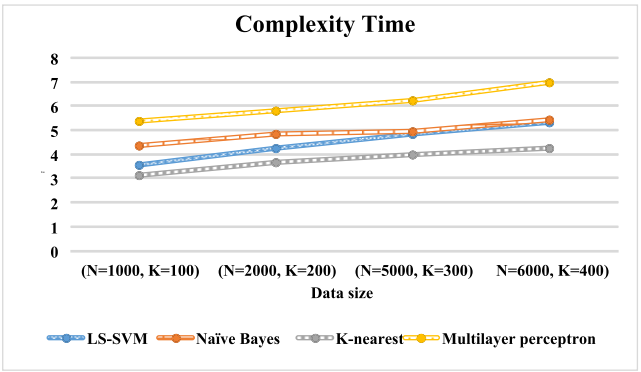


FIGURE 9. Complexity times.

- True negatives (TN): correctly identified normal packets.
- False negatives (FN): incorrectly identified normal packets.

$$CS\_DDoS_{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \times 100\% \quad (1)$$

The sensitivity represents the rate of correctly identified abnormal packets over the entire range of positive results obtained by the CS_DDoS. The sensitivity of the CS_DDoS system is measured by Equation (2).

$$CS\_DDoS_{Sensitivity} = \frac{TP}{TP + FN} \times 100\% \quad (2)$$

The specificity represents the rate of incorrectly identified abnormal packets over the entire range of negative results produced by the CS_DDoS.

The specificity of the CS_DDoS system is measured by Equation (3).

$$CS\_DDoS_{Specificity} = \frac{FP}{FP + TN} \times 100\% \quad (3)$$

The proposed CS_DDoS system is evaluated under both single source and multiple source attack environments, as described below.

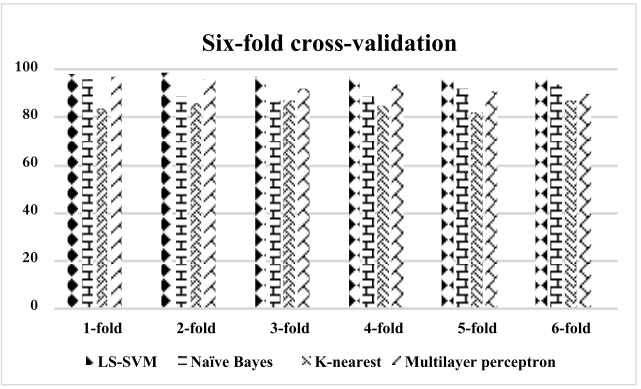


FIGURE 10. Six-fold cross-validation diagram.

a: EVALUATION UNDER SINGLE SOURCE ATTACK

Four test data sizes (n) of 1000, 2000, 5000 and 6000 packets were randomly selected, and four thresholds (K) of 100, 200, 300 and 400 requests. Algorithm 1 was applied to the data according to the window size, n, and was tested according to the threshold K. We have two features fed to each classifier; these two features are the source IP address and the destination IP address. Each classifier was used to classify the data using the four windows and four thresholds. The results are shown in Tables 1-5:

Tables 1 to 4 show the classification performances of the proposed CS_DDoS system with different data sizes and thresholds. The performance measurements are accuracy (correctly detected data over the entire dataset), sensitivity (correctly detected attacks, detection rate), specificity (incorrectly detected attacks, false alarm rate), and Kappa coefficient (stability rate).

According to Tables 1 to 4, the results of each classifier were not significantly affected by the window sizes and thresholds, since there are only small differences between the tables. Tables 1 to 4 are summarized in Table 5.

From Table 5, it can be seen that the LS-SVM classifier has the highest average percentage accuracy (97%) and the highest Kappa coefficient (0.89). Conversely, the k-nearest classifier achieved the lowest accuracy percentage of about 81%, and the multilayer perceptron classifier had the lowest Kappa coefficient 0.69. Overall, the proposed CS_DDoS system is more effective and stable in resisting a single-source attack when adopting the LS-SVM classifier regardless of the window size and threshold.

b: EVALUATION UNDER MULTIPLE-SOURCE ATTACKS

To evaluate the performance under attacks from multiple sources, the same four window sizes were used (1000, 2000, 5000 and 6000) and the same four thresholds (100, 200, 300 and 400). Algorithm 1 was also used. The results are shown in Tables 6–10: Tables 6–9 show the results of the classification accuracy of the proposed CS_DDoS system when under multiple DDoS attacks. Tables 6–9 also show that the results of each classifier were not significantly affected

TABLE 11. Sequence diagram generation codes implemented using quick sequence diagram editor 4.2.

First scenario	Second scenario	Third scenario
Admin:Actor Client:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listening to port() Admin:p.Listening to port() +1 Client First scenario { +1 (1) Client:d.Request data() d:IP address not found=b.Checking IP address() d:Normal packets detected=c.Packet stream c:Process=c.Classify c:stop d:cs.Clean packet (2)cs:Client.Send data +2 d First scenario } +2	Admin:Actor Attacker:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listening to port() Admin:p.Listening to port() +3 Attacker Second scenario { +3 (3)Attacker:d.Request data() d:IP address not found=b.Checking IP address() d:Abnormal packets detected=c.Packets stream() c:Process=c.Classify c:stop d:p.Send abnormal packets() p:Packet termination=p.Process p:b.Add IP address to blacklist() d:Attacker.Request rejected p:cs.Warrning message() cs:Attacker.Warrning message (4)p:Admin.Send info & alarm +4 b Second scenario } +4	Admin:Actor Attacker:Actor d:Detection system b:Blacklist p:Prevention system c:Classifier cs:Cloud service Admin:d.Listening to port() Admin:p.Listening to port() +5 Attacker Third scenario { +5 (5)Attacker:d.Request data() d:IP address found=b.Checking IP address() d:Terminating IP address=p.Attacker IP address found() d:Attacker.Request rejected d:cs.Warrning message() cs:Attacker.Warning message (6)d:Admin.Send info & alarm +6 Admin Third scenario } +6

by the window size and the threshold. LS-SVM was again the best performing classifier with percentage accuracy of around 94%, and a Kappa coefficient of about 0.9. Tables 6-9 are summarized in Table 10.

Overall, the proposed CS_DDoS system is also effective and stable in resisting both multiple-source and single-source attacks when using the LS-SVM classifier, regardless of the window size and threshold. Therefore, the proposed CS_DDoS system can be implemented in a large-scale cloud project, such as a health cloud, as well as in smaller projects such as a private cloud for a medium-sized company. CS_DDoS can prevent DDoS attacks with a 94% accuracy and is highly stable (Kappa coefficient 0.9). CS_DDoS outperforms previous approaches, since either the percentage accuracy of previous approaches is lower than those achieved by CS_DDoS, for example 91% in [45], or are without Kappa coefficient stability measurements for example in [46].

In addition, the false alarm rate (specificity) of the benchmark algorithms are 69.57% on average [47]. Thus, we can claim that our proposed CS_DDoS system is more effective.

To shed more light on the performance evaluation of the proposed CS_DDoS system, the simulation was repeated with various numbers of attackers (source IP) under similar conditions and the performance measurements were calculated.

Figure 8 shows the performance of CS_DDoS with an increasing number of attackers. There are slight fluctuations in the performance measurements of all four classifiers, although LS-SVM was still the best performer of the four.

In addition, the process complexity times of the four classification algorithms is shown in Figure 9. While LS-SVM is only the second least time-consuming, the fastest classifier, k-nearest, has lower performance measurements and a smaller Kappa coefficient compared to LS-SVM. It can therefore be considered that the LS-SVM is the most efficient and effective classifier for use in the CS_DDoS system to resist DDoS TCP flood attacks.

2) K-FOLD CROSS-VALIDATION

K-fold cross-validation is a validation model for measuring how the outcomes of a numerical examination will simplify to an independent dataset. Generally, it is utilized to validate the estimation of performance accuracy in practice for a predictive model [48-51].

K-fold cross-validation was used to carry out a performance comparison of the four predictive modeling algorithms used in CS_DDoS: LS-SVM, naïve Bayes, k-nearest, and multilayer perceptron. These four algorithms were compared in terms of their prediction results.

The dataset was divided into six equal-sized chunks, k=6. As a validation for model testing, one of the six chunks was retained, and the remainder (five chunks) were used as training data. Then, the process of the six-cross model was repeated six times, so that each of the six chunks were used as validation data for each model. The results are shown in

Figure 10. We can see that the values of all folds are almost the same, which means that each fold has approximately the same rate for each of the four classification algorithms. Thus, we can claim that the classification results are stable and accurate, since each algorithm gives almost the same results for each fold.

VI. CONCLUSION

The use of cloud computing in many sectors is becoming widespread, as this helps to improve the system in many respects. However, this cloud project is vulnerable to certain types of attacks, such as DDoS TCP flood attacks. Therefore, we propose a new approach called CS_DDoS for the detection and prevention of DDoS TCP flood attacks. The system is based on classification to ensure the security and availability of stored data, especially important for eHealth records for emergency cases. In this approach, the incoming packets are classified to determine the behavior of the source within a time frame, in order to discover whether the sources are associated with a genuine client or an attacker. The results show that using LS-SVM the CS_DDoS system can identify the attacks accurately. The system has an accuracy of about 97 percent with a Kappa coefficient of about 0.89 when under single attack; it is 94 percent accurate with a Kappa coefficient of about 0.9 when under multiple attacks. The performance is validated using K-fold validation and is shown to be stable and accurate. Thus, the proposed approach can efficiently improve the security of records, reduce bandwidth consumption and mitigate the exhaustion of resources. In the future, we aim to extend CS_DDoS to overcome the problem of DDoS using spoofed IP addresses as well as to improve the proposed work to identify the attackers even when they satisfy the threshold value.

APPENDIX

CONFLICTS OF INTEREST

The authors declare that there are no conflicts of interest regarding the publication of this paper.

REFERENCES

[1] A. Girma, K. Abayomi, and M. Garuba, "The design, data flow architecture, and methodologies for a newly researched comprehensive hybrid model for the detection of DDoS attacks on cloud computing environment," in *Proc. Inf. Technol. Generat. 13th Int. Conf. Inf. Technol.*, 2016, pp. 377–387.

[2] B. Cashell, W. D. Jackson, M. Jickling, and B. Webel, *The Economic Impact of Cyber-Attacks*, document CRS RL32331, Congressional Research Service Documents, Washington, DC, USA, 2004.

[3] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, 1st Quart., 2016.

[4] P. A. Laplante, J. Zhang, and J. Voas, "What's in a name? Distinguishing between SaaS and SOA," *IT Prof.*, vol. 10, no. 3, pp. 46–50, May 2008.

[5] C. Balding. (2012). *What Everyone Ought to Know About Cloud Security*. [Online]. Available: http://www.slideshare.net/craigbalding/what-everyone-ought-to-know-about-cloud-security

[6] I. M. Khalil, A. Khreishah, and M. Azeem, "Cloud computing security: A survey," *Computers*, vol. 3, pp. 1–35, 2014.

[7] B. Wang, Y. Zheng, W. Lou, and Y. T. Hou, "DDoS attack protection in the era of cloud computing and software-defined networking," *Comput. Netw.*, vol. 81, pp. 308–319, Apr. 2015.

[8] M. Xia, W. Lu, J. Yang, Y. Ma, W. Yao, and Z. Zheng, "A hybrid method based on extreme learning machine and k-nearest neighbor for cloud classification of ground-based visible cloud image," *Neurocomputing*, vol. 160, pp. 238–249, Jul. 2015.

[9] A. Taravat, F. D. Frate, C. Cornaro, and S. Vergari, "Neural networks and support vector machine algorithms for automatic cloud classification of whole-sky ground-based images," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 3, pp. 666–670, Mar. 2015.

[10] A. Sahi, D. Lai, and Y. Li, "Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan," *Comput. Biol. Med.*, vol. 78, pp. 1–8, Nov. 2016.

[11] A. S. Khader and D. Lai, "Preventing man-in-the-middle attack in Diffie–Hellman key exchange protocol," in *Proc. 22nd Int. Conf. Telecommun. (ICT)*, 2015, pp. 204–208.

[12] A. Sahi, D. Lai, and Y. Li, "Parallel encryption mode for probabilistic scheme to secure data in the cloud," in *Proc. 10th Int. Conf. Inf. Technol. Appl. (ICITA)*, Sydney, NSW, Australia, 2015.

[13] A. A. Hameed, B. Karlik, and M. S. Salman, "Back-propagation algorithm with variable adaptive momentum," *Knowl.-Based Syst.*, vol. 114, pp. 79–87, Dec. 2016.

[14] U. R. Acharya *et al.*, "Automated characterization and classification of coronary artery disease and myocardial infarction by decomposition of ECG signals: A comparative study," *Inf. Sci.*, vol. 377, pp. 17–29, Jan. 2017.

[15] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 39–53, 2004.

[16] W. Wei, F. Chen, Y. Xia, and G. Jin, "A rank correlation based detection against distributed reflection DoS attacks," *IEEE Commun. Lett.*, vol. 17, no. 1, pp. 173–175, Jan. 2013.

[17] P. E. Ayres, H. Sun, H. J. Chao, and W. C. Lau, "ALPi: A DDoS defense system for high-speed networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 10, pp. 1864–1876, Oct. 2006.

[18] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: An architecture for mitigating DDoS attacks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 1, pp. 176–188, Jan. 2004.

[19] X. Wang and M. K. Reiter, "Using Web-referral architectures to mitigate denial-of-service threats," *IEEE Trans. Depend. Sec. Comput.*, vol. 7, no. 2, pp. 203–216, Apr. 2010.

[20] Y. Xuan, I. Shin, M. T. Thai, and T. Znati, "Detecting application denial-of-service attacks: A group-testing-based approach," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 8, pp. 1203–1216, Aug. 2010.

[21] K. Salah, K. Elbadawi, and R. Boutaba, "Performance modeling and analysis of network firewalls," *IEEE Trans. Netw. Service Manage.*, vol. 9, no. 1, pp. 12–21, Mar. 2012.

[22] W. Dou, Q. Chen, and J. Chen, "A confidence-based filtering method for DDoS attack defense in cloud environment," *Future Generat. Comput. Syst.*, vol. 29, no. 7, pp. 1838–1850, 2013.

[23] J. Yu, H. Lee, M.-S. Kim, and D. Park, "Traffic flooding attack detection with SNMP MIB using SVM," *Comput. Commun.*, vol. 31, pp. 4212–4219, Nov. 2008.

[24] K. Lee, J. Kim, K. H. Kwon, Y. Han, and S. Kim, "DDoS attack detection method using cluster analysis," *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1659–1665, 2008.

[25] A. Chonka, Y. Xiang, W. Zhou, and A. Bonti, "Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks," *J. Netw. Comput. Appl.*, vol. 34, no. 4, pp. 1097–1107, 2011.

[26] K. Lu, D. Wu, J. Fan, S. Todorovic, and A. Nucci, "Robust and efficient detection of DDoS attacks for large-scale Internet," *Comput. Netw.*, vol. 51, no. 18, pp. 5036–5056, 2007.

[27] S. Khanna, S. S. Venkatesh, O. Fatemieh, F. Khan, and C. A. Gunter, "Adaptive selective verification: An efficient adaptive countermeasure to thwart DoS attacks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 715–728, Jun. 2012.

[28] R. Guo, H. Yin, D. Wang, and B. Zhang, "Research on the active DDoS filtering algorithm based on IP flow," *Int. J. Commun., Netw. Syst. Sci.*, vol. 7, pp. 600–607, Sep. 2009.

[29] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in *Proc. IEEE 35th Conf. Local Comput. Netw. (LCN)*, Oct. 2010, pp. 408–415.

[30] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection," *Pattern Recognit. Lett.*, vol. 51, pp. 1–7, Jan. 2015.

[31] G. Somani, M. S. Gaur, D. Sanghi, and M. Conti, "DDoS attacks in cloud computing: Collateral damage to non-targets," *Comput. Netw.*, vol. 109, pp. 157–171, Nov. 2016.
[32] E. Al-Shaer and S. F. Gillani, *Agile Virtual Infrastructure For Cyber Deception Against Stealthy DDoS Attacks in Cyber Deception: Building the Scientific Foundation*, S. Jajodia, V. S. Subrahmanian, V. Swarup, and C. Wang, Eds., Cham, Switzerland: Springer, 2016, pp. 233–257.
[33] S. Kumar and O. Gomez, "Denial of service due to direct and indirect ARP storm attacks in LAN environment," *J. Inf. Secur.*, vol. 1, pp. 88–94, Jan. 2010.
[34] (Jun. 7, 2016). *Wireshark Analyzer 2.0.0*, [Online]. Available: https://www.wireshark.org/
[35] M. Albanese, E. Battista, and S. Jajodia, *Deceiving Attackers by Creating a Virtual Attack Surface in Cyber Deception: Building the Scientific Foundation*, S. Jajodia, V. S. Subrahmanian, V. Swarup, and C. Wang, Eds., Cham, Switzerland: Springer, 2016, pp. 167–199.
[36] P. Wang, H.-T. Lin, and T.-S. Wang, "An improved ant colony system algorithm for solving the IP traceback problem," *Inf. Sci.*, vol. 326, pp. 172–187, Jan. 2016.
[37] (Jul. 20, 2016). *Quick Sequence Diagram Editor 4.2*, [Online]. Available: https://github.com/sdedit/sdedit
[38] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, Jun. 1999.
[39] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Conf. Uncertainty Artif. Intell.*, 1995, pp. 338–345.
[40] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Hoboken, NJ, USA: Wiley, 1973.
[41] A. Depeursinge *et al.*, "Comparative performance analysis of state-of-the-art classification algorithms applied to lung tissue categorization," *J. Digit. Imag.*, vol. 23, no. 1, pp. 18–30, 2010.
[42] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, vol. 4, no. 2, pp. 4–22, Apr. 1987.
[43] R. K. Madyastha and B. Aazhang, "An algorithm for training multilayer perceptrons for data classification and function interpolation," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 41, no. 12, pp. 866–875, Dec. 1994.
[44] H. C. Kraemer, V. S. Periyakoil, and A. Noda, "Kappa coefficients in medical research," *Statist. Med.*, vol. 21, no. 14, pp. 2109–2129, 2002.
[45] M. Moradi and M. Zulkernine, "A neural network based system for intrusion detection and classification of attacks," in *Proc. IEEE Int. Conf. Adv. Intell. Syst.-Theory Appl.*, Nov. 2004, pp. 1–6.
[46] R. Jalili, F. Imani-Mehr, M. Amini, and H. R. Shahriari, "Detection of distributed denial of service attacks using statistical pre-processor and unsupervised neural networks," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.*, 2005, pp. 192–203.
[47] A. Kuzmanovic and E. W. Knightly, "Low-rate TCP-targeted denial of service attacks: The shrew vs. the mice and elephants," in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2003, pp. 75–86.
[48] G. McLachlan, K.-A. Do, and C. Ambroise, *Analyzing Microarray Gene Expression Data*, vol. 422. Hoboken, NJ, USA: Wiley, 2005.
[49] D. M. Allen, "The relationship between variable selection and data agumentation and a method for prediction," *Technometrics*, vol. 16, no. 1, pp. 125–127, 1974.
[50] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *J. Roy. Statist. Soc. B (Methodol.)*, vol. 36, no. 2, pp. 111–147, 1974.
[51] S. Geisser, "The predictive sample reuse method with applications," *J. Amer. Statist. Assoc.*, vol. 70, pp. 320–328, Jun. 1975.

AQEEL SAHI received the bachelor's degree in computer science from Thi-Qar University, Iraq, in 2007, and the master's degree in information technology from University Utara Malaysia, Malaysia, in 2010. He is currently pursuing the Ph.D. degree with the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD, Australia. His current research interests are in cryptography and parallel processing with a focus on block cipher modes of operation and key exchange protocols.

DAVID LAI received the B.Sc., PGDipEd, and M.Phil. degrees from The Chinese University of Hong Kong, the GDipCompSc degree from the Vaal University of Technology, the MIT degree from the Queensland University of Technology, and the Ph.D. degree from the University of Southern Queensland.

He is currently a Senior Lecturer with the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD, Australia.

YAN LI received the B.Eng. and M.Eng. degrees from the Huazhong University of Science and Technology, and the Ph.D. degree from Flinders University.

She is currently an Associate Professor with the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD, Australia.

She is also a Approved Research Supervisor in the area of signal processing (090609), computer communications, networks (100503), fields of research, biomedical engineering, artificial intelligence, image processing, signal processing, and computer communications networks.

Her research interests are machine learning algorithms, big data analytics, signal/image processing, EEG research, graph theory, and networking technologies.

MOHAMMED DIYKH received the B.Sc. degree in computer science from Thi-Qar University, Iraq, in 2003, and the M.Sc. degree in information technology from Voronezh State University, Russia, in 2010. He is currently pursuing the Ph.D. degree with the Faculty of Health, Engineering and Sciences, University of Southern Queensland, Australia. His current research interests include biomedical signal analysis, data mining, and graph theory.

A review of the state of the art in privacy and security in the eHealth cloud

Aqeel Sahi, David Lai, and Yan Li

Abstract

The widespread proliferation and usefulness of cloud computing within eHealth requires high levels of security and privacy for health records. However, eHealth clouds pose serious security and privacy concerns for sensitive health data. Therefore, practical and effective methods of resistance are essential in order to preserve the privacy and security of this data. In view of this, and to fully explore current research directions in security and privacy in eHealth cloud, this study summarizes and analyses the state of the art in security and privacy in the eHealth cloud. An extensive review is conducted, and over 100 studies from several peer-reviewed databases such as IEEE Xplore are examined. The selected studies are reviewed and summarized in terms of their benefits and risks. Thus, eHealth stakeholders can benefit from the knowledge and information presented in this paper. In addition, this study compares several research works in terms of data security requirements. This paper can assist in an understanding of the research trends in the areas of privacy and security.

Key words Cloud security; cloud privacy; eHealth cloud.

Aqeel Sahi
Department of Math and Computing, Faculty of Health, Engineering and Sciences, University of Southern Queensland, Australia & Computer Center, University of Thi-Qar, Iraq
Present address:
487/521-535 West St, Darling Heights QLD 4350, Australia
e-mail: akeel_sahy@yahoo.co.uk

David Lai
Department of Math and Computing, Faculty of Health, Engineering and Sciences, University of Southern Queensland, Australia
487/521-535 West St, Darling Heights QLD 4350, Australia
e-mail: David.Lai@usq.edu.au

Yan Li
Department of Math and Computing, Faculty of Health, Engineering and Sciences, University of Southern Queensland, Australia
487/521-535 West St, Darling Heights QLD 4350, Australia
e-mail: Yan.Li@usq.edu.au

- The number of words of the abstract: 168
- The number of figures: 9
- The number of tables: 1

Authors biography

Aqeel Sahi is a Ph.D. student in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia. He received a Bachelor degree of Computer Science from Thiqar University, Iraq in 2007, and Master degree of Information Technology from University Utara Malaysia, Malaysia in 2010. His current research interests are in cryptography and eHealth cloud security and privacy.

David Lai is a **s**enior lecturer in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia.
Qualifications BSc CUHK , PGDipEd CUHK , GDipCompSc VUT , MPhil CUHK , MIT QUT , PhD USQ.

Yan Li is a full professor in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia.
Qualifications BEng HUST , MEng HUST , PhD Flinders.
Approved research supervisor in the area of : Signal Processing (090609), Computer Communications, Networks (100503), Fields of Research (FoR), Biomedical Engineering, Artificial Intelligence, Image Processing, Signal Processing and Computer Communications Networks.
Research interests: Machine Learning Algorithms, Big Data Analytics, Signal/Image Processing, EEG Research, Graph Theory, and Networking Technologies.

A review of the state of the art in privacy and security in the eHealth cloud

Abstract

The widespread proliferation and usefulness of cloud computing within eHealth requires high levels of security and privacy for health records. However, eHealth clouds pose serious security and privacy concerns for sensitive health data. Therefore, practical and effective methods of resistance are essential in order to preserve the privacy and security of this data. In view of this, and to fully explore current research directions in security and privacy in eHealth cloud, this study summarizes and analyses the state of the art in security and privacy in the eHealth cloud. An extensive review is conducted, and over 100 studies from several peer-reviewed databases such as IEEE Xplore are examined. The selected studies are reviewed and summarized in terms of their benefits and risks. Thus, eHealth stakeholders can benefit from the knowledge and information presented in this paper. In addition, this study compares several research works in terms of data security requirements. This paper can assist in an understanding of the research trends in the areas of privacy and security.

Key words Cloud security; cloud privacy; eHealth cloud.

1. Introduction

The official definition of cloud computing, according to the National Institute of Standards and Technology (NIST) is as follows: "cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" (Mell & Grance, 2009). Over the last decade, cloud computing has gained popularity within the health sector, as it offers several advantages such as low costs and flexible processes (Li et al., 2013). Cloud-based health services allow physicians, patients, and owners of health data to control and share their data easily. However, eHealth cloud computing poses a range of challenges, such as data security and privacy for clients and cloud service providers (CSPs) (Yu et al., 2017). Security and privacy issues threaten an open network and semi-trusted servers, which may lose, leak or disclose data (Tang et al., 2016). This can breach users' privacy when sharing data in a public cloud.

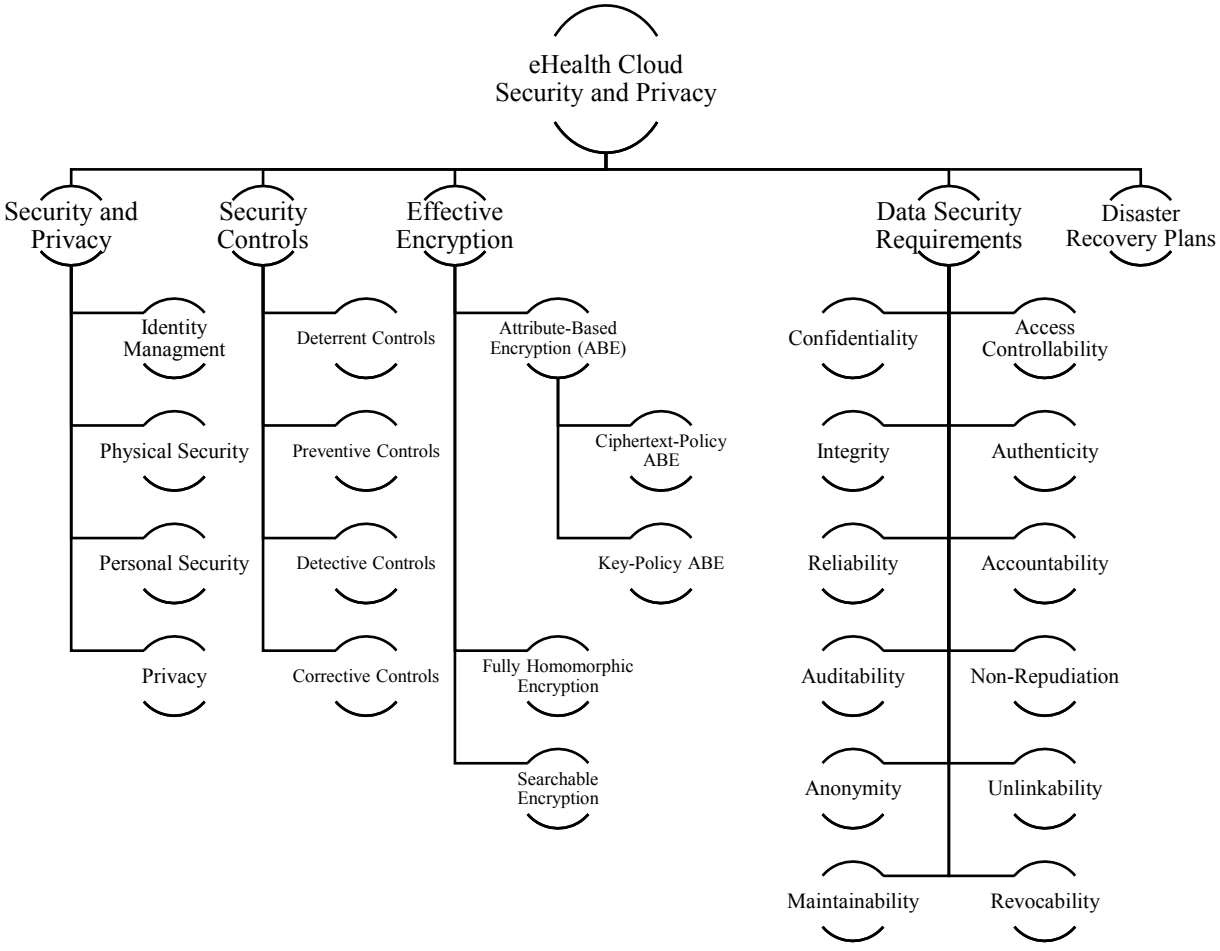


Fig. 1. Security and privacy taxonomy for the eHealth cloud

A great deal of research has explored the security and privacy issues in the eHealth cloud, and many solutions have been suggested to resolve these issues. To obtain a clear picture of the security and privacy problems that can affect the eHealth cloud, this study summarizes and analyses the current state of the art in eHealth security and privacy. The aim of this study is to deliver a clear and complete picture of eHealth security and privacy issues, and to discuss recent research that targets these issues and the proposed solutions. As shown in Figure 1, we divide our study into five main categories: security and privacy, security controls, effective encryption, data security requirements, and disaster recovery plans.

Although though cloud computing is widely used in the health sector, numerous issues remain unresolved (Jing et al., 2013; Sahi et al., 2015). Several studies have attempted to review the state of the art in security and privacy in the eHealth cloud (Abbas & Khan, 2014; Fernandez-Aleman et al., 2013; Gonzalez-Martinez et al., 2015; Sajid & Abbas, 2016; Yuksel et al., 2017); however, some of these studies are now rather outdated, and others do not cover certain vital features of cloud security and privacy, such as access control, revocation and data recovery plans. In addition, some of the existing review papers focus either on the privacy of the cloud or the security of the cloud, but not both. We therefore classify and evaluate these review papers of eHealth security and privacy. We believe that our review covers most of the recent studies in this area, and that it can be considered a good literature base for researchers in the area of health clouds.

1.1. Paper Selection

For this research, papers were collected from well-known research databases including IEEE Xplore, Springer, Elsevier, Science Direct, and Google Scholar. These databases contain many studies from journals and conferences which are relevant to security and privacy in the eHealth cloud. The primary selection was limited to the five years between 2013 and 2017, with certain exceptions. We used search terms such as "*eHealth cloud security and privacy*", "*eHealth cloud access control*", "*eHealth cloud encryption*", "*eHealth cloud security requirements*", and "*eHealth cloud recovery plans*". The function words AND, OR, and NOT were also used to perform advanced searches, such as "*eHealth cloud revocation*" AND ("integrity" OR "access control"). Finally, we reviewed the selected papers according to their titles, abstracts, keywords, and conclusions to include the most relevant papers and to exclude irrelevant ones from the study. Figure 2 shows the inclusion and exclusion processes, and Figure 3 shows the distribution of the selected articles over the years.

The contributions of this study are as follows: we collect and evaluate papers on the current state of the art in eHealth security and privacy schemes. The collected papers are classified into five categories, as shown in Figure 1. We discuss the proposals and issues covered in these selected papers to facilitate better security and privacy in eHealth clouds.

The remainder of this paper is organized as follows, using a structure similar to that of Figure 1. Section 2 describes the proposed schemes with regard to the security and privacy of the eHealth cloud; Section 3 describes the proposed schemes with regard to security controls; Section 4 describes effective encryption of the eHealth cloud; Section 5 discusses the data security requirements of the eHealth cloud; Section 6 describes disaster recovery plans; and finally, Section 7 concludes the study.

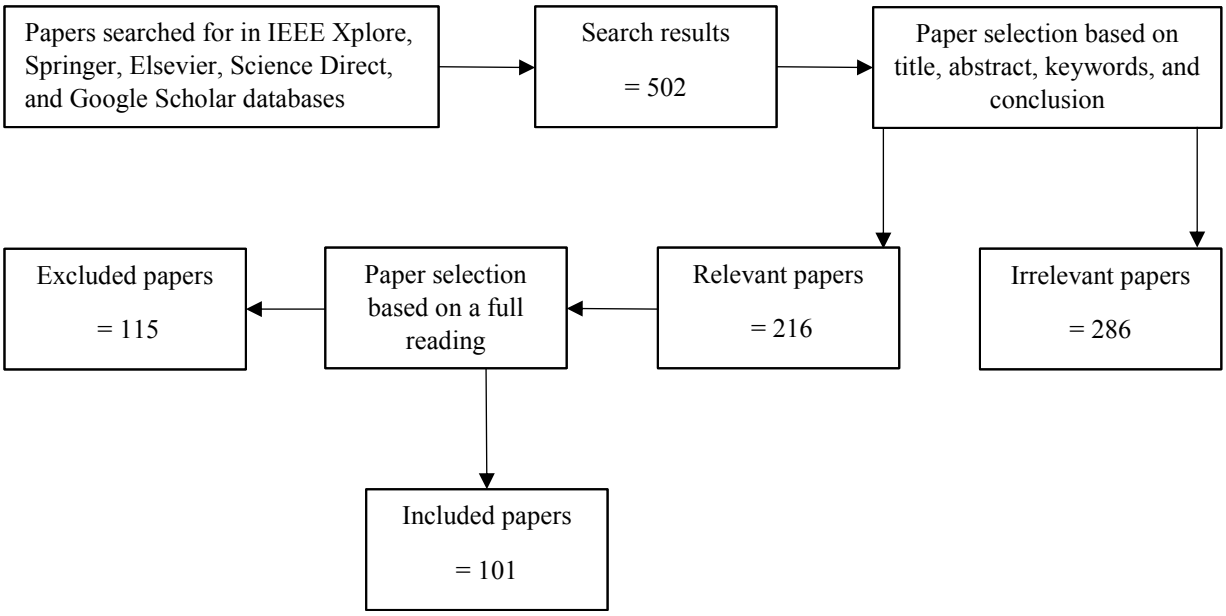


Fig. 2. The inclusion/exclusion process

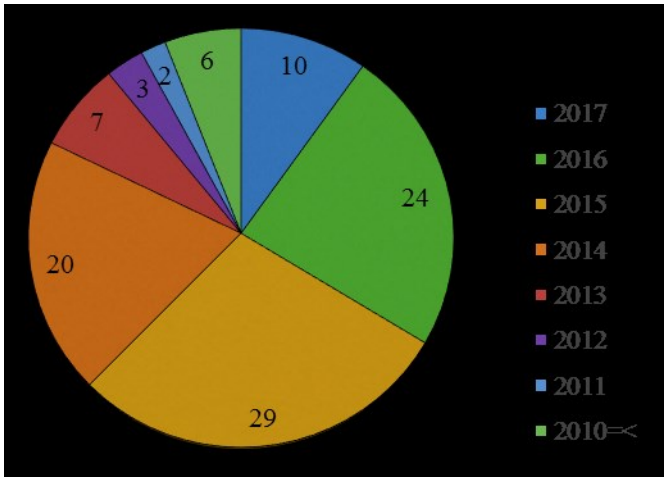


Fig. 3. Distribution of the selected articles by year

2. Security and Privacy in the Cloud

Cloud computing is a model commonly used to save money and effort in many sectors, and particularly in the health sector. However, despite the benefits of the eHealth cloud, there are still many unresolved issues regarding security and privacy which require a great deal of research to be resolved.

2.1. Identity Management

An Identity Management System (IMS) is a comprehensive organizational system used to identify entities in a cloud project. Access to information and resources in the project is managed by linking client privileges and constraints with a proven identity. The main aim of an IMS is to determine what clients can do within a cloud project and under what conditions. In addition, an IMS is utilized to improve the security and privacy of a cloud system, and to reduce the running costs and effort.

In order to manage access to data and resources, cloud service providers (CSP) either use their own IMS (such as CloudID (Haghighat et al., 2015)), or incorporate the client's IMS into their infrastructure, for example using a biometric-based IMS to preserve the privacy of the cloud project's information (Haghighat et al., 2015). A biometric-based IMS is used to connect the private data of the clients to their biometrics, which are saved as ciphertexts. To ensure that the CSP or any possible attackers cannot obtain any type of access to this private information, the proposed biometric-based IMS is implemented in an encrypted domain using a searchable cryptographic system.

Recently, Wang et al. proposed a cost-effective secure eHealth cloud system using the Identity Based Encryption (IBE) method (Wang et al., 2017). In this system, there are four parties with different roles: the cloud, the health community, physicians, and patients. The system works as follows. Firstly, the system sets up public and private keys for all parties, according to their published identities (e.g. email addresses). These identities are considered as public keys, and are used to generate private keys using an IBE algorithm. Next, the Electronic Health Records (EHRs) are encrypted by the parties using a block cipher algorithm such as AES, and the keys are encrypted using the IBE and sent to the cloud. Following this, the parties can receive the encrypted EHRs from the cloud, and decrypt them using their identity keys.

According to a survey conducted in (Khalil et al., 2014), more than 66% of users' identities are stored in unsafe places. Khalil et al. (Khalil et al., 2014) therefore proposed a new IMS system called the Consolidated Identity Management (CIDM) system, which they claim is resistant to certain attacks, such as server compromise attacks, mobile device compromise attacks, and traffic interception attacks (Khalil et al., 2014). The CIDM mechanism splits permission identifications and spreads them between the parties at the IMS to prevent traffic interception attacks. In order to mitigate mobile device compromise attacks, a challenge-response approach was adopted. Finally, the security of the communication channels between the CIDM and the CSP is addressed to reduce the possibility of any effective compromise of that channel (Khalil et al., 2014). However, further investigation is required to resolve the problem of insufficient dynamic federated identities and privacy in most of the current IMS systems (Sanchez et al., 2012). This is an architectural problem and must be considered at the design level.

Haufe et al. proposed a new framework named the Information Security Management System (ISMS) (Haufe et al., 2014), consisting of many vital security procedures for the eHealth cloud. The proposed security management framework was implemented based on the ISO 27000 family of standards. The ISMS was able to identify the most frequent cloud computing threats and the information that these aimed to collect from the cloud system (Haufe et al., 2014). One drawback is that the ISMS

needs specific details from processes, such as input, output, and interfaces, to facilitate communication and interaction between processes.

In another study, the concept of the Identity Management as a Service (IDaaS) was discussed (Nunez & Agudo, 2014). In this work, the authors proposed a new IMS called BlindIdM which preserved the privacy of data and delivered them as IDaaS. Specifically, the authors described how a system based on the Security Assertion Markup Language (SMAL) was employed with proxy encryption to enhance the security of the cloud project with respect to the CSP (Nunez & Agudo, 2014). To improve the proposed system, extending the IDaaS from a single domain to a cross-domain approach has been suggested, as in the System for Cross-domain Identity Management (SCIM) (Grizzle et al., 2015; SCIM, 2017).

Xiong et al. proposed a scheme named Privacy pReserving Identity and Access Management (PRIAM) (Xiong et al., 2014), which had five components: registration, token withdrawal, tenant pre-authorization, access control, and token spending. PRIAM is described as being able to fulfil all the requirements of cloud security. The proposed scheme used a hash function, signature, and mutual authentication to ensure the privacy of clients. In order to deliver secure access control for clients and CSP, it utilized a service-level agreement. Finally, Burrows Abadi Needham (BAN) logic was used to confirm the correctness of the scheme (Burrows et al., 1989).

2.2. Physical Security

Physical security is the concept of securing and controlling access to servers, storage, and workstations. In other words, the aim of physical security is to prevent intruders from accessing cloud physical facilities. Cloud hardware, such as servers, switches etc., is also physically secured by the CSP from any unusual activities such as attacks, threats, and floods (Regola & Chawla, 2013), and is provided with the necessary power supplies to reduce any potential interruptions.

Mxoli et al. showed that in order to protect Personal Health Records (PHRs) from any physical intrusion, the system hardware must have a physical security border (Mxoli et al., 2014); for example, physical access control, offices and rooms must be secured, and resistance against disasters and other environmental situations must be available. All of these security borders must be in place to ensure that the cloud and network equipment is not readily accessible to the public. The equipment and applications used by the CSP, which may contain PHRs, must not be moved out of the site or repositioned without the administrator's authorization (Mxoli et al., 2014).

The IT equipment building, or the site where data centres and other cloud hardware are located, must be properly secured. Rodrigues et al. highlighted that these buildings must be secured by security staff members, video surveillance systems and Intrusion Detection Systems (IDS); in addition, only authorized people should be allowed to enter the building using authenticated access controls (Rodrigues et al., 2013).

Carlson stated that CSPs should adopt Federal Information Security Management Act (FISMA) standards in order to ensure the physical security of their records. Since the physical entrances to the physical machines and storage devices are a possible route for data compromise, FISMA must be implemented at the client sites as well as at the server sites (Carlson, 2014).

2.3. Privacy

CSP uses encryption and other techniques to preserve the privacy of clients' critical information, such as credit card numbers, and only authorized clients have the right to access this kind of information.

Abbas et al. reviewed the state of the art in eHealth cloud privacy in 2014 (Abbas & Khan, 2014). Here, we aim to cover not only the issues regarding privacy, but also other security concerns, such as storage security, access controls, and disaster recovery plans etc. In this section, we will therefore first review some of the proposed approaches with regard to eHealth privacy. More details on privacy preservation approaches can be found in Abbas et al.'s paper (Abbas & Khan, 2014).

A three-factor authentication protocol based on Elliptic Curve Cryptography (ECC) was proposed by Yeh et al. in 2013 (Yeh et al., 2013). The protocol had certain disadvantages, such as a vague procedure, impractical IDs, and no shared key (Wu et al., 2015). In addition, the protocol could not prevent spoofing attacks (Wu et al., 2015). Another authentication protocol based on a fingerprint was proposed by Khan et al. (Khan & Kumari, 2013). However, this protocol could not mitigate impersonation or desynchronization attacks (Wu et al., 2015). In order to overcome the weaknesses of these protocols, Wu et al. proposed a new biometrics-based three-factor authentication protocol that can overcome all those drawbacks as well as ensuring the privacy of clients (Wu et al., 2015). This protocol used the ECC and mobile devices, and adopted a fuzzy extractor to deal with inadequate biometric inputs. The protocol proposed by Wu et al. was formally proved using random oracles and Elliptic Curve Gap Diffie–

Hellman (ECGDH) problem assumption to demonstrate the low probability of success of these attacks. However, this protocol is vulnerable to other attacks, such as impersonation and offline password guessing attacks, if the mobile device falls into the wrong hands; in addition, the user revocation procedure was not included in that protocol (Jiang et al., 2016 (A)). Therefore, another three-factor authentication protocol that is able to resist these attacks and offers more security features was proposed by Jiang et al. in 2016 (Jiang et al., 2016 (A)).

Yang et al. presented a privacy preservation approach for health records in the eHealth cloud (Yang et al., 2015). This approach was based on the classification of health record attributes; it collected these attributes vertically from the health dataset in order to ensure that these were collected from all areas of the dataset with different privacy aspects. Their approach consisted of four steps: (1) vertical data collection, (2) data merging, (3) integrity checks, and (4) plain and cipher text searches. Cryptography and statistical analysis were combined in order to create multiple approaches which can strike a balance between the use of health records and privacy preservation (Yang et al., 2015). However, this approach does not consider the situation where several users are using the service at the same time.

Another scheme proposed by Sahi et al. aims to preserve the privacy of the PHRs (Sahi et al., 2016). This scheme adopted a three-party password-based authenticated key exchange protocol (3PAKE) based on the computational Diffie–Hellman assumption proposed by Khader and Lai (Khader & Lai, 2015). This scheme used a different generator and primitive root in each session to ensure that only the specific client has complete access to his/her PHR; clients are revoked at the end of the session. This can ensure that old session keys cannot be used to access a client's PHRs. A disaster recovery plan and a break-glass technique are also addressed in this scheme.

According to Wang et al., cryptography can be very expensive when it is used to preserve the privacy of health records in the cloud (Wang et al., 2015 (B)). They therefore proposed a privacy preserving scheme which transferred sensitive health information to a trusted private cloud and the remainder to a public one. Two protocols were involved in this scheme; the first was used to preserve the privacy of the clients, and the second was used to resist any potential collusion between user records and the public CSP. To ensure the privacy of the sensitive information, the dataset was divided into several parts. The fragmented information was distributed among clouds and could be re-joined (Wang et al., 2015 (B)).

Based on the HireSome-I method, an improved history record-based service optimization method (HireSome-II) was proposed by Dou et al. in 2015 (Dou et al., 2015). HireSome-II was proposed in order to ensure the privacy of big data such as health records in cloud computing. The cloud rejects requests which can reveal transaction information for privacy reasons, and the proposed method can efficiently support the cloud service structure to complete transactions securely (Dou et al., 2015).

Another framework to ensure the privacy of patient data was proposed by Page et al. (Page et al., 2015). This framework combines monitoring and analytic methods to deliver secure and authenticated health records. This framework was based on fully homomorphic encryption (FHE). However, FHE is known as a heavy technique; therefore, to measure the practicality of the proposed framework, the authors developed a proof of concept and a prototype system (Page et al., 2015).

3. Cloud Security Controls

Security approaches are effective in cloud environments when an excellent protection mechanism is adopted. This mechanism must identify the potential problems that may arise during the management process. These problems will be addressed and considered by the security controls, thus preserving the security of the system from its own weaknesses and reducing the number of attacks (Sajid & Abbas, 2016). There are many cloud security controls, which can be categorized as follows.

3.1. Deterrent Controls

Deterrent controls aim to reduce the number of attacks on a cloud project. A "No Trespassing" sign can alert security personnel to watch out for intruders as well as highlighting the consequences of intrusion. Deterrent controls serve to warn attackers that there will be penalties and punishments if they proceed with attacks (Nedelcu et al., 2015; Rajamani et al., 2016).

3.2. Preventive Controls

Preventive controls aim to secure cloud projects by preventing or decreasing vulnerabilities. For example, an effective authentication protocol can ensure the security of the cloud's clients, and prevent any unauthorized access to that cloud. Preventive controls can therefore help the cloud system to confidently identify their clients (Nedelcu et al., 2015; Rajamani et al., 2016).

3.3. Detective Controls

Detective controls aim to detect and respond appropriately to attacks which could threaten the cloud system. During an attack, the detective control will notify the preventive control or the corrective control to report the problem. An intrusion detection system (IDS) is typically used as a detective control (Li et al., 2013; Yu et al., 2017).

3.4. Corrective Controls

Corrective controls aim to reduce the damage of an attack. These controls are usually initiated during or after attacks. Restoring a cloud system from a backup to ensure the availability of services is an example of a corrective control (Nedelcu et al., 2015; Rajamani et al., 2016).

Generally, access controls are linked to security policies delivered to clients while accessing the service. A company typically has its own security controls which allow staff members access to a set of data rather than giving them full data access. This control limits the access of a staff member to a particular group of data. These kinds of security controls need to be put in place in cloud projects to avoid unauthorized access. The Software as a Service (SaaS) model must be sufficiently elastic to combine the set of controls offered by the company (Subashini & Kavitha, 2011).

Recently, much research has been done on cloud security controls. We discuss some of these works in the following paragraphs.

Many stakeholders might access PHRs without authorization. Access control is therefore a major problem for the privacy of data when health records are stored and shared in the cloud. Thus, a dynamic access control is necessary to ensure the privacy of these stored health records. Son et al. proposed a new dynamic access control scheme for securing the privacy of the PHRs in cloud projects (Son et al., 2017). Their scheme can detect unauthorized access dynamically by altering the context information, meaning that even if the subject has the same role, access authorization will not be defined in the same way, according to the conditions and the context information. The proposed scheme was tested using a real-life health system.

Tong et al. proposed another access control architecture which was designed to ensure the privacy of data (Tong et al., 2014). The proposed architecture had several features including key exchange, storage data privacy, emergency retrieval, and auditability to overcome any misuse of health records. A pseudorandom number generator was used as a key exchange to ensure unlinkability, and a redundancy-based secure indexing feature was proposed to preserve the privacy of the data by hiding the search and access patterns. Finally, in order to mitigate any potential misbehaviour, an attribute-based encryption was integrated with threshold signing, to be used in emergency and normal situations as an access control with auditability.

Based on two-stage keyed access control and a zero-knowledge protocol, Kahani et al. proposed another security control method (Kahani et al., 2016). Their method aimed to facilitate access control and authentication in electronic health cloud systems. When a user requests access to a health record, a limited amount of access will be extracted based on the user's rights. In order to connect two parties in the system securely, a two-stage key management is used. This two-stage key management is a combination of public key encryption and Derived Unique Key Per Transaction (DUKPT)

Fernando et al. proposed a new approach which aimed to reduce leaks of patient information using unlinkability (Fernando et al., 2016). This approach provided the health data owner with the ability to make decisions in terms of access control. To fulfil the policies of the service provider, the proposed approach utilized a personal information management protocol which could improve the privacy of the patients. This approach depended on a scenario in which patient EHRs were stored on a Health Information Exchange (HIE) cloud service. The approach demonstrated the communication techniques between EHR consumers, EHR owners, EHR creators, and the HIE service. The authors claimed that the privacy of the EHR was ensured by the unlinkability of consumers' sessions with the HIE service; in addition, the HIE service could not reach the consumer classes even when they had access policies. The proposed approach works as follows: a patient consults a doctor and the doctor prescribes a medical test. The patient goes to a laboratory with the doctor's instructions, and the laboratory carries out the test. The results of the test are sent by the laboratory to the HIE. Finally, the patient provides access to the doctor and the HIE (Fernando et al., 2016). The proposed approach is illustrated in Figure 4.

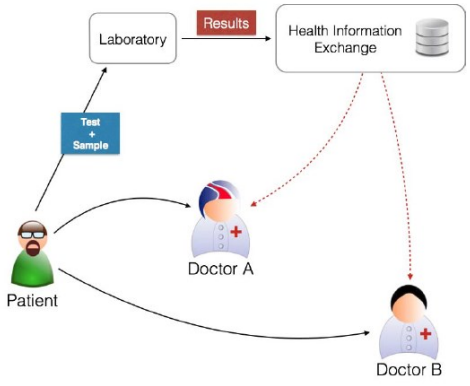


Fig. 4. Approach proposed by Fernando et al. (Fernando et al., 2016)

In 2015, Wand et al. proposed a new scheme called Constant-Size Ciphertext Policy Comparative Attribute-Based Encryption (CCP-CABE). This method inserts similar characteristics from all attributes into a key, and combines the restrictions of these attributes into a single chunk of a ciphertext. This procedure is carried out during the encryption process in order to apply elastic access control rules with a variety of relationships. The authors showed that the CCP-CABE scheme was efficient, as it produced keys and ciphertexts of the same size each time for any number of attributes, as well as reducing the cost of the computation to a trivial size. In order to ensure access privacy, the authors extended CCP-CABE to numerous attribute domains (Wang et al., 2015 (C)).

Younis et al. proposed a new model named Access Control for Cloud Computing (AC3) (Younis et al., 2014). This model utilized the role and task principles, as shown in Figure 5, and used clients' jobs as a categorizing factor. Based on the clients' jobs roles, security domains are created to restrict each client to a particular security domain. Each role within the AC3 is given a group of related and required tasks for performing those roles. For access to data and resources, security classification is done for each task, and an authentic permission is required to complete the task. The authors employed a risk engine to interact with unpredictable client behaviours. However, an authentication protocol that can deal with massive storage complexity and high performance is required.

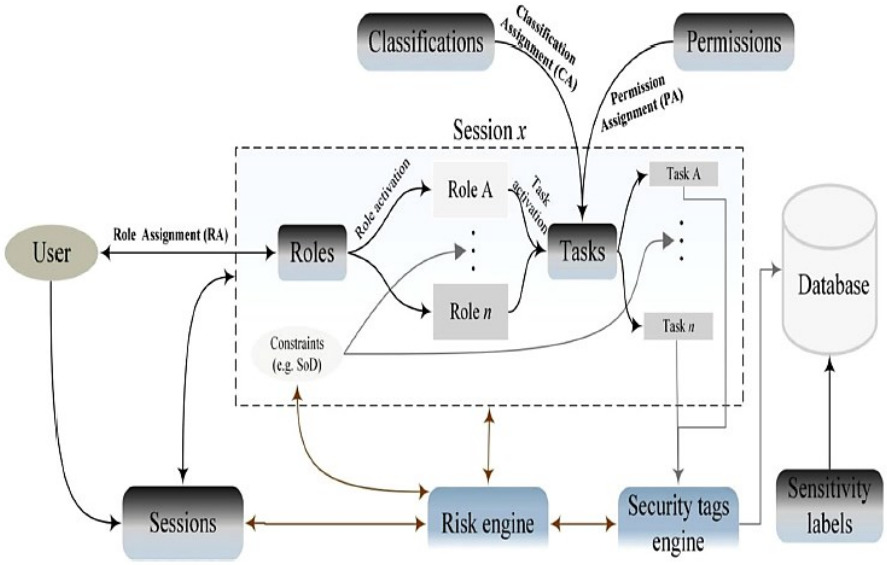


Fig.5. Access Control for Cloud Computing (AC3), adapted from (Younis et al., 2014)

In 2014, Yang and Jia proposed a new multi-authority access control scheme (Yang & Jia, 2014). In this scheme, the authors presented a Ciphertext-Policy Attribute-based Encryption (CP-ABE) scheme. This is an extension to a single-authority scheme proposed by Lewko and Waters in (Lewko & Waters, 2012). Yang and Jia adopted Chase's multi-authority scheme (Chase, 2007) in which all generated secret keys were combined together for the same client. CP-ABE also used a revocable scheme and could mitigate collusion attacks. More specifically, the functionality of a single authority is divided into a certificate authority and multiple attribute authorities. The proposed multi-authority access control scheme is shown in Figure 6.

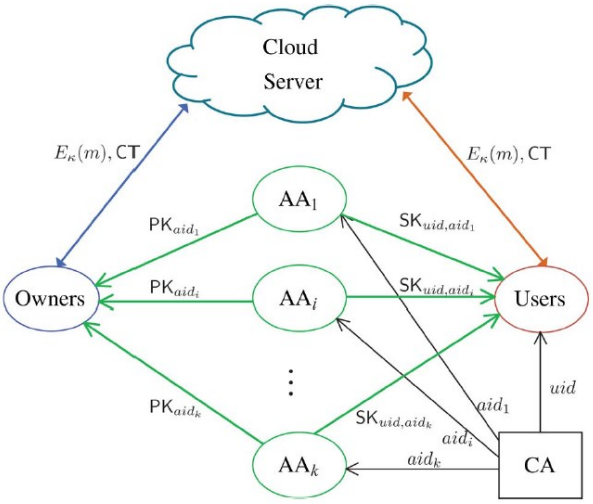


Fig.6. Multi-authority access control scheme, adapted from (Yang & Jia, 2014)

Li et al. adopted Semantic-Based Access Control (SBAC) techniques to propose a new architecture called IntercroSsed Secure Big Multimedia Model (2SBM) for securing accesses between different cloud systems (Li et al., 2016 (B)). In addition to the architecture shown in Figure 7, the 2SBM architecture can be summarized in three steps:

- In order to relate attributes to each other, the proposed architecture formats the data by linking the attributes in a matrix;
- Based on these relationships, the architecture creates interrelations between attributes in the matrix; and
- The architecture builds a tree of attributes and sorts the attributes according to their frequency, to improve the efficiency of access control.

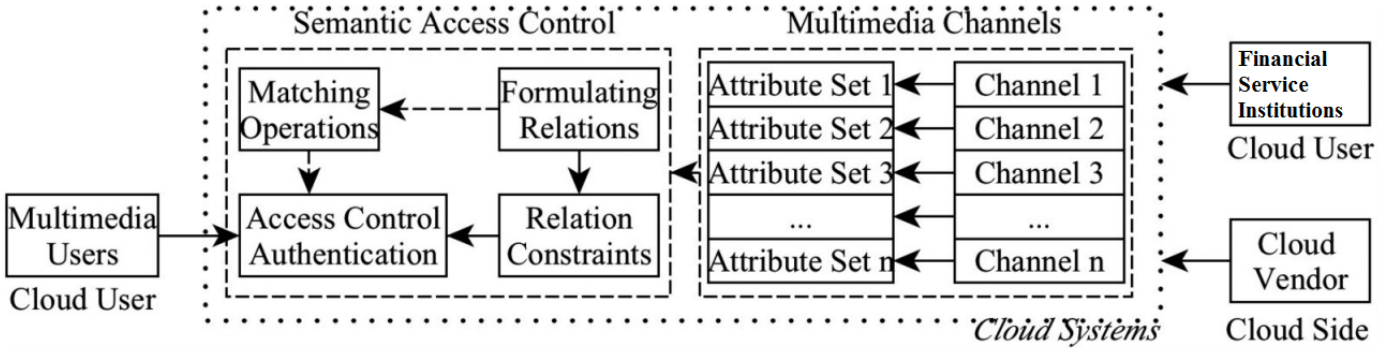


Fig.7. Architecture of 2SBM, adapted from (Li et al., 2016 (B))

Choi et al proposed the ontology-based Access Control Model (Onto-ACM) in 2014 (Choi et al., 2014). Onto-ACM is a model of analysis which recognizes and presents the differences between providers and clients. Based on this analysis, the ontology cognitive and context-aware technologies, the proposed model can decide whether or not to allow data access. This model can be seen as a detailed access control, which can be used to establish cloud feature boundaries.

Yu et al. proposed a new scheme which claimed to achieve secure, scalable, and fine-grained access policies for cloud projects (Yu et al., 2010). The proposed scheme uses attribute-based encryption (ABE), proxy re-encryption (PRE), and lazy re-encryption. Specifically, it allows the data owner to pass the operations of computation to the servers without revealing the original data. In this scheme, the data owner is therefore responsible for the accessibility of the data, which is particularly suitable for cloud projects.

Ruj et al. proposed another form of access control in 2014 (Ruj et al., 2014). In this method, there are three types of clients: creator, reader, and writer. For example, Alice is the client and a trusted party gives her a token. The trusted party could be any government office controlling health records. When submitting a claim, Alice presents her identification (e.g., a health card), and the trusted party provides her with the token. In this scheme, there are two key distribution centres (KDCs) which are responsible for distributing the keys to the clients. Based on the information in the token and the keys from one or two of the KDCs, a creator makes a decision on the claim, ensuring the identity of Alice and authenticating and encrypting the messages under this claim. The signed ciphertext is then sent to the cloud. The cloud system authenticates the signature of the ciphertext and keeps it on the cloud servers. When the reader asks to read a message, the cloud system will send the ciphertext. Without the appropriate keys, the user would not be able to retrieve the plaintext; however, the access control manager has full access to all client information and can decrypt the ciphertexts.

4. Effective Encryption

Several advanced encryption algorithms have been used in cloud computing to protect the security and privacy of eHealth data. Encryption schemes such as public key encryption (PKE) and symmetric key encryption (SKE) are frequently used to protect data in eHealth cloud projects (Abbas & Khan, 2014). Other encryption schemes are also used to ensure the security and privacy of eHealth records, including attribute-based encryption (ABE), fully homomorphic encryption (FHE), and searchable encryption (SE).

4.1. Attribute-Based Encryption Algorithms

The first ABE algorithms were presented by Sahai and Waters in 2005 (Sahai & Waters, 2005), and by Goyal et al. in 2006 (Goyal et al., 2006).

ABE is a type of PKE where the ciphertext and shared key of a client are reliant on attributes. In ABE systems, retrieving a plaintext from ciphertext is applicable for clients who have a group of key attributes that match ciphertext attributes. One of the

most important features of the ABE system is that it is collusion resistant; an attacker who has many keys can only access the system when at least one key has an endorsed access. Recently, numerous researchers have proposed ABE algorithms; some of these are discussed in the following paragraphs.

Fabian et al. proposed a new ABE-based scheme for secure data sharing in eHealth clouds (Fabian et al., 2015). The proposed scheme aims to preserve the security and privacy of patients' records in partly trustworthy cloud servers. It uses the ABE algorithm to manage users' accessibility to health records and shared keys, and to distribute information and health records among several clouds. Figure 8 shows a patient visiting three different health centres (HCs A, B and C). His/her health record is updated at each of the three centres. When the patient visits HC C, the doctors at HC C can request the full health record for that patient from HCs A and B through the multi-cloud proxy. However, the key management process needs to be reconsidered and solved. In addition, the key authority of the ABE algorithm has to be distributed, and security responsibilities must be separated.

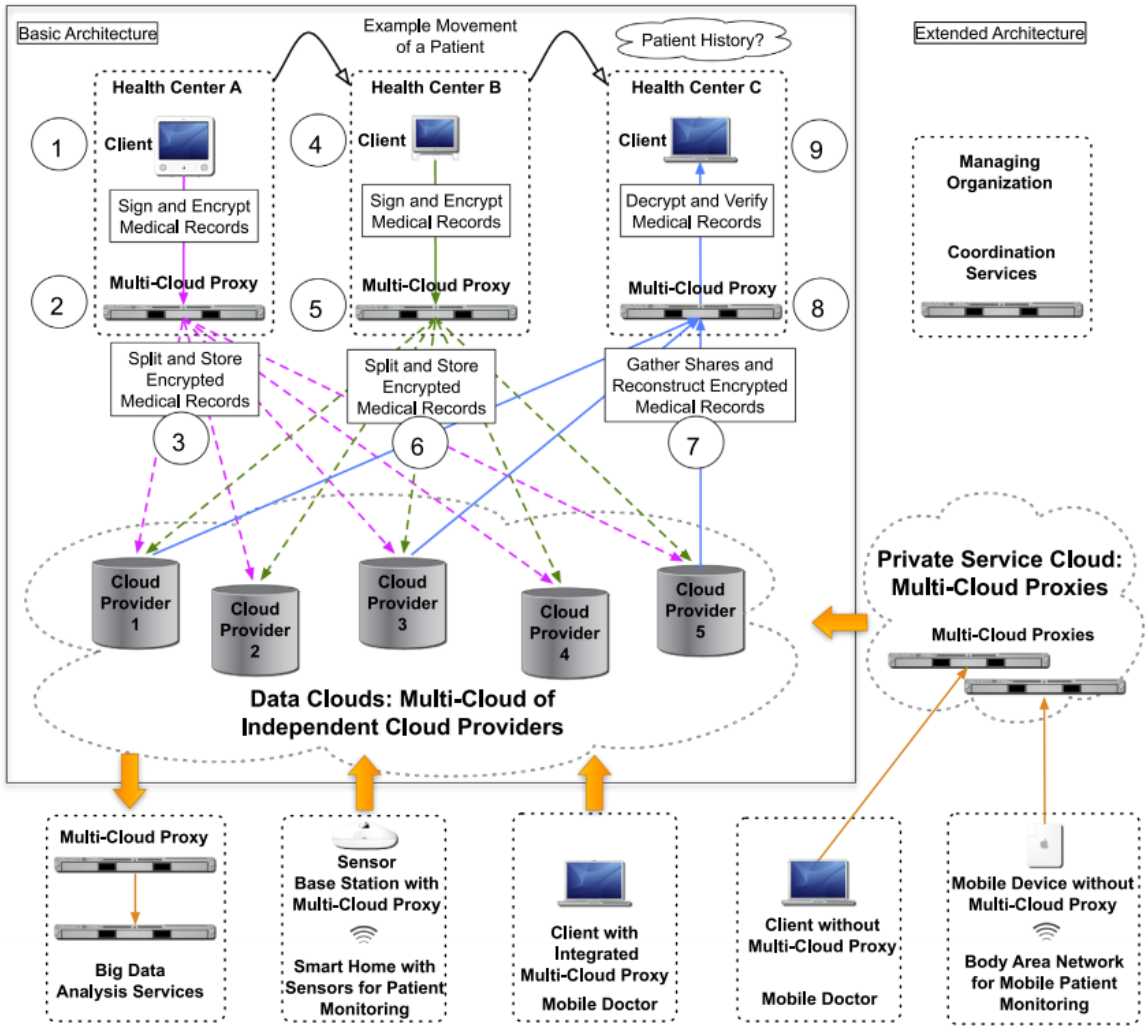


Fig. 8. Scheme proposed by Fabian et al. (Fabian et al., 2015)

Li et al. proposed a new ABE-based framework for secure sharing of PHRs in the eHealth cloud (Li et al., 2013). The authors assumed that the cloud servers were semi-trusted; they also argued that the PHR records had to be encrypted to ensure the privacy of the patients. They used the ABE algorithm to encrypt PHRs, and patients can delegate others from public domains to access their PHR records. This work involved verifying key management complexity reduction and privacy enhancement. The proposed framework involves multiple data owners, clients, attribute authorities (AAs), and SDs. The framework can use one of two ABE algorithms: the revocable key policy ABE system proposed by Yu et al. was used for each public and personal domain (PSD) (Yu et al., 2010), and the authors proposed their own revocable MA-ABE system to be used for each personal and public domain (PUD).

Outsourced ABE (OABE) approaches can significantly decrease the computational cost of encryption by moving the large computation to a CSP. However, large encrypted files which are saved on the cloud are likely to affect the query processing in a negative way. Li et al. therefore proposed a keyword search function (KSF-OABE) approach, which aimed to solve this problem (Li et al., 2017). KSF-OABE offers key issuing, decryption, and keyword search functions. It retrieves part of the ciphertext

according to a particular keyword. In this approach, operations that consume a large amount of time will be moved to the CSP, while users needing less time would process consuming operations. Thus, the processing time can be reduced on both the CSP and user sides. However, the proposed KSF-OABE approach does not offer verifiability features. The proposed approach was tested only for a replayable chosen-ciphertext attack (RCCA) and was not tested for a chosen-ciphertext attack (CCA). CCA-secure approaches are RCCA-secure, although RCCA-secure approaches are not CCA-secure. Therefore, testing under both CCA and RCCA conditions is suggested.

A PHR system based on the ABE algorithm was presented by Xhafa et al. for secure sharing and storing of PHRs in the cloud (Xhafa et al., 2015 (B)). This system permits users to share their PHRs and personal information selectively with health service providers. The proposed system is practical as it provides searchability, revocation, and local decryption.

Based on their operations, ABEs can be classified as ciphertext-policy or key-policy ABEs.

4.1.1. Ciphertext-Policy ABEs

In the ciphertext-policy (CP-ABE) approaches, the encryptor manages the access operation. The public key process is more complex due to the complexity of the access operation and hardens the system. Most CP-ABE research concentrates on the access control design (Su et al., 2011).

Liu et al. proposed a new approach based on CP-ABE with a signature (SignCryption), called CP-ABSC. It delivers PHR authentication, encryption, and access control (Liu et al., 2015 (C)). The proposed approach permits a patient to sign the PHR record using a secret key and a group of personal attributes, as shown in Figure 9. CP-ABSC has two features: access control and signature encryption (SignCryption). The authors claim that a combination of these two features can deliver the authenticity, unforgeability, confidentiality and collusion prevention required by a PHR system. However, a revocation process was not considered. In addition, according to Rao, this approach cannot provide verifiability for a public ciphertext property, which is necessary to resist any invalid ciphertext decryption in order to decrease the redundant load on the decryptor (Rao, 2017).

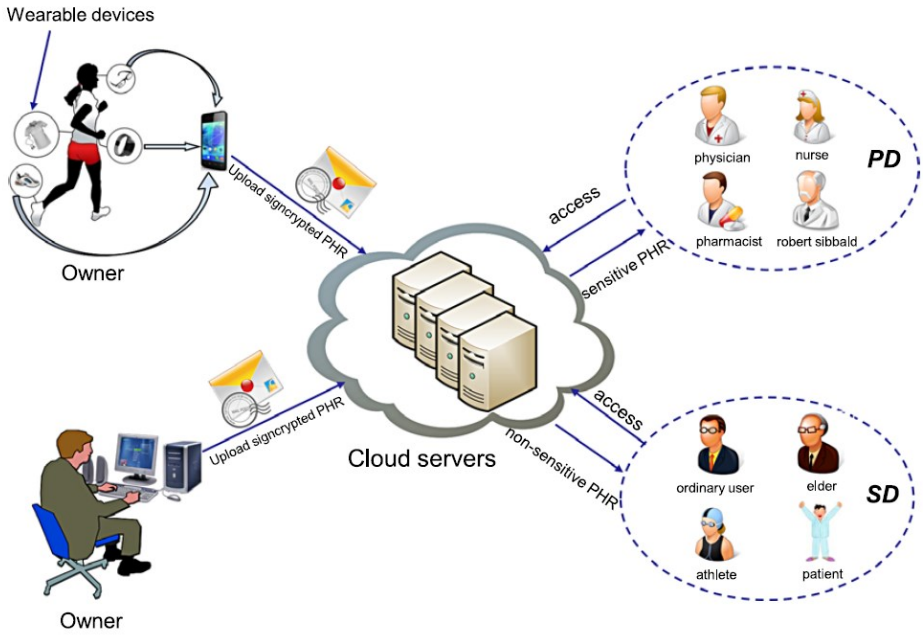


Fig. 9. Approach proposed by Liu et al. (Liu et al., 2015 (C))

As a result, in 2017, Rao proposed another CP-ABSC approach for PHR cloud projects, which claimed to be verifiable for a public ciphertext (Rao, 2017). This approach satisfies the important security properties of the attribute-based signature (ABS) and ABE. Furthermore, it uses communication links to a lesser extent than other approaches. This CP-ABSC has two assumptions: existential unforgeability in selective signing predicate and adaptive chosen message attack (EUF-sSP-CMA) and the resistance of the computational Diffie–Hellman Exponent (cDHE) problem, and decisional Bilinear Diffie–Hellman Exponent (dBDHE) problem" (Rao, 2017). These assumptions can prevent the "indistinguishability of ciphertext in selective encryption predicate and adaptive chosen ciphertext" attack (IND-sEP-CCA2).

Wang et al. (Wang et al., 2015 (A)) introduced another cloud-based PHR (CB-PHR) system. CB-PHR permits the owners of PHRs to safely store their records in a partly trustworthy CSP, and to share them with several clients of their choice. PHR clients were divided into public and personal domains to decrease the complexity of key management. In this approach, health records are encrypted by the owner of the PHRs using CP-ABE for presentation to the public domain, whereas health records are encrypted using a nameless multi-receiver identity-based encryption algorithm for the personal domain. Therefore, only accredited clients whose identification can meet the CP specifications can decrypt health records (Wang et al., 2015 (A)). The authors suggest services such as statistics and retrieval of data. It should be mentioned that the CB-PHR has a high computational cost, as it encrypts the same record twice.

Motivated by cloud security requirements, Xu et al. modified the CP-ABE scheme to propose a new Verifiable Delegation CP-ABE (VDCPABE) (Xu et al., 2016). This cloud computing scheme is based on verifiable technology and multilinear maps. Hybrid encryption is used to encrypt data by its owner. For each ciphertext block, a verifiable message authentication code (MAC) is generated privately, and the full ciphertext is then uploaded to the cloud. When the data owner is not online, the client who requested the data can ask the cloud server directly (Xu et al., 2016).

Health records are usually represented using a multilayer hierarchical structure. However, according to Wang et al., this hierarchical characteristic of health records has not been investigated thoroughly in terms of CP-ABE (Wang et al., 2016). They therefore proposed a data hierarchy ABE approach for such cloud projects. A single access control method was used rather than levelled access control methods, and the hierarchical data were encrypted using this single access control method. The proposed scheme was shown to reduce storage and time costs, since the parts of the ciphertext which were related to attributes are distributed by the records (Wang et al., 2016).

A PHR privacy preserving approach based on a multi-authority CP-ABE which offers revocation features and ensures fine-grained access was proposed by Qian et al. (Qian et al., 2015). The authors show that this approach can be implemented in a partly trustworthy server and encrypted PHRs with multiple owners can be stored on that server. The proposed approach was able to work in public cloud PHR systems (Li et al., 2013). Once PHRs encryption is complete, in order to achieve a fine-grained access, the patient can combine ciphertext with multilayer access attributes. A key exchange scheme was used to preserve the privacy of the PHRs. This key exchange scheme ensures that if cracked, authorities will expose zero information regarding the client's global identifier (GId). As a result, the tracing of a GId by an attacker yields no information about the client's attributes. The revocation of lazy client and on-demand services are features provided by this approach to decrease the computational overhead (Qian et al., 2015).

Another approach based on CP-ABE was proposed by Guo et al. to secure EHRs in health cloud environments (Guo et al., 2016). This approach uses a CP-ABE algorithm to encrypt tables published by healthcare providers, such as EHRs. The patient's identification number is used as a primary key to store these records in a database. This permits multiple clients with multiple constraints to search multiple database columns. The authors highlight that their work differs from others in terms of securing outsourcing records, as the search management of columns in the database is emphasized (Guo et al., 2016).

Xhafa et al. presented a multi-authority CP-ABE approach with a patient accountability feature to secure PHR sharing in a health cloud project (Xhafa et al., 2015 (A)). In the proposed work, patient privacy was secured by the access control policy being hidden. The reduction of authority and PHRs trust assumptions were ensured through the accountability feature.

4.1.2. Key-Policy ABE (KP-ABE)

In the Key-Policy Attribute-Based Encryption (KP-ABE) schemes, ciphertext has a group of attributes, and the access regulations are controlled by the client's private key. Ciphertext can be decrypted only when these groups of attributes match the structure of access to the client's private key (Attapadung et al., 2012).

Based on the Decisional Bilinear Deffie-Hellman (DBDH) assumption, a privacy-preserving KP-ABE (PP KP-ABE) approach was proposed for secure data sharing in a cloud system (Rahulamathavn et al., 2016). This approach permits clients to retrieve data from the cloud and then decrypt it, without exposing any attribute information to a third party. The issue of collusion attacks has been resolved in this research, as PP KP-ABE is collusion resistant. The authors of PP KP-ABE utilized a key management scheme to strengthen the connection between client and the secret key; thus, multiple clients cannot use their secret keys to produce a secret key for an unapproved client (Rahulamathavn et al., 2016).

Another KP-ABE-based scheme named access policy re-definable ABE (APR-ABE) was proposed by Qin et al. for securing EHRs in cloud environments (Qin et al., 2015). In APR-ABE, attribute vectors were used to implement access control. This access control was linked to clients' secret keys; higher level clients can easily redefine their access control to be commensurate with their roles, and can then provide lower level clients with a secret key that has more limitations.

4.2. Fully Homomorphic Encryption (FHE)

FHE is a type of encryption that has a special feature permitting operations to be done on a ciphertext as well as on plaintext (Maral et al., 2016). This is a vital feature, especially in modern ICT systems, as it enables the possibility of chaining several services together without leaking information. There are several schemes which secure health records using FHE, and we discuss some of these in the following paragraphs.

A FHE-based scheme was proposed to secure computations for the Genome-Wide Association Study (GWAS) (Lu et al., 2015). The proposed scheme aims to preserve the privacy of patients' genomic data. It adapts FHE to encrypt genotype and phenotype data for all patients in order to implement meaningful operations on a ciphertext. However, the authors do not consider the computational complexity of the FHE in their proposed scheme, which is a major issue for the proposed FHE scheme (Kumarage et al., 2016).

Another approach based on FHE was proposed to preserve the privacy of health data in a public cloud (Kocabas & Soyata, 2015; Page et al., 2014). A detailed analysis was provided based on heart rate (average), heart rate (max/min), and the automated detection of irregular heartbeats. The authors provided a set of experimental results over 24 hours using an electrocardiogram (ECG) signal dataset and a homomorphic encryption library (HElib). The results show that the proposed approach can be adapted for a health cloud system to secure data from these issues (Kocabas & Soyata, 2015; Page et al., 2014). However, the proposed scheme does not solve the problem of computational complexity in FHE. The implementation of this approach in a real-time parallel system also needs to be considered to reduce the procedure time.

Zhao et al. proposed a different FHE-based system to solve the issue of lack of data safety in a health cloud (Zhao et al., 2014). The authors claim that the proposed method is suitable for both retrieving and processing ciphertext for secure storage of health data on cloud servers and transmission of data between the cloud and the clients. This method was able to offer search date for a third party. However, in the same way as the previous methods, this method also suffers from high computation requirements.

4.3. Searchable Encryption (SE)

SE is a cryptographic scheme that provides safe search in a ciphertext. For enhanced effectiveness, SE typically constructs keyword indexes to verify client requests. SE schemes can be based either on a public key or secret key. Many proposals have been proposed to deliver secure search over encrypted text, and some of these are described below.

Yang and Ma have proposed a time-dependent SE approach with a designated tester and timing enabled proxy re-encryption function (Re-dtPECK) (Yang & Ma, 2016). This approach permits patients to give limited access privileges to others, which helps to control search procedures over the health records within a certain timeframe. People who are given access privileges by patients can search and decrypt health records within this limited timeframe. In addition, Re-dtPECK offers a linked word search, and can prevent guessing attacks (Yang & Ma, 2016). However, the revocation feature was not considered in this approach, as the patient holds the same key most of the time, meaning that Re-dtPECK needs to consider redistributing secret keys among authorized clients.

A scheme named secure channel-free searchable encryption (SCF-PEKS) has been proposed to offer a secure search over encrypted EHR (Wu et al., 2016 (B)). This version of SCF-PEKS was shown to be able to reduce storage and computational costs when compared to the previous SCF-PEKS. Moreover, it can resist keyword guessing attacks. However, despite reductions in storage and computation costs, ranked and fuzzy keyword searches were not provided, and integrity checks were missing.

Another proposed scheme uses a Bloom filter tree index to permit accredited users to retrieve data from ciphertext in a cloud (Song et al., 2017). In addition to the proposed scheme, the authors introduced a ranking method based on keyword membership, to retrieve only vital keywords. The authors argued that their work was the first to be able to retrieve full encrypted text from a large cloud storage database. However, a collusion attack could possibly threaten the proposed scheme.

Liu et al. proposed a novel EHR cloud project which aimed to safely share and store EHR records in a cloud environment (Liu et al., 2016). The proposed approach is based on binary trees for saving EHR ciphertext, and the ABE algorithm was adopted for efficient encryption of the shared keys. The authors claimed that the proposed project was designed to secure EHRs, and these were encrypted using a symmetric algorithm. With fewer cryptographic operations, a searchable encryption scheme might improve the system further. However, integrity checks were not offered by the proposed system.

Since the security of data sharing is an important factor for any cloud-based system, especially health cloud systems, Liang and Susilo defined a new notation searchable attribute-based proxy re-encryption (ABPRE) scheme to address this issue (Liang & Susilo, 2015). However, the authors did not state how they might reduce the search token size, and how a key holder could create tokens. A modified scheme was recommended to address these issues.

In addition, Li et al. introduced two fine-grained multi-keyword search (FMS) schemes, FMS_I and FMS_II (Li et al., 2016 (A)). FMS_I was designed to provide an accurate search by considering common keyword factors and related scores. FMS_II was built to offer a secure complex search, which might contain several keywords connected with logical operations such as "AND"

and “OR” operations. Finally, to enhance the efficiency of the proposed schemes, FMS classified support (FMSCS) sub-dictionaries were proposed. However, the proposed method cannot deal with a multi-user cloud.

Finally, a multi-keyword SE method was proposed to safely search over encrypted text on a cloud (Xia et al., 2016). This method was able to offer dynamic operations such as insert and delete operations. The authors designed their own tree-based index, as well as a “greedy depth-first search” method to enhance the ranked search using multiple keywords. They chose the KNN algorithm to encrypt the query and the index. In addition, this algorithm was chosen to compute the score of the connections between the query and the index. Shade terms were inserted into the index to prevent statistical attacks. However, a revocation feature is not offered by the proposed approach, as the patient holds the same key most of the time, as in Re-dtPECK discussed above.

5. Data Security Requirements

Several security issues are related to cloud systems, such as EHR cloud-based systems. These issues include not only common concerns such as DDoS attacks (Sahi et al., 2017), but also specific issues in the cloud such as side channel attacks, etc. (Tang et al., 2016). Thus, setting security requirements for any cloud systems is essential and needs to be included in our review. From an eHealth cloud perspective, the security requirements (R) of cloud system include the following.

5.1. Confidentiality (R1)

The confidentiality of data in a health cloud system means that unauthorized clients cannot decrypt or retrieve health records. The data owner, for example the patient, does not control the health records stored in the cloud (Li et al., 2013). Authorized clients are the only users who can access the records; even CSPs are not allowed to access any information regarding the data. Furthermore, patients expect full control over their health records in the cloud, without any leakage to other legitimate users or attackers.

5.2. Access Controllability (R2)

Access controllability means that a data owner controls his/her data by implementing certain carefully constructed rules in order to ensure the security and privacy of records, and by allowing only legal users to have controlled access (Tang et al., 2016). Other users cannot access health records without permission. Users have different access rights to access different parts of the data; this is called fine-grained access control. In an untrusted cloud system, the data owner is the only one permitted to grant access.

5.3. Integrity (R3)

Integrity is a security feature that ensures the completeness and accuracy of data. In other words, data must stay complete and must not be altered or deleted; users normally expect their data to be kept safe in cloud storage (Liu et al., 2015 (A)). Furthermore, users must be able to detect any unsolicited modification, loss, or corruption of this data, and to retrieve lost pieces.

5.4. Authenticity (R4)

Authenticity means that only an authentic user can request access (Abbas & Khan, 2014). In the health sector, EHR service providers must provide verified information to ensure the authenticity of the cloud.

5.5. Reliability (R5)

Reliability means that the system performs as users expect (Lehr, 2015). One of the main factors of reliability is availability, which means the continuity of services provided. In other words, availability means how long the system is expected to serve users without interruption (Lehr, 2015).

5.6. Accountability (R6)

As cited in (Felici & Pearson, 2015), *“defining what exactly accountability means in practice is complex”*. One definition is that the controller of the data must be responsible for acting in accordance with procedures that affect the privacy of data.

5.7. Auditability (R7)

Auditability means monitoring security, privacy, and all access activities on the eHealth cloud (Abbas & Khan, 2014). Auditing must be done from time to time, to ensure that no errors accrue.

5.8. Non-Repudiation (R8)

Non-repudiation means that no one can falsely deny any unethical behaviour (Chen et al., 2014; Mihaita et al., 2017). In the eHealth cloud environment, patients and physicians cannot deny any misuse or mishandling of health records.

5.9. Anonymity (R9)

The anonymity of the user means preventing a third party from obtaining valid user information that leads to accessing the server (Jiang et al., 2016 (B); Sharma & Kalra, 2016). Anonymity ensures the privacy of legitimate users in the cloud, as the attacker is unable to learn any personal information. A lack of anonymity means an attacker can be fake an identity as an authenticated user.

5.10. Unlinkability (R10)

Unlinkability means that in order to ensure a user's privacy, associating information with a particular user must be difficult (Wu et al., 2016 (A)). Although sometimes a group of words needs to be used for a particular function, this group of words should be different each time. Thus, a random generation function is required (Liu et al., 2015 (B)).

5.11. Maintainability (R11)

Maintainability means the ability to perform fast maintenance on a project, as the development of very large projects is often not fully complete (Biswas et al., 2014). Maintainability can therefore ensure the delivery of services without error for different parties. In addition, a testing method is needed to decrease the time of maintenance.

5.12. Revocability (R12)

Revocability means that users' access rights should be revoked after a period of time, so that they cannot access specific data later on using old keys (Thilakanathan et al., 2014). Revocability is a vital feature for eHealth cloud systems and needs to be well implemented in order to ensure the privacy of users and the secrecy of the contents (Yang, 2015). Once a manager chooses to revoke a particular user's rights, the corresponding keys need to be eliminated from the system.

Table 1 shows a comparison of security approaches for the eHealth cloud in terms of data security requirements.

Table 1. Comparison of security approaches for the eHealth cloud

Ref.	Technique(s)	Aim(s)	Limitation(s)	Server assumption(s)	Data Security Requirements											
					R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
(Thilakanathan et al., 2014)	Proxy re-encryption, El-Gamal encryption	In-home monitoring	Data sharing service assumed to be fully trusted party, usability tests not available	–	✓	✓	✗	✗	–	✗	✗	✗	✗	✗	✗	✓
(Castiglione et al., 2015)	Least Significant Bit	3D medical images secure management, limited computational power	Unparalleled, missing many security requirements	–	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✓	✗
(Fabian et al., 2015)	ABE, cryptographic secret sharing	Multiple cloud data distribution, reduction in attackers' abilities	Centralized ABE key authority, separated security duties, usability tests not available	Semi-trusted servers	✓	✓	✓	✓	✓	–	✓	✓	✓	✓	✓	✓
(Hu et al., 2017)	Internet of Things (IoT) medical sensor	Secure elder monitoring system, medical resource reduction	Computationally inefficient, missing some vital security requirements such as revocation	Trusted servers	✓	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✗
(Li et al., 2013)	ABE	Patient-centric PHR access control, multiple clouds, key complexity reduction	Computationally inefficient	Semi-trusted servers	✓	✓	✓	✓	✓	✓	✗	–	✗	✗	–	✓
(He et al., 2014)	Multiple hashes	Preventing Denial of Service (DoS) attacks, dissemination protocol for WBANs	Unparalleled as it uses CBC, many assumptions, missing vital requirements such as revocation	–	✓	✓	✓	✓	–	✗	✗	✓	✗	✗	✓	✗
(Gope & Hwang, 2016)	IoT, Body Sensor Network (BSN)	Secure IoT-based healthcare system using BSN , computationally efficient	Missing vital security requirements such as revocation	–	✓	✓	✓	✓	–	✗	✗	✓	✓	✓	✗	✗
(Wu et al., 2016 (A))	Bilinear pairing, Authenticated key Exchange	Secure anonymous authentication for WBAN, computationally efficient	Vulnerable to client impersonation attack	Trusted servers	✓	✓	✓	✓	–	✗	✗	✓	✓	✓	✗	✗
(Jiang et al., 2016 (B))	Elliptic curve cryptosystem (ECC), bilinear pairing	Secure anonymous authentication for WBAN, computationally efficient	Missing vital security requirements such as revocation	Trusted servers	✓	✓	✓	✓	–	✗	✗	✓	✓	✓	✗	✗
(Chen et al., 2014)	Symmetric encryption, MAC, RFC 2631	Anywhere anytime access	Missing vital security requirements such as anonymity and unlinkability	Trusted servers	✓	✓	✓	✓	–	✗	✗	✓	✗	✗	✗	✓

(Sharma & Kalra, 2016)	Quantum key distribution	Claimed to resist all attacks, generate keys over distance of 100km of optical fibre	Missing vital security requirements such as revocation	Trusted servers	√	√	√	√	√	√	X	√	√	X	X	X
(Yang, 2015)	ABE, SE, bilinear pairing	Ciphertext retrieval, SE scheme supports fine-grained access control	Missing vital security requirements such as anonymity and unlinkability	Trusted servers	√	√	√	X	X	X	X	√	X	X	X	√

Note: √ = Valid X = Invalid – = Not specified

6. Disaster Recovery Plans

The CSPs must establish continuity and recovery plans to ensure that services will remain available, and can recover all lost data even after disasters such as floods, earthquakes, or electricity power failures. The data recovery plan may be established solely by CSPs, or in consultation with clients.

Several suggestions have been made to facilitate disaster recovery, and some of these are discussed below.

Sahi et al. presented a disaster recovery plan to ensure the availability of PHRs and HERs in a health cloud environment (Sahi et al., 2016). The authors assume that the cloud storage consists of three or more data centres. Distributing signals called heartbeats are used between data centres and the CSP, in order to keep track of the status of these data centres. Each health record is divided into several parts, and multiple copies of each part are stored in different data centres. In the case of a disaster, the heartbeat from a data centre would stop if the data centre machine was damaged, which would alert the manager. The manager can recover or retrieve the records from the other data centres, with no need to access the damaged one. Finally, the authors pointed out that the data centres must be physically located in different geographic locations (for example in different countries) to ensure the availability of the data and the services (Sahi et al., 2016).

Another disaster recovery plan has been proposed based on three different techniques: TCP/IP, VM snapshots, and replication (Chang, 2015). The proposed plan claims to achieve 99.94% data recovery in the event of a disaster. The proposed approach was implemented with real data and tests involving backing up all sister site records in London, Southampton, and Leeds. However, the data centres in the proposed approach are not integrated with any existing ones. In addition, all data centres are located within the same geographical area, which could be considered a major drawback.

Gu et al. proposed backup and recovery models for implementing a disaster recovery plan (Gu et al., 2014). In terms of the backup model, clients are provided with accounts with limited rights. The CSP is responsible for sending and receiving data to/from clients. A client is able to request a backup from the CSP within a certain timeframe; the CSP will hold this request, make three copies of the data and store these in different locations. In the recovery model, the client can request a data recovery from the CSP. The CSP can retrieve the data from the stored three copies and send it back to the client. However, storing the data in full at three different locations can significantly increase the backup data size.

Mansoori et al. presented a disaster recovery plan based on two servers, a local server and a disaster recovery server (Mansoori et al., 2014). The proposed plan considers four scenarios to provide availability and continuity of services. The authors implemented the proposed plan within a university hospital health system to ensure constant access to the picture archiving and communication system (PACS) application and its controlled radiology images. However, the authors did not consider a scenario in which a disaster affects a relatively wide geographic area leading to damage to the backup images.

7. Conclusion

The security and privacy of health data in the cloud requires secure solutions that are capable of controlling security and privacy while keeping all features of eHealth under consideration. In this paper, we review the state of the art of security and privacy in the eHealth cloud from five main perspectives: security and privacy, security controls, effective encryption, data security requirements, and disaster recovery plans. This paper therefore provides a clear overall picture of the development of eHealth to stakeholders, in order to facilitate better designs and decisions. In summary, this paper collects, evaluates, and classifies state-of-the-art eHealth security and privacy schemes. In addition, it covers the most recent studies in this area, and discusses the drawbacks of certain proposals to help improve the security and privacy of the eHealth cloud.

References

Abbas, A & Khan, SU 2014, 'A review on the state-of-the-art privacy-preserving approaches in the e-health clouds', *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 4, pp. 1431-41.

Attrapadung, N, Herranz, J, Laguillaumie, F, Libert, B, De Panafieu, E & Ràfols, C 2012, 'Attribute-based encryption schemes with constant-size ciphertexts', *Theoretical Computer Science*, vol. 422, pp. 15-38.

Biswas, S, Akhter, T, Kaiser, M & Mamun, S 2014, 'Cloud based healthcare application architecture and electronic medical record mining: An integrated approach to improve healthcare system', in *Computer and Information Technology (ICCIT), 2014 17th International Conference on*: *proceedings of theComputer and Information Technology (ICCIT), 2014 17th International Conference on* IEEE, pp. 286-91.

Burrows, M, Abadi, M & Needham, RM 1989, 'A logic of authentication', in *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*: *proceedings of theProceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* The Royal Society, pp. 233-71.

Carlson, FR 2014, 'Security analysis of cloud computing', *arXiv preprint arXiv:1404.6849*.

Castiglione, A, Pizzolante, R, De Santis, A, Carpentieri, B, Castiglione, A & Palmieri, F 2015, 'Cloud-based adaptive compression and secure management services for 3D healthcare data', *Future Generation Computer Systems*, vol. 43, pp. 120-34.

Chang, V 2015, 'Towards a Big Data system disaster recovery in a Private Cloud', *Ad Hoc Networks*, vol. 35, pp. 65-82.

Chase, M 2007, 'Multi-authority attribute based encryption', in *Theory of Cryptography Conference*: *proceedings of theTheory of Cryptography Conference* Springer, pp. 515-34.

Chen, C-L, Chen, Y-Y, Lee, C-C & Wu, C-H 2014, 'Design and analysis of a secure and effective emergency system for mountaineering events', *The Journal of Supercomputing*, vol. 70, no. 1, pp. 54-74.

Choi, C, Choi, J & Kim, P 2014, 'Ontology-based access control model for security policy reasoning in cloud computing', *The Journal of Supercomputing*, vol. 67, no. 3, pp. 711-22.

Dou, W, Zhang, X, Liu, J & Chen, J 2015, 'HireSome-II: Towards privacy-aware cross-cloud service composition for big data applications', *IEEE transactions on parallel and distributed systems*, vol. 26, no. 2, pp. 455-66.

Fabian, B, Ermakova, T & Junghanns, P 2015, 'Collaborative and secure sharing of healthcare data in multi-clouds', *Information Systems*, vol. 48, pp. 132-50.

Felici, M & Pearson, S 2015, 'Accountability for data governance in the cloud', in *Accountability and Security in the Cloud*, Springer, pp. 3-42.

Fernández-Alemán, JL, Señor, IC, Lozoya, PÁO & Toval, A 2013, 'Security and privacy in electronic health records: A systematic literature review', *Journal of biomedical informatics*, vol. 46, no. 3, pp. 541-62.

Fernando, R, Ranchal, R, An, B, Othman, LB & Bhargava, B 2016, 'Consumer Oriented Privacy Preserving Access Control for Electronic Health Records in the Cloud', in *Cloud Computing (CLOUD), 2016 IEEE 9th International Conference on*: *proceedings of theCloud Computing (CLOUD), 2016 IEEE 9th International Conference on* IEEE, pp. 608-15.

González-Martínez, JA, Bote-Lorenzo, ML, Gómez-Sánchez, E & Cano-Parra, R 2015, 'Cloud computing and education: A state-of-the-art survey', *Computers & Education*, vol. 80, pp. 132-51.

Gope, P & Hwang, T 2016, 'BSN-Care: A secure IoT-based modern healthcare system using body sensor network', *IEEE Sensors Journal*, vol. 16, no. 5, pp. 1368-76.

Goyal, V, Pandey, O, Sahai, A & Waters, B 2006, 'Attribute-based encryption for fine-grained access control of encrypted data', in *Proceedings of the 13th ACM conference on Computer and communications security*: *proceedings of theProceedings of the 13th ACM conference on Computer and communications security* Acm, pp. 89-98.

Grizzle, K, Wahlstroem, E, Mortimore, C & Hunt, P 2015, 'System for cross-domain identity management: Core schema', *System*.

Gu, Y, Wang, D & Liu, C 2014, 'DR-Cloud: Multi-cloud based disaster recovery service', *Tsinghua Science and Technology*, vol. 19, no. 1, pp. 13-23.

Guo, C, Zhuang, R, Jie, Y, Ren, Y, Wu, T & Choo, K-KR 2016, 'Fine-grained database field search using attribute-based encryption for e-healthcare clouds', *Journal of medical systems*, vol. 40, no. 11, p. 235.

Haghighat, M, Zonouz, S & Abdel-Mottaleb, M 2015, 'CloudID: Trustworthy cloud-based and cross-enterprise biometric identification', *Expert Systems with Applications*, vol. 42, no. 21, pp. 7905-16.

Haufe, K, Dzombeta, S & Brandis, K 2014, 'Proposal for a security management in cloud computing for health care', *The Scientific World Journal*, vol. 2014.

He, D, Chan, S, Zhang, Y & Yang, H 2014, 'Lightweight and confidential data discovery and dissemination for wireless body area networks', *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 2, pp. 440-8.

Hu, J-X, Chen, C-L, Fan, C-L & Wang, K-h 2017, 'An Intelligent and Secure Health Monitoring Scheme Using IoT Sensor Based on Cloud Computing', *Journal of Sensors*, vol. 2017.

Jiang, Q, Khan, MK, Lu, X, Ma, J & He, D 2016, 'A privacy preserving three-factor authentication protocol for e-Health clouds', *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3826-49. (A)

Jiang, Q, Lian, X, Yang, C, Ma, J, Tian, Y & Yang, Y 2016, 'A bilinear pairing based anonymous authentication scheme in wireless body area networks for mHealth', *Journal of medical systems*, vol. 40, no. 11, p. 231. (B)

Jing, S-Y, Ali, S, She, K & Zhong, Y 2013, 'State-of-the-art research study for green cloud computing', *The Journal of Supercomputing*, pp. 1-24.

Kahani, N, Elgazzar, K & Cordy, JR 2016, 'Authentication and Access Control in e-Health Systems in the Cloud', in *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2016 IEEE 2nd International Conference on* IEEE, pp. 13-23.

Khader, AS & Lai, D 2015, 'Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol', in *Telecommunications (ICT), 2015 22nd International Conference on*: *proceedings of theTelecommunications (ICT), 2015 22nd International Conference on* IEEE, pp. 204-8.

Khalil, I, Khreishah, A & Azeem, M 2014, 'Consolidated Identity Management System for secure mobile cloud computing', *Computer networks*, vol. 65, pp. 99-110.

Khan, MK & Kumari, S 2013, 'An improved biometrics-based remote user authentication scheme with user anonymity', *BioMed research international*, vol. 2013.

Kocabas, O & Soyata, T 2015, 'Utilizing homomorphic encryption to implement secure and private medical cloud computing', in *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*: *proceedings of theCloud Computing (CLOUD), 2015 IEEE 8th International Conference on* IEEE, pp. 540-7.

Kumarage, H, Khalil, I, Alabdulatif, A, Tari, Z & Yi, X 2016, 'Secure data analytics for cloud-integrated internet of things applications', *IEEE Cloud Computing*, vol. 3, no. 2, pp. 46-56.

Lehr, W 2015, '3 Reliability and the Internet Cloud', *Regulating the Cloud: Policy for Computing Infrastructure*, p. 87.

Lewko, AB & Waters, B 2012, 'New Proof Methods for Attribute-Based Encryption: Achieving Full Security through Selective Techniques', in *CRYPTO*: *proceedings of theCRYPTO* Springer, pp. 180-98.

Li, H, Yang, Y, Luan, TH, Liang, X, Zhou, L & Shen, XS 2016, 'Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data', *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 3, pp. 312-25. (A)

Li, J, Lin, X, Zhang, Y & Han, J 2017, 'KSF-OABE: outsourced attribute-based encryption with keyword search function for cloud storage', *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 715-25.

Li, M, Yu, S, Zheng, Y, Ren, K & Lou, W 2013, 'Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption', *IEEE transactions on parallel and distributed systems*, vol. 24, no. 1, pp. 131-43.

Li, Y, Gai, K, Ming, Z, Zhao, H & Qiu, M 2016, 'Intercrossed access controls for secure financial services on multimedia big data in cloud systems', *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 12, no. 4s, p. 67. (B)

Liang, K & Susilo, W 2015, 'Searchable attribute-based mechanism with efficient data sharing for secure cloud storage', *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 1981-92.

Liu, C, Yang, C, Zhang, X & Chen, J 2015, 'External integrity verification for outsourced big data in cloud and IoT: A big picture', *Future Generation Computer Systems*, vol. 49, pp. 58-67. (A)

Liu, D, Dai, Y, Luan, T & Yu, S 2015, 'Personalized search over encrypted data with efficient and secure updates in mobile clouds', *IEEE Transactions on Emerging Topics in Computing*. (B)

Liu, J, Huang, X & Liu, JK 2015, 'Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption', *Future Generation Computer Systems*, vol. 52, pp. 67-76. (C)

Liu, Z, Weng, J, Li, J, Yang, J, Fu, C & Jia, C 2016, 'Cloud-based electronic health record system supporting fuzzy keyword search', *Soft Computing*, vol. 20, no. 8, pp. 3243-55.

Lu, W-J, Yamada, Y & Sakuma, J 2015, 'Privacy-preserving genome-wide association studies on cloud environment using fully homomorphic encryption', *BMC medical informatics and decision making*, vol. 15, no. 5, p. S1.
Mansoori, B, Rosipko, B, Erhard, KK & Sunshine, JL 2014, 'Design and implementation of disaster recovery and business continuity solution for radiology PACS', *Journal of digital imaging*, vol. 27, no. 1, pp. 19-25.
Maral, V, Kale, S, Balharpure, K, Bhakkad, S & Hendre, P 2016, 'Homomorphic Encryption for Secure Data Mining in Cloud', *International Journal of Engineering Science*, vol. 4533.
Mell, P & Grance, T 2009, 'The NIST definition of cloud computing. National Institute of Standards and Technology (NIST)', *Information Technology Laboratory.[Online]. Available: http://csrc. nist. gov/groups/SNS/cloud-computing/index. html*, vol. 53, p. 50.
Mihaita, A-E, Dobre, C, Pop, F, Mavromoustakis, CX & Mastorakis, G 2017, 'Secure Opportunistic Vehicle-to-Vehicle Communication', in *Advances in Mobile Cloud Computing and Big Data in the 5G Era*, Springer, pp. 229-68.
Mxoli, A, Gerber, M & Mostert-Phipps, N 2014, 'Information security risk measures for Cloud-based Personal Health Records', in *Information Society (i-Society), 2014 International Conference on*: *proceedings of theInformation Society (i-Society), 2014 International Conference on* IEEE, pp. 187-93.
Nedelcu, B, Stefanet, M-E, Tamasescu, I-F, Tintoiu, S-E & Vezeanu, A 2015, 'Cloud Computing and its Challenges and Benefits in the Bank System', *Database Systems Journal*, vol. 5, no. 1, pp. 45-58.
Nuñez, D & Agudo, I 2014, 'BlindIdM: A privacy-preserving approach for identity management as a service', *International Journal of Information Security*, vol. 13, no. 2, pp. 199-215.
Page, A, Kocabas, O, Ames, S, Venkitasubramaniam, M & Soyata, T 2014, 'Cloud-based secure health monitoring: Optimizing fully-homomorphic encryption for streaming algorithms', in *Globecom Workshops (GC Wkshps), 2014*: *proceedings of theGlobecom Workshops (GC Wkshps), 2014* IEEE, pp. 48-52.
Page, A, Kocabas, O, Soyata, T, Aktas, M & Couderc, JP 2015, 'Cloud-Based Privacy-Preserving Remote ECG Monitoring and Surveillance', *Annals of Noninvasive Electrocardiology*, vol. 20, no. 4, pp. 328-37.
Qian, H, Li, J, Zhang, Y & Han, J 2015, 'Privacy-preserving personal health record using multi-authority attribute-based encryption with revocation', *International Journal of Information Security*, vol. 14, no. 6, pp. 487-97.
Qin, B, Deng, H, Wu, Q, Domingo-Ferrer, J, Naccache, D & Zhou, Y 2015, 'Flexible attribute-based encryption applicable to secure e-healthcare records', *International Journal of Information Security*, vol. 14, no. 6, pp. 499-511.
Rahulamathavan, Y, Veluru, S, Han, J, Li, F, Rajarajan, M & Lu, R 2016, 'User Collusion Avoidance Scheme for Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption', *IEEE Transactions on computers*, vol. 65, no. 9, pp. 2939-46.
Rajamani, T, Sevugan, P & Purushotham, S 2016, 'An Investigation on the techniques used for encryption and authentication for data security in cloud computing', *IIOAB Journal*, vol. 7, no. 5, pp. 126-38.
Rao, YS 2017, 'A secure and efficient Ciphertext-Policy Attribute-Based Signcryption for Personal Health Records sharing in cloud computing', *Future Generation Computer Systems*, vol. 67, pp. 133-51.
Regola, N & Chawla, NV 2013, 'Storing and using health data in a virtual private cloud', *Journal of medical Internet research*, vol. 15, no. 3.
Rodrigues, JJ, De La Torre, I, Fernández, G & López-Coronado, M 2013, 'Analysis of the security and privacy requirements of cloud-based electronic health records systems', *Journal of medical Internet research*, vol. 15, no. 8.
Ruj, S, Stojmenovic, M & Nayak, A 2014, 'Decentralized access control with anonymous authentication of data stored in clouds', *IEEE transactions on parallel and distributed systems*, vol. 25, no. 2, pp. 384-94.
Sahai, A & Waters, B 2005, 'Fuzzy identity-based encryption', in *Eurocrypt*: *proceedings of theEurocrypt* Springer, pp. 457-73.
Sahi, A, Lai, D & Li, Y 2015, 'Parallel encryption mode for probabilistic scheme to secure data in the cloud', in *Proc. 10th Int. Conf. Inf. Technol. Appl.(ICITA)*: *proceedings of theProc. 10th Int. Conf. Inf. Technol. Appl.(ICITA)*.
Sahi, A, Lai, D & Li, Y 2016, 'Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan', *Computers in biology and medicine*, vol. 78, pp. 1-8.
Sahi, A, Lai, D, Li, Y & Diykh, M 2017, 'An Efficient DDoS TCP Flood Attack Detection and Prevention System in a Cloud Environment', *IEEE Access*, vol. 5, pp. 6036-48.
Sajid, A & Abbas, H 2016, 'Data privacy in cloud-assisted healthcare systems: state of the art and future challenges', *Journal of medical systems*, vol. 40, no. 6, p. 155.
Sánchez, R, Almenares, F, Arias, P, Díaz-Sánchez, D & Marín, A 2012, 'Enhancing privacy and dynamic federation in IdM for consumer cloud computing', *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1.
Sharma, G & Kalra, S 2016, 'Identity based secure authentication scheme based on quantum key distribution for cloud computing', *Peer-to-Peer Networking and Applications*, pp. 1-15.
Son, J, Kim, J-D, Na, H-S & Baik, D-K 2016, 'Dynamic access control model for privacy preserving personalized healthcare in cloud environment', *Technology and Health Care*, vol. 24, no. s1, pp. S123-S9.
Song, W, Wang, B, Wang, Q, Peng, Z, Lou, W & Cui, Y 2017, 'A privacy-preserved full-text retrieval algorithm over encrypted data for cloud storage applications', *Journal of Parallel and Distributed Computing*, vol. 99, pp. 14-27.
Su, J-S, Cao, D, Wang, X-F, Sun, Y-P & Hu, Q-L 2011, 'Attribute based encryption schemes', *Journal of Software*, vol. 22, no. 6, pp. 1299-315.
Subashini, S & Kavitha, V 2011, 'A survey on security issues in service delivery models of cloud computing', *Journal of network and computer applications*, vol. 34, no. 1, pp. 1-11.
System for cross-domain identity management, 2017, <http://www.simplecloud.info/>.
Tang, J, Cui, Y, Li, Q, Ren, K, Liu, J & Buyya, R 2016, 'Ensuring security and privacy preservation for cloud data services', *ACM Computing Surveys (CSUR)*, vol. 49, no. 1, p. 13.
Thilakanathan, D, Chen, S, Nepal, S, Calvo, R & Alem, L 2014, 'A platform for secure monitoring and sharing of generic health data in the Cloud', *Future Generation Computer Systems*, vol. 35, pp. 102-13.
Tong, Y, Sun, J, Chow, SS & Li, P 2014, 'Cloud-assisted mobile-access of health data with privacy and auditability', *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 2, pp. 419-29.
Wang, C, Xu, X, Shi, D & Fang, J 2015, 'Privacy-preserving Cloud-based Personal Health Record System Using Attribute-based Encryption and Anonymous Multi-ReceiverIdentity-based Encryption', *Informatica*, vol. 39, no. 4. (A)
Wang, S, Zhou, J, Liu, JK, Yu, J, Chen, J & Xie, W 2016, 'An efficient file hierarchy attribute-based encryption scheme in cloud computing', *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1265-77.
Wang, W, Chen, L & Zhang, Q 2015, 'Outsourcing high-dimensional healthcare data to cloud with personalized privacy preservation', *Computer networks*, vol. 88, pp. 136-48. (B)
Wang, XA, Ma, J, Xhafa, F, Zhang, M & Luo, X 2017, 'Cost-effective secure E-health cloud system using identity based cryptographic techniques', *Future Generation Computer Systems*, vol. 67, pp. 242-54.
Wang, Z, Huang, D, Zhu, Y, Li, B & Chung, C-J 2015, 'Efficient attribute-based comparable data access control', *IEEE Transactions on computers*, vol. 64, no. 12, pp. 3430-43. (C)
Wu, F, Xu, L, Kumari, S & Li, X 2015, 'A novel and provably secure biometrics-based three-factor remote authentication scheme for mobile client–server networks', *Computers & Electrical Engineering*, vol. 45, pp. 274-85.
Wu, L, Zhang, Y, Li, L & Shen, J 2016, 'Efficient and anonymous authentication scheme for wireless body area networks', *Journal of medical systems*, vol. 40, no. 6, p. 134. (A)
Wu, Y, Lu, X, Su, J & Chen, P 2016, 'An efficient searchable encryption against keyword guessing attacks for sharable electronic medical records in cloud-based system', *Journal of medical systems*, vol. 40, no. 12, p. 258. (B)
Xhafa, F, Feng, J, Zhang, Y, Chen, X & Li, J 2015, 'Privacy-aware attribute-based PHR sharing with user accountability in cloud computing', *The Journal of Supercomputing*, vol. 71, no. 5, pp. 1607-19. (A)

Xhafa, F, Li, J, Zhao, G, Li, J, Chen, X & Wong, DS 2015, 'Designing cloud-based electronic health record system with attribute-based encryption', *Multimedia Tools and Applications*, vol. 74, no. 10, pp. 3441-58. (B)
Xia, Z, Wang, X, Sun, X & Wang, Q 2016, 'A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data', *IEEE transactions on parallel and distributed systems*, vol. 27, no. 2, pp. 340-52.
Xiong, J, Yao, Z, Ma, J, Liu, X, Li, Q & Ma, J 2014, 'PRIAM: Privacy Preserving Identity and Access Management Scheme in Cloud', *KSII Transactions on Internet & Information Systems*, vol. 8, no. 1, p. 23.
Xu, J, Wen, Q, Li, W & Jin, Z 2016, 'Circuit ciphertext-policy attribute-based hybrid encryption with verifiable delegation in cloud computing', *IEEE transactions on parallel and distributed systems*, vol. 27, no. 1, pp. 119-29.
Yang, J-J, Li, J-Q & Niu, Y 2015, 'A hybrid solution for privacy preserving medical data sharing in the cloud environment', *Future Generation Computer Systems*, vol. 43, pp. 74-86.
Yang, K & Jia, X 2014, 'Expressive, efficient, and revocable data access control for multi-authority cloud storage', *IEEE transactions on parallel and distributed systems*, vol. 25, no. 7, pp. 1735-44.
Yang, Y & Ma, M 2016, 'Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds', *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 746-59.
Yang, Y 2015, 'Attribute-based data retrieval with semantic keyword search for e-health cloud', *Journal of Cloud Computing*, vol. 4, no. 1, p. 10.
Yeh, H-L, Chen, T-H, Hu, K-J & Shih, W-K 2013, 'Robust elliptic curve cryptography-based three factor user authentication providing privacy of biometric data', *IET Information Security*, vol. 7, no. 3, pp. 247-52.
Younis, YA, Kifayat, K & Merabti, M 2014, 'An access control model for cloud computing', *Journal of Information Security and Applications*, vol. 19, no. 1, pp. 45-60.
Yu, S, Wang, C, Ren, K & Lou, W 2010, 'Achieving secure, scalable, and fine-grained data access control in cloud computing', in *Infocom, 2010 proceedings IEEE*: *proceedings of theInfocom, 2010 proceedings IEEE* Ieee, pp. 1-9.
Yu, Y, Miyaji, A, Au, MH & Susilo, W 2017, *Cloud computing security and privacy: Standards and regulations*, Elsevier, 0920-5489.
Yüksel, B, Küpçü, A & Özkasap, Ö 2017, 'Research issues for privacy and security of electronic health services', *Future Generation Computer Systems*, vol. 68, pp. 1-13.
Zhao, F, Li, C & Liu, CF 2014, 'A cloud computing security solution based on fully homomorphic encryption', in *Advanced Communication Technology (ICACT), 2014 16th International Conference on*: *proceedings of theAdvanced Communication Technology (ICACT), 2014 16th International Conference on* IEEE, pp. 485-8.

Computers in Biology and Medicine 78 (2016) 1–8

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/cbm

Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan

Aqeel Sahi [a,b,*], David Lai [a], Yan Li [a]

[a] Department of Math and Computing, University of Southern Queensland, 487/521-535 West St, Darling Heights, QLD 4350, Australia
[b] Computer Center, University of ThiQar, ThiQar, Iraq

ARTICLE INFO

Article history:
Received 27 June 2016
Received in revised form
9 September 2016
Accepted 9 September 2016

Keywords:
Disaster recovery plan
eHealth cloud
EHR
PHR
Privacy
Security

ABSTRACT

Cloud computing was introduced as an alternative storage and computing model in the health sector as well as other sectors to handle large amounts of data. Many healthcare companies have moved their electronic data to the cloud in order to reduce in-house storage, IT development and maintenance costs. However, storing the healthcare records in a third-party server may cause serious storage, security and privacy issues. Therefore, many approaches have been proposed to preserve security as well as privacy in cloud computing projects. Cryptographic-based approaches were presented as one of the best ways to ensure the security and privacy of healthcare data in the cloud. Nevertheless, the cryptographic-based approaches which are used to transfer health records safely remain vulnerable regarding security, privacy, or the lack of any disaster recovery strategy. In this paper, we review the related work on security and privacy preserving as well as disaster recovery in the eHealth cloud domain. Then we propose two approaches, the Security-Preserving approach and the Privacy-Preserving approach, and a disaster recovery plan. The Security-Preserving approach is a robust means of ensuring the security and integrity of Electronic Health Records, and the Privacy-Preserving approach is an efficient authentication approach which protects the privacy of Personal Health Records. Finally, we discuss how the integrated approaches and the disaster recovery plan can ensure the reliability and security of cloud projects.

1. Introduction

The technologies of Cloud Computing (CC) provide simple and easy on-demand network access to a shared group of computing resources, which are simple to install and maintain with minimal effort. They have become an important technology milestone and many scientists and researchers claim that cloud computing has changed the computing processes and IT markets. When access is powered by cloud computing, users can use comprehensive sets of tools for assessing various applications, storage and platforms through the Internet, as well as using the services offered by cloud producers.

The National Institute of Standards and Technology (NIST) stated that CC is a model for using computer resources and other modern technological functionality in the information technology world to provide services such as storage and applications [1]. Users can access and use cloud computing services without the need to acquire knowledge, expertise or even administration of infrastructures that support these services. There are three main types of services offered by the cloud [2]: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [3]. In addition, four deployment models have been identified for cloud architecture solutions, namely Private cloud; Community cloud; Public cloud and Hybrid cloud [4,5]. Since cloud computing shares resources distributed throughout the Internet and among the intranets, security is therefore an important issue.

CC can be applied in many sectors; in this research we will focus on the health sector. CC gives healthcare providers the ability to diagnose and evaluate a patient's health even though the patient is not at the hospital. This requires the patient's medical data to be distributed among physicians, nurses, insurance companies as well as the data owner to deliver flexible and prompt care to patients. The distribution of the patient's medical data raises security and privacy issues. Therefore, in this research we propose two approaches which we designed to preserve privacy and security in the eHealth cloud.

Before we present the approaches, we will define Personal Health Records (PHRs) and Electronic Health Records (EHRs).

* Corresponding author at: Department of Math and Computing, University of Southern Queensland, 487/521-535 West St, Darling Heights, QLD 4350, Australia.
E-mail address: akeel_sahy@yahoo.co.uk (A. Sahi).

http://dx.doi.org/10.1016/j.compbiomed.2016.09.003
0010-4825/

1.1. Personal Health Records (PHRs)

A PHR is medical data owned and managed by the patient himself [6]. A good PHR contains a precise and short record of the patient's medical history data collected from various sources (such as EHRs). These data can be easily reached by everyone having the required authorizations to display the PHR.

1.2. Electronic Health Records (EHRs)

According to Zapata [7], the International Organization for Standardization (ISO) stated that the EHR is a "repository of information regarding the health status of a subject of care, in computer processable form". Using the words of ISO/TS 18308 [8], the main purpose of an EHR is to deliver a registered record of healthcare which supports current and upcoming healthcare received by the patient from various healthcare providers.

Today, cloud computing is used by millions of people around the world. Cloud computing gives users the opportunity to store data in the cloud for easy access anytime and anywhere [9]. However, in the cloud environment, the user's data are controlled by service providers and not by the users themselves. The potential for data leaks is real, either intentionally or accidentally, which is unacceptable [10]. Common problems of security and privacy in the eHealth cloud include: confidentiality, integrity, authentication, access control, and non-repudiation [11].

The overall cloud computing theme is that we should be able to use all kinds of functionality and services provided by the cloud, but we would like to maintain our data privacy and security. Therefore, issues regarding privacy and security of data are the main factors limiting the widespread use of cloud computing. Much research has been done on these issues and several researchers claim that a versatile cryptography system may handle data security and data privacy issues more effectively than other methods [12]. With the use of cryptography systems, our proposed security and privacy approaches can tackle the security and privacy issues. Although security is the most important factor in any cloud project, disaster recovery (DR) planning needs to be considered also. There are three types of disaster that may cause major damage to any system: (1) natural disasters, such as flood, earthquake, and volcano, (2) man-made disasters, such as cybercrime and technological terrorism, and (3) technological accidents, such as infrastructure failure, and transportation failure [13]. In order to overcome these kinds of disaster, a recovery plan needs to be set up.

2. Related work

In this section, we review the related work on the preservation of security and privacy in the eHealth cloud as well as the DR planning.

Users are using CC in various ways, such as checking email by Yahoo, writing documents by Google Docs, and storing data in iCloud. CC delivers numerous benefits, for example, low costs due to the pay-as-you-go model, extraordinary availability as data is commonly distributed between a number of servers, and load balancing [1]. Furthermore, CC is benefiting health organisations [14].

Health organisations have been quick to move to CC for the obvious advantages of data storage and sharing. Those organisations are keen to store and share PHRs and EHRs using the cloud, thereby eliminating the geographical boundaries between health organisations and patients [15]. Sharing data using CC has rapidly become a very important component for healthcare providers and many other organisations. According to Thilakanathan et al. [16], for most organisations, the percentage of the data shared with clients using CC is about 74% and with dealers is about 64%.

On the other hand, it is important for medical data to be safe from unauthorized access and unwanted modifications. CC, however, is vulnerable to various security and privacy attacks. Consequently, many healthcare providers are unwilling to implement CC technologies, as a patient's information privacy may be breached. According to Van et al. [17], the main hurdle delaying the growth and extensive acceptance of CC is privacy and security issues. Actually, most privacy and security attacks are caused by the Cloud Service Providers (CSP) themselves [18] as they commonly have access to the Cloud Storage (CS) and they may also sell the data records to gain profits. Indeed, insider attacks are one of the main problems related to CC, as pointed out by El-Gazzar et al. and Pasupuleti et al. [19,20].

Fujisaki et al. proposed a PKE-based (Public Key Encryption) approach named RSA-OAEP [21]; however, PKE-based approaches are computationally inefficient because of the larger key size and the slower operation.

Jafari et al. introduced an approach which gives the patient the possibility of controlling his EHRs. This approach limits the patient to managing records authored by other parties, such as physicians and nurses [22]. On the other hand, the cloud service provider cannot retrieve the records in plaintext format. The patient himself and data consumers are given the private and public keys for encryption and decryption [23].

Another approach presented by Zhang et al. [24] is a time-based approach. The approach is efficient in ensuring the privacy of the EHRs at the cloud storage and enhances the operation of key distribution between trusted parties. This approach adopts time-bound hierarchical key management [25]. Time-bound hierarchical key management permits trusted parties to gain short-term access to the EHRs, which are encrypted using Symmetric Key Encryption (SKE). However, Zhang's approach is logically inadequate due to the fact that users have to take on multiple roles. Therefore, the users are required to hold and control multiple keys.

Tran et al. [26] proposed an approach based on the proxy re-encryption idea. A trusted user can obtain a data record as the proxy will convert the encrypted data on the data owner side to differently encrypted data which can be decrypted to plain text by the trusted user's key. However, because Tran's approach uses ElGamal public key cryptography, the encryption or decryption of very large data is not practical and unfortunately, very large data is a feature of medical data [16]. In addition, this approach does not solve the situation where a revoked party re-joins using another access key.

Tu et al. presented an approach which adopts Ciphertext-Policy Attribute-Based Encryption (CP-ABE) for data sharing in the situation of a revocation operation, which at the same time allows high flexibility, access control and revocation [27]. However, this approach is also not effective when considering very large data [16].

An Access Control technique is a policy or rule that allows the restriction of access to a cloud project [28]. It also detects unauthorized users who try to access a cloud project. Access Control allows one application to trust the identity of another application [29]. While a robust authentication technique is a compulsory requirement for any cloud project, access control cannot secure data at rest and in transit [30,31], and it is not satisfactory enough to achieve privacy for PHRs [32]. Encryption methods are definitely a better choice for protecting data at rest, as well as the choice for protecting data in transit [30]. In addition, cryptography offers an integrity check to verify that the data is not compromised or corrupted in transit.

Wood et al. presented a DR plan which utilizes three servers and one database, as shown in Fig. 1. One of the servers is

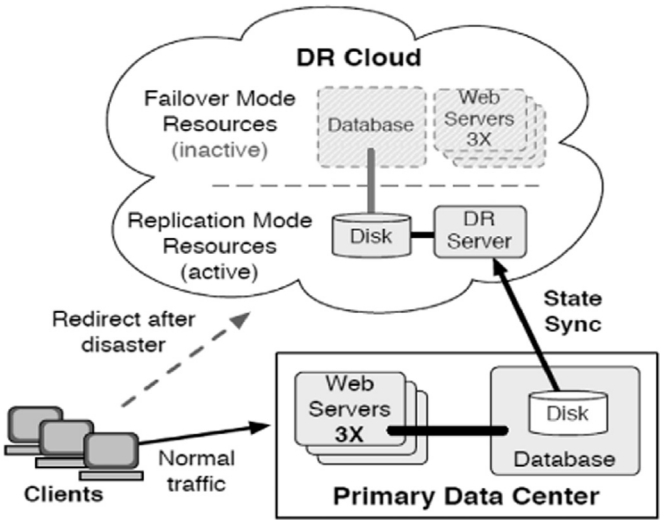


Fig. 1. Wood et al. DR plan [33].

nominated to be used in the event of a disaster, and all users are redirected to that server during the disaster [33]. However, the authors did not consider the case in which the disaster also affects the nominated server itself. In addition, the redirecting may influence the performance of the system and all users will be disconnected while they are being redirected to the nominated server.

In this paper, we are concerned with the following issues when strengthening the security and privacy of health records in the cloud:

- How to design compatible security and privacy approaches which can preserve the security of the EHR and the privacy of the PHR at the same time in the eHealth cloud.
  - How the proposed approaches perform in emergency situations.
  - How to revoke old session keys.
  - How to ensure the availability and the continuity of the system during a disaster.

From our point of view, security, privacy, and disaster recovery plans are crucial and need to be designed together to be homogeneous, accurate, and easy to implement. The lack of any one of them will certainly affect the performance of the others. Thus, designing one of them only, as in most of the related studies, is not enough to deal with the real world.

To build a better cryptography cloud project, four requirements need to be satisfied: authentication, non-repudiation, integrity, and confidentiality [4]. Most of the previous studies focused on either the security of the EHR or the privacy of PHR, which is not enough to achieve the requirements. Many researchers proposed good security approaches to ensure confidentiality, while others proposed good privacy approaches to ensure authentication. However, those approaches may not integrate seamlessly. Besides, non-repudiation and integrity need to be provided by the same system. Therefore, we were motivated to propose approaches that can be easily implemented and integrated in any distributed system to cover the requirements for both security and privacy. Moreover, on top of security of the EHR and privacy of the PHR, we designed our disaster recovery plan to guarantee the availability and the continuity of the system during a time of disaster. While disaster recovery is not considered by most of the studies in the eHealth domain, our approaches and the disaster recovery plan will enable data owners and patients to have full and safe control over their records.

In addition, our contributions:

- Proposed a security-preserving approach which can ensure the security and integrity of Electronic Health Records.
- Proposed a privacy-preserving approach which can ensure the privacy of Personal Health Records.
- Provided a break-glass access feature to be used in emergency situations. A revocation feature is also provided.
- Designed our disaster recovery plan to guarantee the availability and the continuity of the system during a disaster.

3. Preliminaries

3.1. PEM-AES

The Parallel Encryption Mode (PEM) was first introduced by Sahi et al. [34] as a block cipher mode of operation. The PEM adopted the Advanced Encryption Standards (AES) as an encryption algorithm. The PEM mode significantly enhances the encryption process in terms of speed and provides a data integrity check. In the PEM, each block uses the hash value of the shared data to ensure that the key stream has a very good randomness. Fast parallel processes and integrity checks are the reasons for choosing the PEM-AES. Fig. 2. together with Eqs. (1)–(5) briefly describe the process of the algorithm.

Encryption:

$$W = E_{(K \oplus IV)}(H(X_i)) \tag{1}$$

$$Y_i = E_{(K \oplus IV) \oplus H(X_i)}(X_i) \tag{2}$$

$$C_i = W + Y_i \tag{3}$$

Decryption:

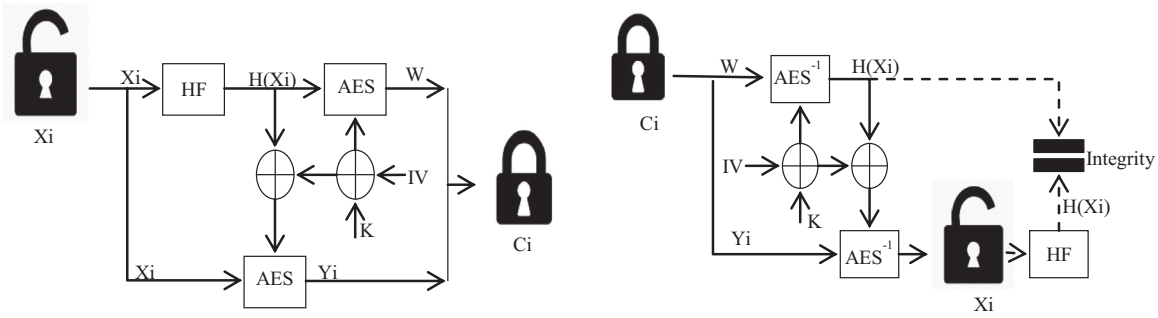


Fig. 2. PEM-AES processes [34].

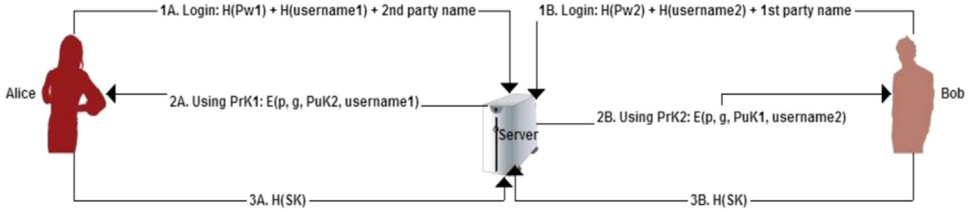


Fig. 3. Key exchange protocol [35].

$$H(X_i)=D_{(K\oplus IV)}(W) \quad (4)$$

$$X_i = D_{H(X_i)\oplus(K\oplus IV)}(Y_i) \quad (5)$$

3.2. Key exchange protocol

In order to ensure the privacy of the users, we adopt a three-party password-based authenticated key exchange protocol (3PAKE), which was introduced in [35]. This protocol ensures both authentication and non-repudiation features. A Geffe generator is used to produce a pseudorandom binary sequence. The resulting sequence is tested using statistical tests including a frequency test, serial test and poker test. Private keys are then generated from the success sequence. In this protocol, data will have a non-repudiation property and no clear data will be sent via the channel (Fig. 3).

4. The proposed approaches

While CC is facing numerous privacy and security issues, and despite the fact that most of these issues are caused by CSPs themselves, all data in CSPs and CSs must be encrypted. In other words, all medical records including PHRs and EHRs must be encrypted before CC stores and shares it. This will resist any attacks from outsiders as well as from insiders (the CSPs themselves) trying to obtain any valuable data without permission [35]. The patient's privacy needs to be ensured on top of the security for medical records.

The proposed approaches are shown in Figs. 4 and 5. They consist of data consumers, trusted party, patients, and cloud, which are defined as follows:

Data consumers: Data consumers are people or companies which are interested in using PHR or EHR data. In other words, data consumers are the healthcare providers, including physicians and nurses.

Controller: The controller is responsible for negotiating and generating session keys in order for them to be used by parties.

Data owner: The data owner is the owner of the data in our system, and is the only party who has full access to the EHR data.

Trusted party: The controller and data owner are trusted by all parties in our proposed approaches.

Patients: The patient is the owner of his PHR and has complete control over the privacy of his PHR information. He can delegate his patient role to other parties, such as a family member or friend in order to access his PHR in an emergency situation.

Cloud: The cloud consists of the cloud service provider and cloud storage. The cloud service provider responds to the demands from the data consumers and provides corresponding services. The cloud storage is used to store the shared encrypted data from the data owner.

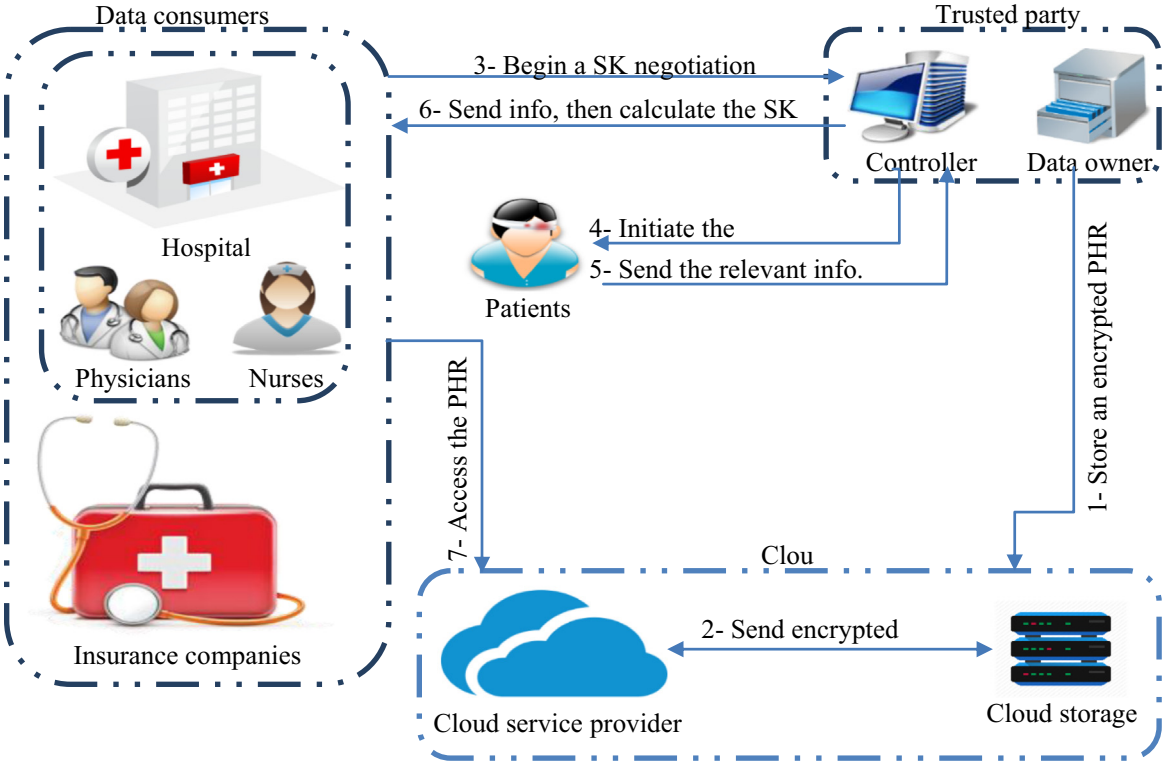


Fig. 4. Privacy-preserving approach.

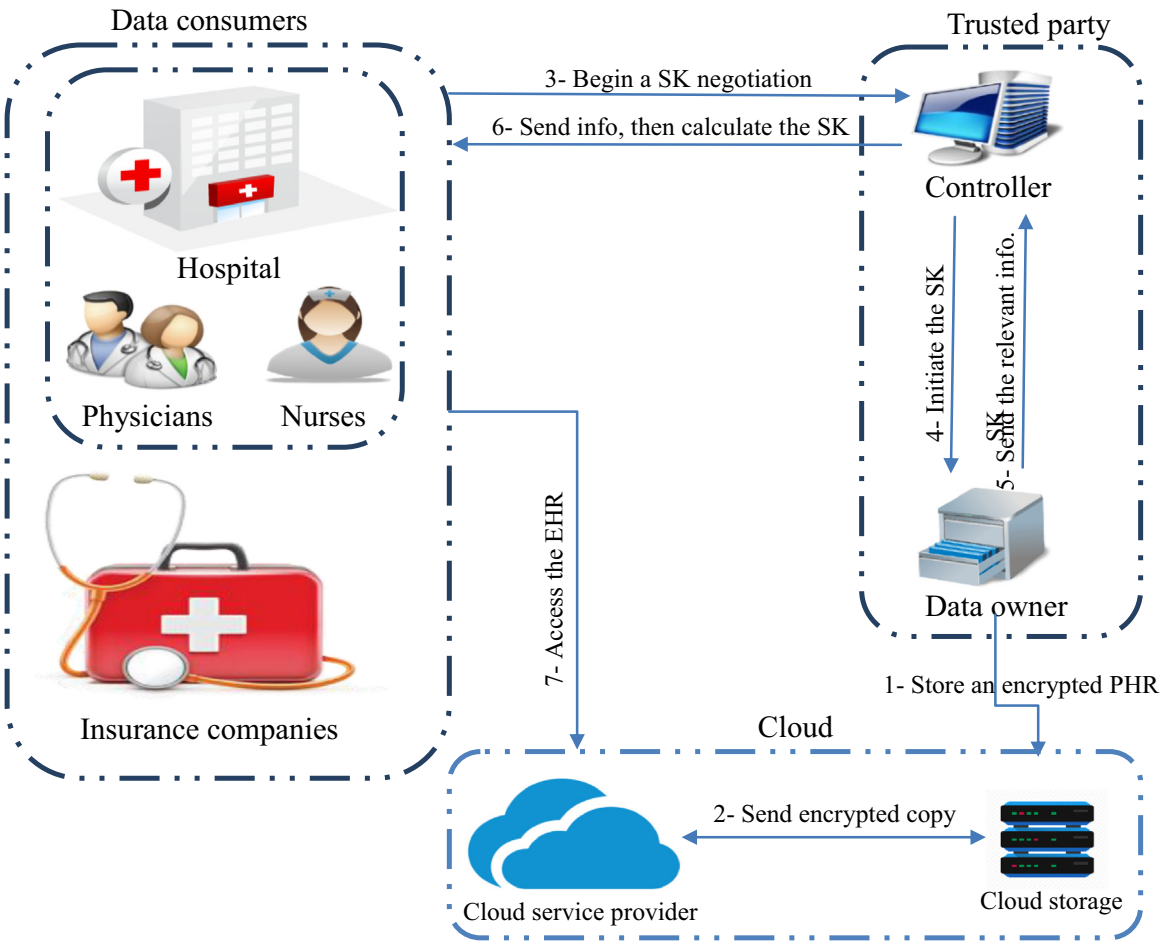


Fig. 5. Security-preserving approach.

To achieve the goals of the proposed approaches, two main points must be satisfied. Since the patient is the only one with full control over the access to his PHR information, the patient's privacy must be ensured in the first place. Secondly, all data consumers must be able to access an up-to-date version of their EHRs in the cloud at any time and in a secure manner, hence data security must be guaranteed.

In the following sections, we will explain the privacy-preserving approach, which can be used to ensure the privacy of the PHRs in the eHealth clouds. We then describe the security-preserving approach, which can be applied to ensure the security of the EHRs in the eHealth clouds.

4.1. *Privacy-preserving approach*

Authentication is vital for archiving and retrieving information from PHRs [36]. Since PHRs are controlled by patients themselves [6], an efficient authentication approach which ensures the privacy of PHRs is required.

In order to ensure the PHRs' privacy, we have adopted a three-party password-based authenticated key exchange protocol (3PAKE) based on the computational Diffie-Hellman assumption by Khader and Lai [35].

According to the 3PAKE protocol, the primitive root (p) and the generator (g) should be changed in each communication session [35]. This suits our approach to ensure that the patient is the only one who has complete access to his PHR, and all data consumers will be revoked after the session. Otherwise, if p and g are not changed, data consumers will be able to access the patient's PHRs using an old session key.

The privacy-preserving approach works as follows:

1. The data owner stores an encrypted PHR at the Cloud Storage (CS). The data are stored according to the disaster recovery plan in Section 6.
2. Therefore, the Cloud Service Provider (CSP) has a copy of the PHR. However, this copy is encrypted and the privacy of the PHR is secured. To make any modification to the PHR, the patient's permission is required. Data are retrieved according to the disaster recovery plan in Section 6.
3. Data consumers ask the controller to begin a Session Key (SK) negotiation in order to access the PHR.
4. The controller will control the communication between the data consumers and the patient. In this step, the controller will ask the patient to initiate the SK agreement.
5. The patient will then send the relevant information through the channel back to the controller in order to authorize the data consumer to gain access to his PHR.
6. The data consumer calculates the SK using the information received from the controller.
7. Data consumers can access the PHR at the CSP once they get the permissions.

4.2. *Security-preserving approach*

EHRs are managed by healthcare providers [6,37] (in our system, healthcare providers are data consumers). However, patients may have to follow various policies, such as medical, dental, and vision, registered with different insurance companies which make

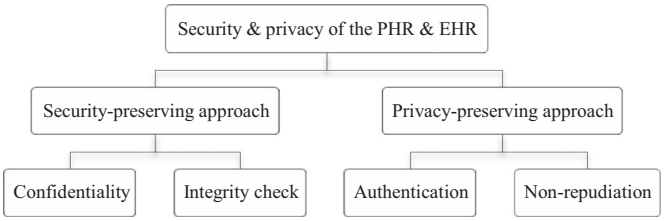


Fig. 6. Security and privacy of the PHR and the HER.

it hard for all parties to access up-to-date EHR records every time. Therefore, we have shifted the management of the data to the data owner or his delegate to ensure that all the consumers can access an up-to-date version of the EHR in the cloud at any time and in a secure manner.

The data owner stores the EHRs at their own preferred CC storage servers, and cloud projects must retrieve the EHRs from these servers. Therefore a robust approach which can ensure the security of the EHRs is required.

The AES encryption algorithm with the PEM block cipher mode was adopted to secure the data at the data owner database as well as at the cloud storage [34].

The security-preserving approach works as follows:

1. The data owner stores encrypted EHRs at the Cloud Storage (CS). Data are stored according to the disaster recovery plan in Section 6.
2. Therefore, the Cloud Service Provider (CSP) has a copy of the EHRs. However, this copy is encrypted to ensure the security of the EHRs. To use the EHRs, the CSP needs the data owner's permission to do so. Data are retrieved according to the disaster recovery plan in Section 6.
3. Data consumers ask the controller to begin the Session Key (SK) negotiation in order to access the EHRs.
4. The controller will control the communication between the data consumers with the data owner. In this step, the controller will ask the data owner to initiate the SK agreement.
5. The data owner will send the relevant information through the channel back to the controller in order to authorize the data consumer to gain access to the EHR.
6. The data consumer calculates the SK using the information received from the controller.
7. Data consumers can access the EHR at the CSP once they get permission.

4.3. Break-glass access

In an emergency situation, such as in a life-threatening situation for an unconscious patient, healthcare providers may require temporary access to a patient's PHR. Those staff members must have a temporary authorisation to decipher the PHR information. While the patient is the only one in our system who has complete control of his PHR, as we mentioned earlier the patient can also delegate the role to a family member or friend. Therefore, a family member or friend can play a patient role in order to authorize the data consumers to access the patient's PHR. In our approaches this can be achieved by encouraging a patient to delegate an emergency key to a family member or friend when the patient registers for the first time in the system. He will be asked to provide an emergency contact detail for a family member or friend, and to assign an emergency key to this person.

5. Discussion

To ensure the security and the privacy of any cryptography-based system, four requirements need to be considered: authentication, non-repudiation, integrity, and confidentiality [4]. In this section, we will analyse and evaluate the security of our proposed approaches from three different perspectives: security requirements, comparison with existing work, and when under several kinds of attacks.

5.1. Security requirements

CSP should utilize a robust authentication and non-repudiation technique to guarantee authentication and non-repudiation. According to Khader and Lai in [35], these two features are ensured by the 3PKAE key exchange protocol. Since we employed the protocol in this paper, our proposed privacy-preserving approach inherited the authentication and non-repudiation features from the adopted 3PAKE protocol.

In addition, the PEM-AES can check the integrity of the message by comparing the hash of the copy of the original data with the hash received from the other party. It has a concept similar to Chaining Blocks Cipher (CBC) in terms of confidentiality. It connects all the blocks by the hash value of the plaintext. Hence, if one cipher block is corrupted, then the corresponding plaintext block will also be corrupted. Since the CBC mode is considered to be secure against many attacks, we can claim that the PEM-AES mode is also secure against these attacks. In addition, it provides integrity on top of the confidentiality [34]. As we adopted the PEM-AES in our approaches, they too can provide these two features.

Fig. 6 illustrates the security and privacy features that can be provided by combining the privacy-preserving approach with the security-preserving approach in the cloud environment.

5.2. Comparison with existing work

In this section, we compare our proposed approaches with several existing approaches in terms of security, privacy, revocation, break-glass access and DR plan.

Table 1 shows the comparison. As shown in the table, our proposed approaches together with the disaster recovery plan achieved all the listed features, whereas some vital features are missing in other approaches. On top of that, some of those approaches have limitations in various aspects. In the RBTBAC approach [24], each user is required to hold and control multiple keys, which is logically inadequate. The approach by Tran et al. and the new CP-ABPRE approach [26,27] are not effective when considering very large data, and unfortunately very large data is a feature of medical data [16]. Furthermore, the approach by Fabian et al. is not well-suited for emergency cases, as stated by the authors themselves [39]. Finally, the approach of Chen et al. gives the ability of managing the PHR to every healthcare staff member in addition to patients and doctors; yet the PHR needs to remain safe at all times and be able to be managed by the patient himself and his doctor (even for the doctor, a revocation feature is needed to cancel his managing right later) [40].

Table 1
Comparison of delivered security features.

	Proposed approaches	Security	Privacy	Revocation	Break-glass	DR plan
1	Jafari et al. [22]	×	√	×	√	×
2	RBTBAC [24]	√	√	×	×	×
3	Tran et al. [26]	√	×	√	×	×
4	New CP-ABPRE [27]	√	×	√	×	×
5	Wang et al. [38]	×	√	×	×	×
6	Fabian et al. [39]	√	√	×	×	×
7	Chen et al. [40]	×	√	√	×	×
8	CP-ABE [41]	√	√	×	×	×
9	Zheng [42]	×	√	√	×	×
10	Our approaches	√	√	√	√	√

5.3. When under attack

When the PHR and EHR are under different attacks, we will show how the proposed approaches can mitigate the effects of some of these attacks.

5.3.1. Outside attacks

In outside attacks, an adversary attempts to gain electronic records without any identity or access credentials. Let us assume that he can compromise the communication channels and is able to retrieve the PHR/EHR records. However, he can only get the PHR/EHR records in an encrypted format, which he cannot decrypt without an appropriate key. While keys are held only by the patient and the controller, obtaining the correct key is impossible. Thus, we can claim that our approaches can prevent outside attacks.

5.3.2. Man-in-the-middle attacks

In our proposed approaches, it is impossible for an adversary to perform a man in the middle attack. Let us assume that the adversary can compromise the communication channels and retrieve any information sent through this channel. Nevertheless, the adversary needs to know at least the public keys to perform a man in the middle attack, whereas these public keys are encrypted using private user keys according to the adopted 3PAKE protocol [35]. In the 3PAKE protocol, only the user (either the patient or controller) himself can know the public and private keys, so he is the only one who can decrypt the electronic records. Thus, the proposed approaches can prevent man-in-the-middle attacks.

5.3.3. Offline and online dictionary attacks

The 3PAKE protocol can also be secure against offline and online dictionary attacks. For an adversary to perform an offline dictionary attack, he needs to guess the identity credential of the user to calculate the session key. This is going to fail during the verification process due to the fact that the session key is required for verification of the identity credential in advance.

An integrity check is provided in this protocol, and it helps to prevent on-line dictionary attacks using checksum validations. As parties are required to validate one another using a stored checksum, online dictionary attacks can thus be prevented.

6. Disaster recovery plan

In the event of a disaster several questions will need to be answered. For example: do we have a backup server? Where is it? How do we reconnect the client back to the system? How long will the system take to resume service? Therefore, a DR plan must be prepared to answer these and other questions.

We will present our DR plan. First of all, data such as PHRs and EHRs are stored in the cloud storage, which includes a minimum of three nodes or data centres. At the beginning of the process, the data owner will send a heartbeat signal (a heartbeat is a signal sent between the data owner, nodes, and the CSP to check whether those nodes are still in working condition or not) to check the status of the nodes. After that, the data owner asks the controller to break the data records into three partitions (three in our example, but it could be varied), distribute and store them among the nodes. When a client attempts to use the data through the CSP, the CSP will send a heartbeat signal to the nodes, and then ask the controller to retrieve the three partitions from the nodes and combine them in one record file.

The controller is responsible for the number of partitions and the size of each of them. Each partition must be stored in several nodes (in our case, three nodes). The three nodes must be located at three different physical locations in order to ensure that the cloud project can be continued even in a disaster situation.

For example, let us assume that the area where the second node is located has an earthquake, which causes complete damage to this node. The original record can be retrieved immediately from node 1 and node 3 with no time wasted or client disconnected. Therefore, this DR plan can answer the above questions and ensure the continuity of the system, as shown in Fig. 7.

7. Conclusion

The security and privacy of data are the most important factors in the cloud. Therefore, this research study aims to set up new approaches to harden the security of the EHRs and the privacy of the PHRs in cloud projects, as well as ensuring the continuity of the projects during a time of disaster. A privacy-preserving approach was proposed based on a key exchange protocol, which can ensure the privacy of the PHRs. Also, a security-preserving approach was proposed based on the PEM-AES, which can ensure the security of the EHRs. These two approaches are able to provide the most important requirements in each cryptosystem, such as authentication, non-repudiation, integrity, and confidentiality. In addition, a DR plan was proposed to enable a client to connect to its system at any time, even at a time of disaster. The two approaches and the DR plan all together provide a private, secure and robust CC environment for the health sector.

Conflicts of interest

None declared.

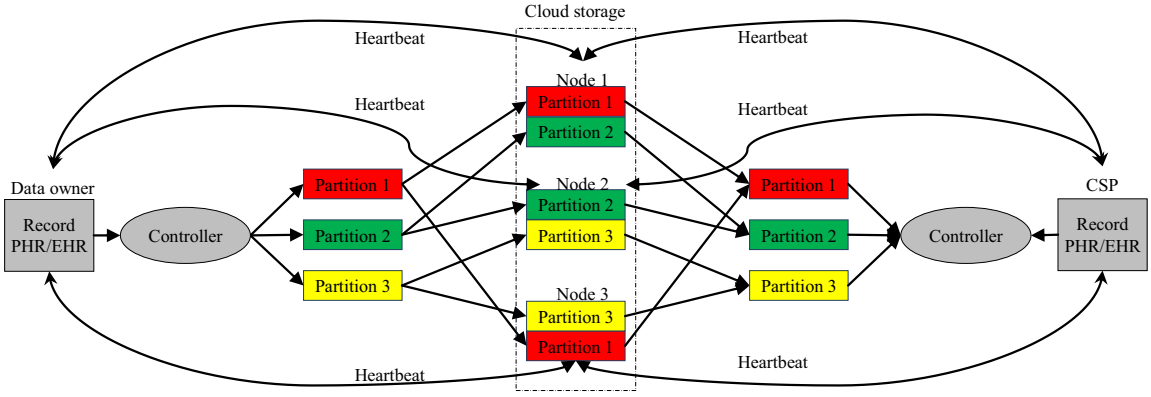


Fig. 7. The proposed DR plan.

Acknowledgements

This research did not receive any specific Grant from funding agencies in the public, commercial, or not-for-profit sectors. The constructive comments from the anonymous reviewers are greatly appreciated. Professor Barbara Harmes is gratefully acknowledged for her help and support.

References

[1] P. Mell, T. Grance, The NIST Definition of Cloud Computing, 2011.
[2] M. Sugumaran, B.B. Murugan, D. Kamalraj, An architecture for data security in cloud computing, in: Proceedings of the 2014 World Congress on Computing and Communication Technologies (WCCCT), 2014, pp. 252–255.
[3] K.E. Kushida, J. Murray, J. Zysman, Cloud Computing: From Scarcity to Abundance, BRIE Working Paper, Springer, 2014.
[4] D. Zissis, D. Lekkas, Addressing cloud computing security issues, Future Gener. Comput. Syst. 28 (2012) 583–592.
[5] I. Hsu, F.Q. Cheng, SAaaS: a cloud computing service model using semantic-based agent, Expert Syst. 32 (2013) 77–93.
[6] A. Abbas, S.U. Khan, A review on the state-of-the-art privacy-preserving approaches in the e-health clouds, IEEE J. Biomed. Health Inform. 18 (2014) 1431–1441.
[7] B.C. Zapata, A.H. Niñirola, A. Idri, J.L. Fernández-Alemán, A. Toval, Mobile PHRs compliance with Android and iOS usability guidelines, J. Med. Syst. 38 (2014) 1–16.
[8] ANSI, ISO, TS 18308 Health Informatics-Requirements for an Electronic Health Record Architecture, ISO (Ed.), 2003.
[9] M. Carroll, A. Van Der Merwe, P. Kotze, Secure cloud computing: benefits, risks and controls, in: Proceedings of Information Security South Africa (ISSA), 2011, pp. 1–9.
[10] N. Gonzalez, C. Miers, F. Redígolo, M. Simplicio, T. Carvalho, M. Näslund, et al., A quantitative analysis of current security concerns and solutions for cloud computing, J. Cloud Comput. 1 (2012) 1–18.
[11] D.G. Rosado, R. Gómez, D. Mellado, E. Fernández-Medina, Security analysis in the migration to cloud environments, Future Internet 4 (2012) 469–487.
[12] D. Talbot, Security in the Ether, Technol. Rev. 113 (2010) 36–42.
[13] S. Snedaker, Business Continuity and Disaster Recovery Planning for IT Professionals, Newnes, 2013.
[14] E.J. Giniat, Cloud computing: innovating the business of health care, Healthc. Financ. Manag.: J. Healthc. Financ. Manag. Assoc. 65 (2011) 130–131.
[15] R. Wu, Secure Sharing of Electronic Medical Records in Cloud Computing, Arizona State University, 2012.
[16] D. Thilakanathan, S. Chen, S. Nepal, R. Calvo, L. Alem, A platform for secure monitoring and sharing of generic health data in the cloud, Future Gener. Comput. Syst. 35 (2014) 102–113.
[17] P. Van Gorp, M. Comuzzi, A. Jahnen, U. Kaymak, B. Middleton, An open platform for personal health record apps with platform-level privacy protection, Comput. Biol. Med. 51 (2014) 14–23.
[18] F. Rocha, S. Abreu, M. Correia, The final frontier: confidentiality and privacy in the cloud, Computer 44 (9) (2011) 44–50.
[19] R. El-Gazzar, E. Hustad, D.H. Olsen, Understanding cloud computing adoption issues: a Delphi study approach, J. Syst. Softw. 118 (2016) 64–84.
[20] S.K. Pasupuleti, S. Ramalingam, R. Buyya, An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing, J. Netw. Comput. Appl. 64 (2016) 12–22.
[21] E. Fujisaki, T. Okamoto, D. Pointcheval, J. Stern, RSA-OAEP is secure under the RSA assumption, J. Cryptol. 17 (2004) 81–104.
[22] M. Jafari, R. Safavi-Naini, N.P. Sheppard, A rights management approach to protection of privacy in a cloud of electronic health records, in: Proceedings of the 11th Annual ACM Workshop on Digital Rights Management, 2011, pp. 23–30.
[23] I. Khalil, A. Khreishah, M. Azeem, Consolidated identity management system for secure mobile cloud computing, Comput. Netw. 65 (2014) 99–110.
[24] R. Zhang, L. Liu, R. Xue, Role-based and time-bound access and management of EHR data, Secur. Commun. Netw. 7 (2014) 994–1015.
[25] E. Bertino, N. Shang, S.S. Wagstaff Jr., An efficient time-bound hierarchical key management scheme for secure broadcasting, IEEE Trans. Dependable Secur. Comput. 5 (2008) 65–70.
[26] D.H. Tran, H.-L. Nguyen, W. Zha, W.K. Ng, Towards security in sharing data on cloud-based social networks, in: Proceedings of the 2011 8th International Conference On Information, Communications and Signal Processing (ICICS), 2011, pp. 1–5.
[27] K. Liang, M.H. Au, J.K. Liu, W. Susilo, D.S. Wong, G. Yang, Y. Yu, A. Yang, A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing, Future Gener. Comput. Syst. 52 (2015) 95–108.
[28] A.R. Khan, Access control in cloud computing environment, ARPN J. Eng. Appl. Sci. 7 (2012) 613–615.
[29] B. Sosinsky, in: illustrated (Ed.), Cloud Computing Bible, 762, John Wiley & Sons, United States of America, 2010.
[30] J. Sen, Security and privacy issues in cloud computing, Archit. Protoc. Secur. Inf. Technol. Infrastruct. (2013) 1–45.
[31] Y.A. Younis, K. Kifayat, M. Merabti, An access control model for cloud computing, J. Inf. Secur. Appl. 19 (2014) 45–60.
[32] Y. Yang, H. Zhu, H. Lu, J. Weng, Y. Zhang, K.-K.R. Choo, Cloud based data sharing with fine-grained proxy re-encryption, Pervasive Mob. Comput. 28 (2015) 122–134.
[33] T. Wood, E. Cecchet, K.K. Ramakrishnan, P.J. Shenoy, J.E. van der Merwe, A. Venkataramani, Disaster recovery as a cloud service: economic benefits & deployment challenges, in: Proceedings of the HotCloud, vol. 10, 2010, pp. 8–15.
[34] A. Sahi, D. Lai, Y. Li, Parallel encryption mode for probabilistic scheme to secure data in the Cloud, in: Proceedings of the 10th International Conference on Information Technology and Applications (ICITA), Sydney, 2015.
[35] A. S. Khader, D. Lai, Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol, in: Proceedings of the 22nd International Conference on Telecommunications (ICT), 2015, pp. 204–208.
[36] D.C. Kaelber, A.K. Jha, D. Johnston, B. Middleton, D.W. Bates, A research agenda for personal health records (PHRs), J. Am. Med. Inform. Assoc. 15 (2008) 729–736.
[37] L.-C. Huang, H.-C. Chu, C.-Y. Lien, C.-H. Hsiao, T. Kao, Privacy preservation and information security protection for patients' portable electronic health records, Comput. Biol. Med. 39 (2009) 743–750.
[38] C. Wang, X. Liu, W. Li, Implementing a personal health record Cloud platform using ciphertext-policy attribute-based encryption, in: Proceedings of Intelligent Networking and Collaborative Systems (INCoS), 2012, pp. 8–14.
[39] B. Fabian, T. Ermakova, P. Junghanns, Collaborative and secure sharing of healthcare data in multi-clouds, Inf. Syst. 48 (2015) 132–150.
[40] T.-S. Chen, C.-H. Liu, T.-L. Chen, C.-S. Chen, J.-G. Bau, T.-C. Lin, Secure dynamic access control scheme of PHR in cloud computing, J. Med. Syst. 36 (2012) 4005–4020.
[41] L. Ibraimi, M. Asim, M. Petković, Secure management of personal health records by applying attribute-based encryption, in: Proceedings of the 2009 6th International Workshop on Wearable Micro and Nano Technologies for Personalized Health (pHealth), 2009, pp. 71–74.
[42] Y. Zheng, Privacy-Preserving Personal Health Record System using Attribute-based Encryption, Worcester Polytechnic Institute, 2011.

Aqeel Sahi is a Ph.D. student in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia. He received a Bachelor degree of Computer Science from Thiqar University, Iraq in 2007, and Master degree of Information Technology from University Utara Malaysia, Malaysia in 2010. His current research interests are in cryptography and parallel processing with a focus on block cipher modes of operation and key exchange protocols.

David Lai is a senior lecturer in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia. Qualifications BSc CUHK , PGDipEd CUHK , GDipCompSc VUT , MPhil CUHK , MIT QUT , PhD USQ.

Yan Li is an associate professor in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia. Qualifications BEng HUST , MEng HUST , PhD Flinders. Approved research supervisor in the area of: Signal Processing (090609), Computer Communications, Networks (100503), Fields of Research (FoR), Biomedical Engineering, Artificial Intelligence, Image Processing, Signal Processing and Computer Communications Networks. Research interests: Machine Learning Algorithms, Big Data Analytics, Signal/Image Processing, EEG Research, Graph Theory, and Networking Technologies.

Three-party password-based authenticated key exchange protocol based on the computational Diffie-Hellman assumption

Aqeel Sahi*

Faculty of Health, Engineering and Sciences,
Department of Math and Computing,
University of Southern Queensland,
487/521-535 West St, Darling Heights,
QLD 4350, Australia
and
Computer Center,
University of Thi-Qar, Iraq
Email: akeel_sahy@yahoo.co.uk
*Corresponding author

David Lai and Yan Li

Faculty of Health, Engineering and Sciences,
Department of Math and Computing,
University of Southern Queensland,
487/521-535 West St, Darling Heights,
QLD 4350, Australia
Email: David.Lai@usq.edu.au
Email: Yan.Li@usq.edu.au

Abstract: The three-party password-based authenticated key exchange protocol gives two clients the ability to negotiate a session key through a trusted server over a public channel. Most of the proposed 3PAKE protocols use public keys to guarantee identities; however, the sharing of public keys may lead to various types of attacks, such as a man-in-the-middle attack, which allows an attacker to simply intercept and insert traffic traversing a network. In this paper, we briefly describe an updated three-party password-based authenticated key exchange protocol and analyse its security. The proposed TPAKE protocol does not share plain-text data. Data shared between the parties are either hashed or encrypted. Using the random oracle model, the security of the proposed TPAKE protocol is formally proven under the computational Diffie-Hillman assumption. Furthermore, the analyses included in this paper show that our protocol can ensure perfect forward secrecy and can also resist many types of common attacks.

Keywords: authentication; cryptography; key exchange protocols; password-based; three-party; Diffie-Hellman; 2PAKE protocols; 3PAKE protocols; random oracle model; communication networks; distributed systems.

Reference to this paper should be made as follows: Sahi, A., Lai, D. and Li, Y. (2018) 'Three-party password-based authenticated key exchange protocol based on the computational Diffie-Hellman assumption', *Int. J. Communication Networks and Distributed Systems*, Vol. 21, No. 4, pp.560–581.

Biographical notes: Aqeel Sahi is a PhD student in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at the University of Southern Queensland, Toowoomba 4350, QLD, Australia. He received his Bachelor degree of Computer Science from the Thiqar University, Iraq in 2007, and Master degree of Information Technology from the University Utara Malaysia, Malaysia in 2010. His current research interests are in cryptography and eHealth cloud security and privacy.

David Lai is a Senior Lecturer in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia. His qualifications are BSc from CUHK, PGDipEd from CUHK, GDipCompSc from VUT, MPhil from CUHK, MIT from QUT and PhD from USQ.

Yan Li is a Full Professor in the Department of Mathematics and Computing, Faculty of Health Engineering and Sciences at University of Southern Queensland, Toowoomba 4350, QLD, Australia. Her qualifications are BEng from HUST, MEng from HUST, and PhD from Flinders. She is approved Research Supervisor in the area of signal processing (090609), computer communications, networks (100503), fields of research (FoR), biomedical engineering, artificial intelligence, image processing, signal processing and computer communications networks. Her research interests are machine learning algorithms, big data analytics, signal/image processing, EEG research, graph theory, and networking technologies.

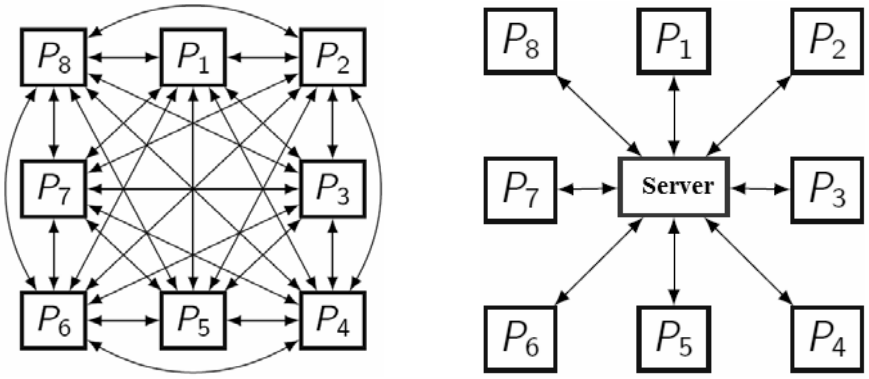
This paper is a revised and expanded version of a paper entitled 'Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol' presented at 22nd International Conference on Telecommunications (ICT), Sydney, 27–29 April 2015.

1 Introduction

Password authenticated key exchange (PAKE) protocols (Katz and Vaikuntanathan, 2013; Lee et al., 2013; Farash and Attari, 2014a, 2014b; Misra et al., 2016) are user friendly solutions to ensure the security of the session key (SK) for cryptosystems. It gives end-users the freedom of picking their own password freely without any machine intervention. The PAKE protocols were firstly brought in as a two-party protocol (2PAKE) (Jiang et al., 2013; Tang et al., 2013; Farash and Attari, 2013). However, the 2PAKE protocols are impractical for large numbers of users in a large network, since each user needs to recall many different passwords for many different users with whom they communicate (Farash and Attari, 2014b). In other words, in a group of m users, there are m(m-1)/2 user pairs. Its management of the passwords is an O(m2) which becomes impractical if m >>2.

Three-party password-based authenticated key exchange (3PAKE) protocols were proposed to overcome problems occurred in 2PAKE protocols (Amin and Biswas, 2015; Deebak et al., 2015; Li et al., 2015). In the 3PAKE protocols two users typically communicate with each other through a trusted server to establish a key in a secure manner. Unlike the 2PAKE protocols, each user needs to remember only a single password to commence the key sharing with his/her partner through a server. Figure 1 shows 2PAKE and 3PAKE protocols.

Figure 1 PAKE protocols, (a) 2PAKE protocol (b) 3PAKE protocol



Normally, users choose a natural word from a language as a password or a simple phrase rather than a long random string (Misra et al., 2011). However, these kinds of passwords are more vulnerable to password guessing attacks (Bellare and Rogaway, 1995), such as offline dictionary attacks and online dictionary attacks (Lee et al., 2013). Therefore, it is desirable to have a confidential password authenticated protocol, which may prevent the password attacks.

Khader and Lai in (2015) proposed a 3PAKE protocol which can prevent man-in-the-middle (MITM) attacks. In this paper we present the TPAKE protocol in detail. We added a counter to ensure that each user has only three attempts to generate the *SK*, otherwise, the user's account will be locked for 24 hours as a security measure. There was neither proof nor experiments provided in the original paper. In this paper, we analyse and prove formally the correctness of the TPAKE protocol using the random oracle model (ROM). Then, we discussed the security of the TPAKE protocol against several kinds of attacks in addition to the MITM attacks.

As a contribution of this study, we present a TPAKE protocol and then analyse its security. The proposed TPAKE protocol shares no plaintext data. Data shared between the parties are either hashed or encrypted. Using the ROM, the security of the proposed TPAKE protocol is formally proven under the Computational Diffie-Hillman (CDH) assumption. Furthermore, the analyses included in this chapter show that our protocol can ensure perfect forward secrecy and also resist many kinds of common attacks, such as the MITM attacks, online dictionary attacks, offline dictionary attacks, replay attacks and known key attacks.

In Section 2 we review the related work, in Section 3 we introduce the proposed work, in Section 4 we analyse the security of the proposed work and in Section 5 we present the conclusion.

2 Related work

The literature of the PAKE protocols with their related issues are reviewed in this section.

The first two-party protocol was proposed in 1992 by Bellovin and Merritt (1992). Two users could negotiate the *SK* using their passwords via public or insecure

communication channels. However, in a large network the management of the passwords is challenging. If we assume that the network has m users and adopting Bellovin and Merritt's protocol to exchange a *SK* between two users, there are $m(m\_1)/2$ passwords to be shared as shown in Figure 1(a). Those passwords have to be kept safely (Bellare and Rogaway, 1995). Therefore, many researchers shifted their attention to the 3PAKE protocols, in which a trusted server serves as a hub for users to ensure that each user needs to manage only his/her own password with the server as shown in Figure 1(b). The 3PAKE protocols are more scalable to large networks than the 2PAKE protocols.

Many well designed 3PAKE protocols were proposed. However cryptanalysis shows many issues and with these protocols and their vulnerability to various attacks.

Yang et al. (2006) proposed a 3PAKE protocol using two servers as the intermediate agents. One of the servers connects directly to the users, while another one stays unconnected to the users. They claimed that their protocol had a number of strong securities features which could secure the SK against various attacks, such as the offline dictionary attacks (Yang et al., 2006). However, Amin and Biswas (2015) stated that this protocol would be vulnerable to numerous security issues, such as insider attacks (Nose, 2011), offline password guessing attacks, and replay attacks (Amin and Biswas, 2015).

Lu and Cao (2007) proposed a simple 3PAKE protocol. They claimed that as the protocol did not require any server's public key, it could prevent many known attacks (Lu and Cao, 2007). On the other hand, Guo et al. (2008) showed that the protocol was suffering from the MITM attacks which possibly could expose authenticated information to an attacker as well as the online dictionary attacks.

Another simple 3PAKE protocol that works without a server's public key was proposed by Huang (2009). Huang stated that his protocol was not only secure against various types of attacks, but it also was more efficient than many other 3PAKE protocols. In contrast, Yoon and Yoo (2011) reported that this protocol was vulnerable to many attacks, such as the online and offline password guessing attacks.

Moreover, Wen et al. (2005) proposed a new provably secure 3PAKE protocol. Wen et al. (2005) claimed that this protocol had a security proof using a formal model and adversary capabilities. However, Nam et al. (2007) pointed out that Wen et al.'s (2005) proposed 3PAKE protocol was absolutely insecure and the security proof was incorrect (Nam et al., 2007). Nam et al. illustrated a full analysis of weaknesses in Wen et al.'s (2005) protocol and its security proof.

Furthermore, a communication-efficient 3PAKE protocol as mentioned in Bellare and Rogaway (1995). This protocol did not require the server's public key nor a symmetric encryption scheme. It had a security proof based on the computational Diffie Hellman assumption. The authors claimed that their protocol was more practical than other 3PAKE protocols. However, Wu et al. pointed out that Chang et al.'s protocol was vulnerable to the partition attacks. Therefore, attackers might guess the real password offline (Wu et al., 2012). The weaknesses of the 3PAKE protocols have motivated us to design a TPAKE protocol which can overcome the issues mentioned above.

3 Preliminaries

Due to the fact that 23.9% of 14.3 million HTTPS servers are supported by the Diffie-Hellman protocol (Adrian et al., 2015), therefore most of the 3PAKE protocols are based

on the Diffie-Hellman problem (DHP). In this paper, the proposed TPAKE protocol is also based on the DHP. The hardness of the traditional DHP is based on the discrete logarithm problem (DLP).

Definition 1 (DLP): The DLP is the problem of determining an integer χ, where $1 \leq \chi \leq p - 1$ (p is prime number), such that $\alpha^x \equiv \gamma$, where both the primitive element α and another element γ are elements in $\mathbb{Z}_p^*$ ($\alpha, \gamma \in \mathbb{Z}_p^*$, $\mathbb{Z}_p^*$ is a finite cyclic group) (Camenisch, 1998).

The only known way to solve the DHP is to compute $\chi = \log_\alpha \gamma$ or α^{xy}, where x, y are randomly picked from $\mathbb{Z}_p^*$. The computation is shown to be infeasible (Chung and Ku, 2008). Thus, to prevent attacks we should choose a large prime number p.

In this proposed protocol, we adopt a well-known complexity assumption, which is the CDH assumption (Diffie and Hellman, 1976; Chevalier et al., 2008).

Definition 2 (CDH assumption): To ensure that the DLP defined in $\mathbb{Z}_p^*$ is hard enough to attack, p should be a large prime number. Let $\mathbb{G} \subseteq \mathbb{Z}_p^*$ be a finite cyclic group of prime order q with generator g, where $p = 2q + 1$. Given (g, p), $A = g^a$, $B = g^b$ (public keys), where a, b (private keys) are randomly picked from $\mathbb{Z}_p^*$ by *Alice* ($\hat{A}$) and *Bob* ($\hat{B}$) (Diffie and Hellman, 1976; Chevalier et al., 2008). $\hat{A}$ and $\hat{B}$ represent the initiator client and the responder client, respectively, of a key exchange protocol run. We consider that the CDH assumption holds for $\mathbb{G}$, if CDH attacker β is given a challenge $\psi = (g^a, g^b)$, in order to compute g^{ab}, the success probability of β in retrieving g^{ab} in time t is denoted by:

$$suc_G^{CDH}(t) = \mathbb{P}_{ab}\left[\beta\left(g^a, g^b\right) = g^{ab}\right] \leq \varepsilon \tag{1}$$

where ε is negligible.

Definition 3 (negligible function): A function $\varepsilon : \mathbb{Z}^+ \to \mathbb{R}^+$ is negligible $\Leftrightarrow \forall c > 0, \exists \lambda_0 > 0$ such that $\forall \lambda \geq \lambda_0, \varepsilon(\lambda) \geq \lambda^{-c}$ (Bellare and Rogaway, 1993).

In other words, function ε is negligible if it approaches zero faster than any inverse polynomial does (Guo et al., 2011; Cash et al., 2009).

Note that modular operations (*mod p*) in this paper are omitted for clarity. All are done under modular p.

Definition 4 (encryption scheme): $\Gamma\{Enc, Dec\}$ is a symmetric encryption/decryption scheme that involves two algorithms (Yi et al., 2013):

- Encryption: $Enc_k(m) \to c$, where k is the SK, m is the entered plaintext message, and c is the resulted ciphertext.
- Decryption: $Dec_k(c) \to m$, where k is the SK, c is the entered ciphertext, and m is the resulted plaintext.

4 The proposed TPAKE protocol

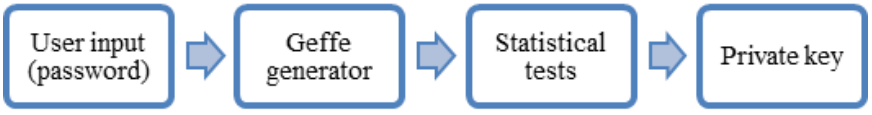
The proposed TPAKE protocol based on the CDH assumption is explained in this section. Suppose that $\hat{A}$ and $\hat{B}$ wish to exchange an agreed SK SK through a trusted server S. The agreement is reached in two rounds, as described below:

4.1 Round 1: private key setup

In this round, users $\hat{A}$ and $\hat{B}$ setup their private keys a and b based on their private information, such as passwords PW_U, the Geffe generator and the statistical tests.

The process of using the Geffe generator to generate the user private key from the user information is shown in Figure 2.

Figure 2 Private key setup (see online version for colours)



Note: S/U means both the server and the users are doing the same process.

Step 1 S/U: $(PW_U) \xrightarrow{convert} (PW_U^{base2})$

User password PW_U are converted from a printable string to a binary PW_U^{base2}.

Step 2 S/U : $(PW_U^{base2}) \xrightarrow{split} (L_1, L_2, L_3)$

PW_U^{base2} is split into three sequences. The length of the sequences must be relatively prime. For example, a 64 bit PW_U^{base2} is divided into sequences of length $L_1 = 20$, $L_2 = 21$, and $L_3 = 23$. Since L_1, L_2, and L_3 are relatively prime, the greatest common divisor (GDC) of $(L_1, L_2, L_3) = 1$, and are the legitimate input to the Geffe generator.

Step 3 S/U: $(L_1, L_2, L_3) \xrightarrow{\text{feed Geffe generator}} (seq^{base2})$

The Geffe generator is used to generate new binary pseudorandom sequence seq^{base2}. The Geffe generator has three linear feedback shift registers (LFSRs). Those LFSRs use L_1, L_2, L_3, as their initial values. Assume that the feedback functions of LFSRs are $K_1 = S_{10} + S_{19}$, $K_2 = S_{10} + S_{20}$, $K_3 = S_{10} + S_{22}$. The outputs of LFSRs, K_1, K_2, and K_3, have length $2^L - 1$ without repetitions. Since $L_1 = 20$, $L_2 = 21$, and $L_3 = 23$, therefore, the length of $K_1 = 1{,}048{,}575$, $K_2 = 2{,}097{,}151$ and $K_3 = 8{,}388{,}607$. Next, the Geffe generator will process K_1, K_2, and K_3, as shown below:

$$Z_1 = K_1 \wedge K_2$$
$$Z_1 = \neg K_2 \wedge K_3$$
$$seq^{base2} = Z_1 \oplus Z_2$$

This model represents the process of the Geffe generator, which is used to generate a new binary pseudorandom sequence. The resulting seq^{base2} has the length of $(2^{L1}–1)\ (2^{L2}–1)\ (2^{L3}–1) = 18{,}446{,}715{,}486{,}418{,}763{,}775$ without repetitions. As mentioned in (Khader and Lai, 2015), the best length to use from seq^{base2} is only the first 256 bits to overcome the time consumption. The Geffe generator is explained briefly in Section 4.2.1.

Step 4 S/U: $(seq^{base2}) \xrightarrow{\text{tests}}$ (test, result)

a Success, continue

b Failed, go to Step 1.

In this step, three types of randomness tests were used to check whether the sequence has a good randomness or not. If all the tests are passed successfully, we go to the next step. If any test fails we have to abort the calculation and start all over again. Randomness tests are explained briefly in Section 4.2.2.

Step 5 S/U: $(seq^{base2}) \xrightarrow{\text{obtained}} (seq^{base10})$

Now we convert the 256 bits from binary to decimal using the ASCII code table, then extract only eight digits from seq^{base10}, as in Khader and Lai (2015).

Step 6 S/U: $(seq^{base10}) \xrightarrow{\text{obtained}}$ (a, b) private keys, where $a, b \in \mathbb{Z}_p^*$

Computing a, b which represent the private keys of $\hat{A}$ and $\hat{B}$, respectively.

After this step, $\hat{A}$ and $\hat{B}$ obtained their private keys, a and b, using their passwords. Server S has a copy of those private keys, a and b on top of the passwords.

Step 7 $S : (PW_U) \xrightarrow{\text{hash||salt}} (\hbar(PW_U) \,||\, \text{salt})$

Server S hashes the PW_U to $\hbar(PW_U)$ to be used as a checksum in order to ensure the identity of the initiator or the responder in the next key exchanging communications. Then it concatenates $\hbar(PW_U)$ with the salt (random number, used to add complexity and security) and stores it in a hash table.

Step 8 $S : (a, b) \xrightarrow{\text{encrypt using admin key||salt}} (Enc(a, b) \,||\, \text{salt})$

S encrypts a and b using the admin key AK, then concatenates the ciphertext of a and b with salt. As shown in equations in equation (2).

$$\begin{aligned} C_a &= Enc_{Ak}(a) \\ C_{as} &= C_a \,||\, \text{salt} \\ C_b &= Enc_{AK}(b) \\ C_{bs} &= C_b \,||\, \text{salt} \end{aligned} \tag{2}$$

Step 9 End a and b were successfully setup.

4.2 Round 2: SK negotiation

In this round, $\hat{A}$ and $\hat{B}$ start negotiating about the *SK* through trusted *S*.

The process of negotiating the *SK* is shown in Figure 3.

Step 1 Set counter $i = 1$

Step 2 S: $(p, g, a, b) \xrightarrow{\text{generate publick keys}} (A, B)$

S generate public keys A and B of $\hat{A}$ and $\hat{B}$ as follows (Kumar et al., 2012; Farash and Attari, 2013):

$$A = g^a \tag{3}$$

$$B = g^b \tag{4}$$

Step 3 S: $(p, g, A|B, ID_U) \xrightarrow{\text{encrypt and sends}} U(C_{\hat{A}Info}, C_{\hat{B}Info})$

ID_U is user identification name *ID*. *S* sends the following encrypted information packages to $\hat{A}$ and $\hat{B}$:

- S to $\hat{A}$

$$C_{\hat{A}Info} = Enc_a\left(p, g, B, ID_{\hat{B}}\right) \tag{5}$$

- S to $\hat{B}$

$$C_{\hat{B}Info} = Enc_b\left(p, g, B, ID_{\hat{A}}\right) \tag{6}$$

Step 4 U: $(C_{\hat{A}Info}, C_{\hat{B}Info}) \xrightarrow{\text{decrypt}} (p, g, A|B, ID_U)$

Upon receiving $C_{\hat{A}Info}$ and $C_{\hat{B}Info}$, $\hat{A}$ and $\hat{B}$ use their private keys a and b to decrypt the packages as follows:

$$\hat{A}Info = Dec_a\left(p, g, A, ID_{\hat{B}}\right) \tag{7}$$

$$\hat{B}Info = Dec_a\left(p, g, A, ID_{\hat{A}}\right) \tag{8}$$

Step 5 S/U : $(p, g, A|B) \xrightarrow{\text{compute}} (SK)$

In this step S, $\hat{A}$ and $\hat{B}$ compute the *SK*, as follows (Kumar et al., 2012; Li, 2010):

- S and $\hat{A}$

$$SK_{\hat{A}} = B^a \tag{9}$$

- S and $\hat{B}$

$$SK_{\hat{B}} = A^b \tag{10}$$

where $SK_{\hat{A}} = SK_{\hat{B}}$

Step 6 S/U: $(SK) \xrightarrow{\text{hash and match the } SKs} (\hbar(SK_U), h(SK_U))$

a true, continue

b false, go to Step 8

S, $\hat{A}$ and $\hat{B}$, hash the generated SKs, send the hash of the SKs to each other and match the checksums received from others to ensure the correctness of the SK.

Step 7 SK is ready, go to 10

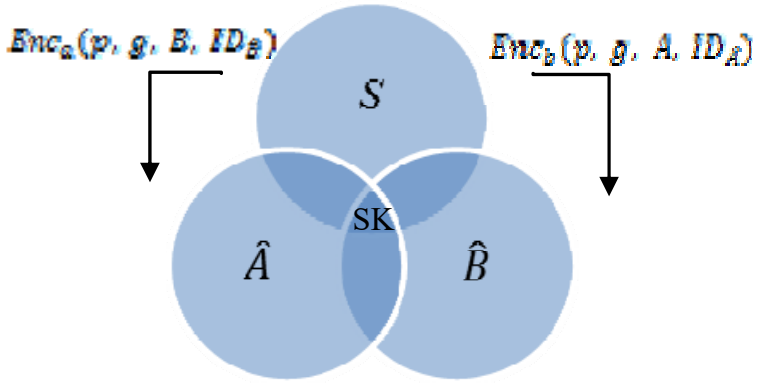
Step 8 If $i =< 3$ then i++ and go to 2, else go to 9

The user has three attempts to generate the SK, if he fails after three attempts, his account will be locked for 24 hours and a warning message will be sent to him.

Step 9 Account locked for 24 hours and sends a warning message "Your data were compromised and your account was locked for 24 hours".

Step 10 End.

Figure 3 SK negotiation (see online version for colours)



4.2.1 The Geffe generator

As shown in Figure 2, the Geffe generator is used to generate a pseudo random sequence of a balanced distribution of 0's and 1's (Wei, 2006) from the user input. The sequence will be used as the user private key if it passes the statistical tests for randomness. The Geffe generator process is illustrated in Figure 4.

Figure 4 The Geffe generator

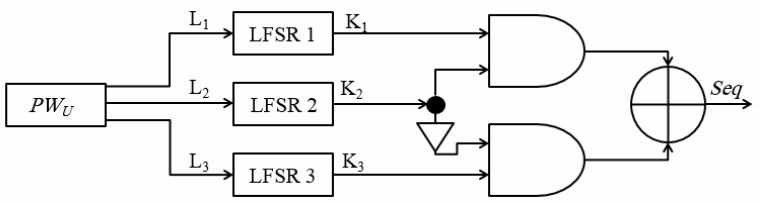


Table 1 The proposed TPAKE protocol

	$\hat{A}$	*Trusted S*	$\hat{B}$
1	Request connection channel with $\hat{B} \xrightarrow{PW_{\hat{A}}, ID_{\hat{B}}}$	-	Request connection channel with $\hat{A} \xrightarrow{PW_{\hat{B}}, ID_{\hat{A}}}$
2	Retrieve $a \in \mathbb{Z}_p^*$, a extracted from $PW_{\hat{A}}$	Retrieve $a, b \in \mathbb{Z}_p^*$, a extracted from $PW_{\hat{A}}$ and b extracted from $PW_{\hat{B}}$	Retrieve $b \in \mathbb{Z}_p^*$, b extracted from $PW_{\hat{B}}$
3	-	Store $\hbar(PW_{\hat{A}}) \|\|$ salt, $\hbar(PW_{\hat{B}}) \|\|$ salt, C_{as} and C_{bs} where: $C_a = Enc_{AK}(a)$, $C_{as} = C_a \|\|$ salt $C_b = Enc_{AK}(b)$, $C_{bs} = C_b \|\|$ salt and the AK is the admin key.	-
4	-	Set counter $i = 1$ and calculate A, B $A = g^a$, $B = g^b$	-
5	-	$\xleftarrow{Enc_a(p, g, B, ID_{\hat{B}})}$ $\xrightarrow{Enc_b(p, g, A, ID_{\hat{A}})}$	-
6	$DEc_a(p, g, B, ID_{\hat{B}})$		$DEc_b(p, g, B, ID_{\hat{A}})$
7	Calculate $SK_{\hat{A}}, SK_{\hat{A}} = B^a$	Calculate $SK_{\hat{A}}, SK_{\hat{B}}$, $SK_{\hat{A}} = B^a, SK_{\hat{B}} = A^b$	Calculate $SK_{\hat{B}}, SK_{\hat{B}} = A^b$
8	-	If $SK_{\hat{A}} = SK_{\hat{B}}$ then $\xleftarrow{\hbar(SK_{\hat{A}})}$ $\xrightarrow{\hbar(SK_{\hat{B}})}$	-
9	-	If the matching failed, then $i \leq 3$, i++, else continued	-
10	If $h(SK_{\hat{A}}) = \hbar(SK_{\hat{B}})$ then $SK_{\hat{A}}$ is secure	-	If $h(SK_{\hat{A}}) = \hbar(SK_{\hat{B}})$ then $SK_{\hat{B}}$ is secure

4.2.2 Statistical tests

The random sequences which are derived from user inputs using the Geffe generator are tested using the frequency test; serial test and Poker test (Shehata et al., 2003) to ensure the randomness of the sequence. Once the sequences pass the statistical tests, they will be converted to user private keys.

The frequency test checks the distributions of zero's (n_0) and one's (n_1) in the sequence (n). The serial test checks the distribution of two-digit patterns (n_{00}, n_{01}, n_{10}, and n_{11}) and the Poker test checks the distribution of patterns with an arbitrary length. The statistical formulas used are:

Frequency test:

$$X_1 = \frac{(n_0 - n_1)^2}{n} \tag{11}$$

Serial test:

$$X_2 = \frac{4}{n-1}\sum_{i=0}^{1}\sum_{j=0}^{1}(n_{ij})^2 - \frac{2}{n}\sum_{i=0}^{1}(n_i)^2 + 1 \tag{12}$$

Poker test:

$$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k \tag{13}$$

The accepted threshold values of the randomness tests are ($X_1 < 3.8415$), ($X_2 < 5.9915$) and ($X_3 < 14.0671$), respectively (Hosseini et al., 2014). The proposed TPAKE protocol is shown in Table 1.

5 Security analyses

Typically, a security proof investigates the communication of many parties. One of these parties is an attacker β. In a traditional proof, most of the users are authentic, which means they will work exactly as defined in the protocol. However, β may do anything she likes to break the protocol.

Once we claim that our TPAKE protocol is provably secure, we mean that the key exchange remains confidential under certain assumptions. In this section, we prove that the proposed protocol is secure under the CDH assumption. The CDH assumption is hard to solve and there is no algorithm which can solve it efficiently (Yao and Zhao, 2014; Yantao and Jianfeng, 2010).

5.1 Formal analysis

In this section we analyse the security of the TPAKE protocol. Typically, authentication between parties, and setting up SK *SK* are essential for the security of an authenticated key exchange (AKE) protocol. This section show that the TPAKE protocol is secure under the AKE security protocol, using the same approach as in (Bellare et al., 2000). In other words, the proposed TPAKE protocol is secure under the AKE security protocol when both communication parties $\hat{A}$ and $\hat{B}$ can have an authenticated SK *SK* after running the TPAKE protocol. However no other parties will be able to find out any information regarding this SK *SK*, except the trusted server *S*. Therefore, once the TPAKE protocol achieves an AKE security protocol, it means that it achieves both security essential authentication and SK setting up. In this paper, we prove that the

proposed TPAKE protocol is secure under the AKE protocol using the ROM (Bellare et al., 2000; Bellare and Rogaway, 1993, 1995; Canetti et al., 2004; Wu and Zhu, 2008).

The ROM (also called the black box) is a random function. It was introduced by Bellare and Rogaway (1993). The random function can be a perfect hash function (Bellare and Rogaway, 1993). However, a random function is impractical due to the fact that it is very big to store as well as very slow to compute. Bellare and Rogaway (1993) stated that using ROM as a random function would be practical for the purposes of security proof. Thus, we have adopted the ROM to proof the security of the proposed TPAKE protocol in this paper.

The formal analysis is discussed in three parts, as follows:

5.1.1 Characteristics of participants' capabilities

In terms of characteristics of participants, the terms of participants and long-lived keys of the adversary are explained below:

- *Participants*: the proposed protocol $\prod$ is divided into two sets of participants: client *C* and trusted server *S*. Two clients $\hat{A} \in C$ and $\hat{B} \in C$ are authenticating each other and setting up a *SK* via *S* in $\prod$. During the execution of $\prod$, participants (*C* | *S*) might have many instances (oracles). We symbolise instance *i* of a participant *U* as $\prod_U^i$.
- *Long-lived keys*: $\hat{A} \in C$ holds her password $PW_{\hat{A}}$ as her long-lived key, $\hat{B} \in C$ holds his password $PW_{\hat{B}}$ as his long-lived key, and S holds its admin password AK as its long-lived key.

In relation to the adversary's capabilities, let assume β (the adversary) is attacking. Assume β has a full control over all communication channels between participants. β's capabilities are modelled in the following oracle queries:

- Send $\left(\prod_U^i, m\right)$: this query models β sending a message *m* to $\prod_U^i . \prod_U^i$ computes the response using the protocol algorithm and resends the result back to β. Moreover, β can send $\left(\prod_U^i, \text{start}\right)$ to start a new execution of $\prod_U^i$ with another user.
- Execute $\left(\prod_{\hat{A}}^i, \prod_{\hat{B}}^i, \prod_S^K\right)$: this query models β gaining an authentic $\prod$ execution among $\prod_{\hat{A}}^i, \prod_{\hat{B}}^i$, and $\prod_S^K . \beta$ should have an authentic execution access. This query deals with dictionary attacks.
- Reveal $\left(\prod_U^i\right)$: this query models β yielding a SK from $\prod_U^i$. If $\prod_U^i$ has accepted the *SK*, it sends the *SK* back to β or else it sends nothing to β. This query deals with known key attacks.

- Corrupt (U): this query models β sending a query to U and yields his/her long-lived key. The previous SK must not be revealed after losing the long-lived key. This query shows the idea of the perfect forward secrecy.
- $\Gamma(\{Enc, Dec\}, k, \{m, c\})$: this query models β accessing to the query oracle of encryption and decryption Γ defined in Definition 4. This means that when β sends $\Gamma(Enc, k, m)$ to Γ, Γ will encrypt m using k and returns c to β. When β sends $\Gamma(Dec, k, c)$ to Γ, Γ will decrypt c using k and returns m to β.
- $h(m)$: this query models β obtaining a hash value from this query. If message m has not been queried before, a completely random hash will be sent back to β and stored in the hash table with m. Otherwise, the previous hash value will be sent to β. The ROM were used as a cryptographic hash function (Bellare and Rogaway, 1993).
- Test $\left(\prod_U^i\right)$: this query models β distinguishing between a SK and a random string. β can send a single test query to $\prod_U^i$, then a value b of query will be flipped by $\prod_U^i$. The returned result from $\prod_U^i$ will be conditioned. If $b = 1$, then the query returns the SK, else it returns a random string having the same length as that of the SK. The Test query is accessible only when $\prod_U^i$ is fresh (freshness term will be explained in the following section).

5.1.2 Definitions

In this section we will define terms used in proving the security of $\prod$.

Definition 5 (partnership): We define as partnered when the following conditions are satisfied (Bresson et al., 2009):

- $\prod_{\hat{A}}^i$ and $\prod_{\hat{B}}^i$ exchange m directly.
- $\prod_{\hat{A}}^j$ and $\prod_{\hat{B}}^j$ have the same SK.
- No one else besides $\prod_{\hat{A}}^i$ and $\prod_{\hat{B}}^i$ has the SK, except S.

Definition 6 (freshness): $\prod_U^i$ is fresh when the following conditions are satisfied (Guo et al., 2011):

- $\prod_U^i$ has an accepted SK.
- No one in the protocol has been sent a corrupted query before $\prod_U^i$ accepts.
- Neither $\prod_{\hat{A}}^i$ nor $\prod_{\hat{B}}^j$ has been sent a reveal query.

A *SK* is fresh if and only if $\prod_U^i$ is fresh.

5.1.3 AKE security

In $\prod$ execution, β wins the game and breaks the security of the AKE of $\prod$, if he makes a fresh instance $\prod_U^i$ from a single test query, and the selected bit (b) by $\prod_U^i$ is guessed correctly (Bresson et al., 2007; Wen et al., 2016). Assume that the guessed bit by β is b' after making some instances queries. Thus, the probability of guessing correctly b by β is denoted by $\mathbb{P}(b'=b)=1$, and the probability of guessing incorrectly b by β is denoted by $\mathbb{P}(b'=b)=0$. The advantage of β in breaking the security of the AKE in $\prod$, denoted by $adc_{\prod}^{AKE}(\beta)$, it is defined to be:

$$adc_{\prod}^{AKE}(\beta)=\left|2\mathbb{P}(b'=b)-1\right| \tag{14}$$

The TPAKE protocol $\prod$ is secure under the AKE protocol, only when $adc_{\prod}^{AKE}(\beta)$ is negligible.

Theorem 1: Let $adc_{\prod}^{AKE}(\beta)$ be the advantage of β in breaking the security of the AKE security of $\prod$ in a time bound t, and making the queries s_q(send), e_q(excute), and h_q(hash). Then we have:

$$adc_{\prod}^{AKE}\left(t,s_q,e_q,h_q\right)\le\frac{s_q}{n}+h_q\cdot\left(s_q+e_q\right)\cdot suc_G^{CDH}(t') \tag{15}$$

where t' is the running time of suc_G^{CDH} and n is the number of possible passwords.

Proof: Let us assume that $\mathbb{P}[x]$ is the probability of β in breaking the AKE security of $\prod$. Let $\mathbb{P}[x_1]$ is the probability of β in breaking the AKE security of $\prod$ by breaking the password security. Let $\mathbb{P}[x_2]$ is the probability of β in breaking the AKE security of $\prod$ without breaking the password security. Then, we get:

$$\mathbb{P}[x]=\mathbb{P}[x_1]+\mathbb{P}[x_2] \tag{16}$$

$\mathbb{P}[x]$ is breaking AKE security:

- $\mathbb{P}[x_1]$ breaking AKE security by breaking the password
- $\mathbb{P}[x_2]$ breaking AKE security without breaking the password.

If $\mathbb{P}[x_1]$ = 0.5, then $\mathbb{P}[x_2]$ must also be 0.5. Because we have two choices only, whether it is breaking the password or not (cannot choose both cases). The population will be divided into two values, and the summation should be less or equal to.

We have built $\mathbb{P}[x]$ from $\mathbb{P}[x_1]$ and $\mathbb{P}[x_2]$ as follows:

1 *The probability* $\mathbb{P}[x_1]$ *of* β *in breaking the AKE security of* $\prod$ *by breaking the password security*. β could break the security of the password (either $PW_{\hat{A}}$ or $PW_{\hat{B}}$) in $\prod$ in two ways: online dictionary attacks or offline dictionary attacks. We denote the probability of β breaking the security by an online dictionary attacks as $\mathbb{P}[x_1^{on}]$, and the probability of β breaking the security by an offline dictionary attack as $\mathbb{P}[x_1^{off}]$. Then, we get:

$$\mathbb{P}[x_1] = \mathbb{P}[x_1^{on}] + \mathbb{P}[x_1^{off}] \tag{17}$$

The probabilities of online dictionary attacks and offline dictionary attacks are analysed below:

Online dictionary attacks: β can check the correctness of picking either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ by executing it in Round 1 algorithm to retrieve either a or b. Then β uses a or b as a decryption key in Round 2 algorithm (Step 4) to retrieve p, g, $A \mid B$ and ID_U. Finally, β uses the obtained information to calculate the SK and then sending $\hbar(SK)$ as in Round 2 algorithm (Step 6) to check the accuracy of either $PW_{\hat{A}}$ or $PW_{\hat{B}}$. Therefore, $\mathbb{P}[x_1^{on}]$ is limited by s_q and n as follows:

$$\mathbb{P}[x_1^{on}] \leq \frac{s_q}{n} \tag{18}$$

Offline dictionary attacks: as mentioned in Definition 2, β can break the CDH assumption with the probability $\grave{}\varepsilon$ within time bound $\grave{}t$, $suc_G^{CDH}(\grave{}t) = \grave{}\varepsilon$. In the CDH assumption, β is given a challenge $\psi = (g^a, g^b)$. While in the proposed protocol a and b are driven from $PW_{\hat{A}}$ and $PW_{\hat{B}}$, then β has to attack $PW_{\hat{A}}$ and $PW_{\hat{B}}$. $PW_{\hat{A}}$ and $PW_{\hat{B}}$ are stored as hash values in the hash table in S, and they are concatenated with salt, as in Round 1 algorithm (Step 7) $\hbar(PW_{\hat{A}}) \,||\, \text{salt}\; \hbar(PW_{\hat{B}}) \,||\, \text{salt}$. Therefore, the probability ε_1 that β can correctly pick the hash value $\hbar(PW_{\hat{A}})$ from the possible hash $(\hbar(PW_{\hat{A}}) \,||\, \text{salt})$ queries or pick the hash value $\hbar(PW_{\hat{B}})$ from the possible hash $(\hbar(PW_{\hat{B}}) \,||\, \text{salt})$ queries from the hash table at S is:

$$\varepsilon_1 \geq \frac{1}{h_q} \tag{19}$$

The probability ε_2 that β correctly picks PW_U to retrieve any user's password is equivalent to the probability of picking the value i by β:

$$\varepsilon_2 \geq \frac{1}{e_q} \tag{20}$$

As a result, the probability $suc_G^{CDH}(\curlyvee)$ of β in breaking the CDH assumption is equivalent to the probability $\mathbb{P}[x_1]$ of β in breaking the AKE security of $\prod$ by breaking the password security using offline dictionary attacks. The probability is equal to the probability $\mathbb{P}[x_1^{off}]$ that β correctly picks the password of either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ by offline dictionary attacks multiplied by the probability ε_1 that β can correctly pick the hash value $\hbar(PW_{\hat{A}})$ from the possible hash $(\hbar(PW_{\hat{A}}) \| \text{salt})$ queries or pick the hash value $\hbar(PW_{\hat{B}})$ from the possible hash $(\hbar(PW_{\hat{B}}) \| \text{salt})$ queries from the hash table at S multiplied by the probability ε_2 that β correctly picks the PW_U:

$$\begin{aligned} &suc_G^{CDH}(\curlyvee) = \grave{\varepsilon} = \mathbb{P}\left[x_1^{off}\right] \cdot \varepsilon_1 \cdot \varepsilon_2 \geq \mathbb{P}\left[x_1^{off}\right] \cdot \frac{1}{h_q} \cdot \frac{1}{e_q}, \\ &suc_G^{CDH}(\curlyvee) \geq \mathbb{P}\left[x_1^{off}\right] \cdot \frac{1}{h_q} \cdot \frac{1}{e_q}, \\ &\mathbb{P}\left[x_1^{off}\right] \leq suc_G^{CDH}(\curlyvee) \cdot h_q \cdot e_q, \\ &\because \mathbb{P}[x_1] = \mathbb{P}\left[x_1^{on}\right] + \mathbb{P}\left[x_1^{off}\right] \leq \frac{s_q}{n} + suc_G^{CDH}(\curlyvee) \cdot h_q \cdot e_q \end{aligned} \tag{21}$$

2 *The probability $\mathbb{P}[x_2]$ of β in breaking the AKE security of $\prod$ without breaking the password security*. β could break the security of the AKE of $\prod$ without knowing either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ by breaking either A or B (public keys) or by guessing the private key (either a or b) by sending fake public key queries to either A to $\hat{A}$ or B to $\hat{B}$. Breaking A or B can lead to the MITM attacks (Khader and Lai, 2015). However, β needs to know the hash value of either A or B in order to do the attacks. The probability ε_3 that β correctly picks a hash value is:

$$\varepsilon_3 \geq \frac{1}{h_q} \tag{22}$$

The probability ε_4 that β correctly guesses the private key (either a or b) by sending fake public key queries to either A to $\hat{A}$ or B to $\hat{B}$ is equivalent to the probability of picking the value j by β:

$$\varepsilon_4 \geq \frac{1}{s_q} \tag{23}$$

As a result, the probability $suc_G^{CDH}(\curlyvee)$ of β in breaking the CDH assumption is equivalent to the probability $\mathbb{P}[x_2]$ that β breaks the AKE security of $\prod$ without breaking the password security multiplied the probability ε_3 that β knows the hash value of $PW_{\hat{A}}$ or $PW_{\hat{B}}$ in order to generate a or b multiplied the probability ε_4 that β

correctly guesses the private key (either *a* or *b*) by sending fake public key queries to either *A* to $\hat{A}$ or B to $\hat{B}$:

$$
\begin{aligned}
&suc_G^{CDH}(\text{ٹ}) = \text{ٹ}\varepsilon = \mathbb{P}[x_2]\cdot\varepsilon_3\cdot\varepsilon_4 \geq \mathbb{P}[x_2]\cdot\frac{1}{h_q}\cdot\frac{1}{s_q},\\
&suc_G^{CDH}(\text{ٹ}) \geq \mathbb{P}[x_2]\cdot\frac{1}{h_q}\cdot\frac{1}{s_q},\\
&\mathbb{P}[x_2] \leq h_q\cdot s_q\cdot suc_G^{CDH}(\text{ٹ}),\\
&\because \mathbb{P}[x] = \mathbb{P}[x_1] + \mathbb{P}[x_2]\\
&\because \mathbb{P}[x_1] \leq \frac{s_q}{n} + suc_G^{CDH}(\text{ٹ})\cdot h_q\cdot e_q\\
&\because \mathbb{P}[x] \leq \frac{s_q}{n} + suc_G^{CDH}(\text{ٹ})\cdot h_q\cdot e_q + h_q\cdot s_q\cdot suc_G^{CDH}(\text{ٹ})
\end{aligned}
\tag{24}
$$

$$
\because \mathbb{P}[x] = adv_{\prod}^{AKE}\left(t_q, s_q, e_q, h_q\right) \leq \frac{s_q}{n} + h_q\cdot\left(e_q + s_q\right)\cdot suc_G^{CDH}(\text{ٹ}) \quad \blacksquare \tag{25}
$$

To sum up, as in Definition 2, the CDH assumption holds for suc_G^{CDH}, and suc_G^{CDH} is less than the negligible *ε*. Also, h_q, e_q, and s_q cannot be large due to the limited attempts and time in Round 2. Therefore, $adv_{\prod}^{AKE}$ is negligible and the CDH assumption holds in this case. Hence, $\prod$ is AKE secured.

Table 2 Comparison with existing protocols

	Protocols	*Attacks*					*Perfect forward secrecy*	*Security prove*
		MITM	*Online dictionary*	*Offline dictionary*	*Replay*	*Known key*		
1	Yang et al. (2006)	×	√	×	×	√	×	√
2	Lu and Cao (2007)	×	×	√	√	√	√	×
3	Huang (2009)	√	×	×	√	√	×	×
4	Wen et al. (2005)	×	×	×	√	√	×	×
5	Xie et al. (2017)	×	√	√	×	×	√	√
6	Amin et al. (2016)	√	×	√	√	√	×	√
7	Islam (2015)	√	×	√	√	×	√	√
8	Farash and Attari (2014b)	×	√	√	√	√	√	×
9	Lee et al. (2013)	×	√	√	√	×	×	×
10	Our TPAKE	√	√	√	√	√	√	√

5.2 Discussion

This section discusses the TPAKE protocol security against relevant attacks. We evaluated the ability of our protocol in resisting multiple attacks and then compared it with related protocols Yang et al. (2006), Lu and Cao (2007), Huang (2009), Wen et al. (2005), Xie et al. (2017), Amin et al. (2016), Islam (2015), Farash and Attari (2014b), and Lee et al. (2013) proposed recently. Table 2 shows the performance comparisons of our protocol and some other related protocols. From Table 2, it is clear that the TPAKE is more secure than other protocols as it is protected against multiple attacks, and provides more security features. The following six proven propositions ensure the security of the proposed protocol against many attacks, and provide perfect forward secrecy features.

Proposition 1: The proposed protocol $\prod$ can mitigate the MITM attacks.

Proof: The public keys A and B are necessary for establishing a MITM attack. In order to begin attacking protocol $\prod$ using the MITM attack, β needs to know both A and B on top of g and p. However, as shown in Round 2 algorithm (Step 3) in $\prod$, all these parameters are encrypted using the user private key (a or b), which is known only by the user himself ($\hat{A}$ or $\hat{B}$) and S. Thus, the proposed protocol $\prod$ can mitigate the MITM attack.■

Proposition 2: The proposed protocol $\prod$ can mitigate online dictionary attacks.

Proof: Online dictionary attacks can be mitigated by validating the correctness of the input every time. In this proposed protocol, integrity checks are provided. $\prod$ checks the sent SK by matching the hashes which are stored at S, as well as checking them with end users $\hat{A}$ and $\hat{B}$. In other word, S, $\hat{A}$ and $\hat{B}$ can validate one another using the stored hash values, as in Round 2 algorithm (Step 6). Thus, the proposed protocol $\prod$ can mitigate online dictionary attacks.■

Proposition 3: The proposed protocol $\prod$ can mitigate offline dictionary attacks.

Proof: Offline dictionary attacks can be performed by a passive attacker β who may have control over the session between $\hat{A}$ and $\hat{B}$. β tries to guess one of the passwords, either $PW_{\hat{A}}$ or $PW_{\hat{B}}$, in order to compute the SK. However, β cannot check the correctness of the password until he passes the verifications process in Round 2 algorithm (Step 6). To do so, β needs to know the SK before knowing either $PW_{\hat{A}}$ or $PW_{\hat{B}}$, which is not possible. Thus, the proposed protocol $\prod$ can mitigate offline dictionary attacks.■

Proposition 4: The proposed protocol $\prod$ can mitigate replay attacks.

Proof: Replay attacks can be performed by an active attacker β who may compromise the message sent between the parties. Assume β can catch $\hat{A}$'s message and replay the compromised message to $\hat{B}$. Due to the non-repudiation feature provided by the proposed protocol, $\hat{B}$ can easily detect that the message was compromised by β (Khader and Lai, 2015). Thus, the proposed protocol $\prod$ can mitigate replay attacks.■

Proposition 5: The proposed protocol $\prod$ can mitigate the known key attacks.

Proof: An attacker β may use a previous *SK* in order to make the known key attacks. However, *p* and *g* are going to be changed in each session (Khader and Lai, 2015), therefore the public keys *A* and *B* will differ. As the consequence the *SK* will differ in each session. Furthermore, it is hard to retrieve the private keys *a* and *b* due to the CDH assumption. Thus, the proposed protocol $\prod$ can mitigate the known key attacks.■

Proposition 6: The proposed protocol $\prod$ can ensure a perfect forward secrecy.

Proof: We can define the perfect forward secrecy as, if either $PW_{\hat{A}}$ or $PW_{\hat{B}}$ is exposed by β in order to compute the *SK*, the previous *SK* should remain safe. The proposed protocol can ensure the perfect forward secrecy as β cannot find out the previous *SK* even when he knows the current *SK*, because all the old *SK*s are stored as hashes. Thus, the proposed protocol $\prod$ can ensure the perfect forward secrecy.■

In addition, we compare our TPAKE protocol with several existing protocols in terms of attacks, perfect forward secrecy, and security proves. Comparison results are shown in Table 2.

6 Conclusions

An enhanced robust TPAKE protocol based on the CDH assumption and the ROM was presented in this paper with its formal analyses. Compared with other 3PAKE protocols, the proposed protocol performs better results due to the fact that the TPAKE protocol never shares clear plain information through insecure channels. The analysis of security using the ROM shows that the proposed protocol achieves the mutual authentication, safe and secure *SK*, ensures the perfect forward secrecy and prevents against multi types of attacks.

Acknowledgements

Dr. Barbara Harmes is gratefully acknowledged for her help and support.

References

Adrian, D., Bhargavan, K., Durumeric, Z., Gaudry, P., Green, M., Halderman, J.A. et al. (2015) 'Imperfect forward secrecy: how Diffie-Hellman fails in practice', *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ACM.

Amin, R. and Biswas, G. (2015) 'Cryptanalysis and design of a three-party authenticated key exchange protocol using smart card', *Arabian Journal for Science and Engineering*, Vol. 40, No. 11, pp.3135–3149.

Amin, R., Islam, S.H., Biswas, G., Khan, M.K., Leng, L. and Kumar, N. (2016) 'Design of an anonymity-preserving three-factor authenticated key exchange protocol for wireless sensor networks', *Computer networks*, Vol. 101, pp.42–62.

Bellare, M., Pointcheval, D. and Rogaway, P. (2000) 'Authenticated key exchange secure against dictionary attacks', *International Conference on the Theory and Applications of Cryptographic Techniques*, Springer.

Bellare, M. and Rogaway, P. (1995) 'Provably secure session key distribution: the three party case', *Proceedings of the twenty-seventh Annual ACM Symposium on Theory of Computing*, ACM.

Bellare, M. and Rogaway, P. (1993) 'Random oracles are practical: a paradigm for designing efficient protocols', *Proceedings of the 1st ACM Conference on Computer and Communications Security*, ACM.

Bellovin, S.M. and Merritt, M. (1992) 'Encrypted key exchange: password-based protocols secure against dictionary attacks', *Proceedings 1992 IEEE Computer Society Symposium on Research in Security and Privacy*, IEEE.

Bresson, E., Chevassut, O. and Pointcheval, D. (2007) 'Provably secure authenticated group Diffie-Hellman key exchange', *ACM Transactions on Information and System Security (TISSEC)*, Vol. 10, No. 3, p.10.

Camenisch, J.L. (1998) *Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem*, ETH Zurich.

Canetti, R., Goldreich, O. and Halevi, S. (2004) 'The random oracle methodology, revisited', *Journal of the ACM (JACM)*, Vol. 51, No. 4, pp.557–594.

Cash, D., Kiltz, E. and Shoup, V. (2009) 'The twin Diffie-Hellman problem and applications', *Journal of Cryptology*, Vol. 22, No. 4, pp.470–504.

Chevalier, Y., Küsters, R., Rusinowitch, M. and Turuani, M. (2008) 'Complexity results for security protocols with Diffie-Hellman exponentiation and commuting public key encryption', *ACM Transactions on Computational Logic (TOCL)*, Vol. 9, No. 4, p.24.

Chung, H-R. and Ku, W-C. (2008) 'Three weaknesses in a simple three-party key exchange protocol', *Information Sciences*, Vol. 178, No. 1, pp.220–229.

Deebak, B.D., Muthaiah, R., Thenmozhi, K. and Swaminathan, P. (2015) 'Evaluating three party authentication and key agreement protocols using IP multimedia server-client systems', *Wireless Personal Communications*, Vol. 81, No. 1, pp.77–99.

Diffie, W. and Hellman, M. (1976) 'New directions in cryptography', *IEEE Transactions on Information Theory*, Vol. 22, No. 6, pp.644–654.

Farash, M.S. and Attari, M.A. (2014a) 'An efficient and provably secure three-party password-based authenticated key exchange protocol based on Chebyshev chaotic maps', *Nonlinear Dynamics*, Vol. 77, Nos. 1–2, pp.399–411.

Farash, M.S. and Attari, M.A. (2014b) 'An enhanced and secure three-party password-based authenticated key exchange protocol without using server's public-keys and symmetric cryptosystems', *Information Technology and Control*, Vol. 43, No. 2, pp.143–150.

Farash, M.S. and Attari, M.A. (2013) 'An enhanced authenticated key agreement for session initiation protocol', *Information Technology and Control*, Vol. 42, No. 4, pp.333–342.

Guo, H., Li, Z., Mu, Y., Zhang, F., Wu, C. and Teng, J. (2011) 'An efficient dynamic authenticated key exchange protocol with selectable identities', *Computers and Mathematics with Applications*, Vol. 61, No. 9, pp.2518–2527.

Guo, H., Li, Z., Mu, Y. and Zhang, X. (2008) 'Cryptanalysis of simple three-party key exchange protocol', *Computers and Security*, Vol. 27, No. 1, pp.16–21.

Hosseini, S.M., Karimi, H. and Jahan, M.V. (2014) 'Generating pseudo-random numbers by combining two systems with complex behaviors', *Journal of Information Security and Applications*, Vol. 19, No. 2, pp.149–162.

Huang, H.F. (2009) 'A simple three-party password-based key exchange protocol', *International Journal of Communication Systems*, Vol. 22, No. 7, pp.857–862.

Islam, S.H. (2015) 'Design and analysis of a three party password-based authenticated key exchange protocol using extended chaotic maps', *Information Sciences*, Vol. 312, pp.104–130.

Jiang, Q., Ma, J., Li, G. and Ma, Z. (2013) 'An improved password-based remote user authentication protocol without smart cards', *Information Technology and Control*, Vol. 42, No. 2, pp.113–123.

Katz, J. and Vaikuntanathan, V. (2013) 'Round-optimal password-based authenticated key exchange', *Journal of Cryptology*, Vol. 26, No. 4, pp.714–743.

Khader, A.S. and Lai, D. (2015) 'Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol', *2015 22nd International Conference on Telecommunications (ICT)*, IEEE.

Kumar, C.K., Jose, G.J.A., Sajeev, C. and Suyambulingom, C. (2012) 'Safety measures against man-in-the-middle attack in key exchange', *ARPN Journal of Engineering and Applied Sciences*, Vol. 7, No. 2, pp.243–246.

Lee, C-C., Li, C-T. and Hsu, C-W. (2013) 'A three-party password-based authenticated key exchange protocol with user anonymity using extended chaotic maps', *Nonlinear Dynamics*, Vol. 73, Nos. 1–2, pp.125–132.

Li, C-T., Lee, C-W. and Shen, J-J. (2015) 'A secure three-party authenticated key exchange protocol based on extended chaotic maps in cloud storage service', *2015 International Conference on Information Networking (ICOIN)*, IEEE.

Li, N. (Ed.) (2010) 'Research on Diffie-Hellman key exchange protocol', *2010 2nd International Conference on Computer Engineering and Technology (ICCET)*, IEEE.

Lu, R. and Cao, Z. (2007) 'Simple three-party key exchange protocol', *Computers and Security*, Vol. 26, No. 1, pp.94–97.

Misra, S., Goswami, S., Pathak, G.P. and Shah, N. (2011) 'Efficient detection of public key infrastructure-based revoked keys in mobile ad hoc networks', *Wireless Communications and Mobile Computing*, Vol. 11, No. 2, pp.146–162.

Misra, S., Goswami, S., Taneja, C. and Mukherjee, A. (2016) 'Design and implementation analysis of a public key infrastructure-enabled security framework for ZigBee sensor networks', *International Journal of Communication Systems*, Vol. 29, No. 13, pp.1992–2014.

Nam, J., Lee, Y., Kim, S. and Won, D. (2007) 'Security weakness in a three-party pairing-based protocol for password authenticated key exchange', *Information Sciences*, Vol. 177, No. 6, pp.1364–1375.

Nose, P. (2011) 'Security weaknesses of authenticated key agreement protocols', *Information Processing Letters*, Vol. 111, No. 14, pp.687–696.

Shehata, K., Hussien, H. and Hamdy, N. (2003) 'Design and implementation of a universal communication security unit on an FPGA', *2003 IEEE 46th Midwest Symposium on Circuits and Systems*, IEEE.

Tang, H., Liu, X. and Jiang, L. (2013) 'A robust and efficient timestamp-based remote user authentication scheme with smart card lost attack resistance', *IJ Network Security*, Vol. 15, No. 6, pp.446–454.

Wei, S. (2006) 'On generalization of Geffe's generator', *IJCSNS International Journal of Computer Science and Network Security*, Vol. 6, No. 8A, pp.161–165.

Wen, H-A., Lee, T-F. and Hwang, T. (2005) 'Provably secure three-party password-based authenticated key exchange protocol using Weil pairing', *IEE Proceedings-Communications*, Vol. 152, No. 2, pp.138–143.

Wen, W., Wang, L. and Pan, J. (2016) 'Unified security model of authenticated key exchange with specific adversarial capabilities', *IET Information Security*, Vol. 10, No. 1, pp.8–17.

Wu, S., Pu, Q., Wang, S. and He, D. (2012) 'Cryptanalysis of a communication-efficient three-party password authenticated key exchange protocol', *Information Sciences*, Vol. 215, pp.83–96.

Wu, S. and Zhu, Y. (2008) 'Password-based authenticated key establishment for wireless group communications in an ad hoc mode', *International Journal of Communication Networks and Distributed Systems*, Vol. 1, Nos. 4–6, pp.398–413.

Xie, Q., Wong, D.S., Wang, G., Tan, X., Chen, K. and Fang, L. (2017) 'Provably secure dynamic id-based anonymous two-factor authenticated key exchange protocol with extended security model', *IEEE Transactions on Information Forensics and Security*, Vol. 12, No. 6, pp.1382–1392.

Yang, Y., Deng, R.H. and Bao, F. (2006) 'A practical password-based two-server authentication and key exchange system', *IEEE Transactions on Dependable and Secure Computing*, Vol. 3, No. 2, pp.105–114.

Yantao, Z. and Jianfeng, M. (2010) 'A highly secure identity-based authenticated key-exchange protocol for satellite communication', *Journal of Communications and Networks*, Vol. 12, No. 6, pp.592–599.

Yao, A.C-C. and Zhao, Y. (2014) 'Privacy-preserving authenticated key-exchange over internet', *IEEE Transactions on Information Forensics and Security*, Vol. 9, No. 1, pp.125–140.

Yi, X., Ling, S. and Wang, H. (2013) 'Efficient two-server password-only authenticated key exchange', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 24, No. 9, pp.1773–1782.

Yoon, E.J. and Yoo, K.Y. (2011) 'Cryptanalysis of a simple three-party password-based key exchange protocol', *International Journal of Communication Systems*, Vol. 24, No. 4, pp.532–542.

Preventing Man-In-The-Middle Attack in Diffie-Hellman Key Exchange Protocol

Aqeel Sahi Khader
Department of Mathematics and Computing
University of Southern Queensland
Toowoomba, Queensland, Australia
akeel_sahy@yahoo.co.uk

David Lai
Department of Mathematics and Computing
University of Southern Queensland
Toowoomba, Queensland, Australia
david.lai@usq.edu.au

***Abstract*— The acceleration in developments in communication technology has led to a consequent increase in the vulnerability of data due to penetration attacks. These attacks often came from outside where non-qualified companies develop IT projects. Cryptography can offer high levels of security but has recently shown vulnerabilities such as the man-in-the-middle (MITM) attack in areas of key exchange protocols, especially in the Diffie-Hellman (DH) protocol. Firstly, this paper presents an overview of MITM attacks targeted at the DH protocol then discusses some of the shortcomings of current defenses. A proposed method to secure DH, which helps secure systems against MITM attacks, is then presented. This method involves the use of Geffe generation of binary sequences. The use of Geffe generator offers high levels of randomness. Data hashed and encrypted using this proposed method will be so difficult to intercept and decrypt without the appropriate keys. This offers high levels of security and helps prevent MITM attacks.**

Keywords*— public key; random number generation; data security; cryptography; message authentication; digital signatures.

I. INTRODUCTION

Nowadays, data security and privacy are a controversial issue. Researchers are trying to do their best to find out the perfect way to secure the data efficiently. One of these solutions is encrypting the data using cryptography algorithms. Cryptography is the science of converting the readable information into unreadable or hidden, and only the authorized persons or machines can retrieve or obtain the original texts. It is divided into two major types (symmetric and asymmetric) regarding their keys. (1) Symmetric cryptosystems require the users to have the same key to be used for encryption and decryption processes. (2) Asymmetric cryptosystems require the users to have two different keys (encryption key and decryption key). Symmetric key should be changed from time to time to make it more secure and unbreakable to prevent other users from obtaining the plain text. Therefore, the security of any symmetric cryptography system depends on key exchange protocol used by the system. Key exchange protocol is the way of distributing the keys in a secure manner among the users [1]. There are so many ways to exchange the keys between Alice and Bob, Alice can choose a key then to send to Bob physically (mail or person). Alternatively, if they have an old key so Alice can choose new key and encrypt it by the old one and send to Bob. Unfortunately, those techniques are not secure in wide distributed systems nowadays for many reasons. Consequently, the secrecy of their messages will rely heavily on the chosen key exchange protocol, for example, Diffie-Hellman. Furthermore, asymmetric cryptosystems process is slower than symmetric one [1]. Symmetric cryptosystems uses a shared secret key to be used for encrypt and decrypt processes. While asymmetric cryptosystems does not use a single shared secret key, as an alternative it uses mathematical key pairs: a private and public key. In this cryptosystems the communications are decrypted with the private key and are encrypted with the public key. As a result asymmetric cryptosystems have consuming too much computing power to produce the two keys (private and public) and that is why it's slower than symmetric cryptosystems.

Diffie-Hellman key exchange protocol was brought in 1976 by Whitfield Diffie and Martin Hellman [2], this protocol is widely used for secure key exchange. The process of this protocol supposes that Alice and Bob have different private keys and they have to agree upon two relatively prime numbers p, g then each of them uses the obtained information to calculate the public keys. After that they share their public keys between each other and use it with the private one, p and g to get the same shared key. As a result, both of Alice and Bob obtained the shared key without sending their private keys through the channel [3].

In this paper, we will explain the problem of Man-In-The-Middle (MITM) attack in Diffie-Hellman protocol and the current defenses. This paper proposes improvements which can help preventing MITM attack. These can verify and secure the communications between Alice and Bob.

Users have usernames and passwords, and these will be used to connect them with their systems. Passwords as known are a mixed string, and it can contain numbers, symbols, lowercase, and uppercase. These mixtures can be represented in binary using ASCII code. The binary can be used to generate a different binary sequence using random number generators and extract new information, which can be helpful to overcome the problem.

978-1-4799-8078-9/15/$31.00 ©2015 IEEE

In section II discuss the related work, section III present the proposed work and section IV describe the conclusion and future work.

II. RELATED WORKS

As mentioned, the scheme of DH was brought in 1976 by Whitfield Diffie and Martin Hellman [2], Diffie-Hellman Key Exchange (DHKE) is one of the best protocols to exchange the keys safely. It has so many advantages that made most of the cryptographers feel confidante when they use it in their cryptosystems. However, DHKE has a serious issue called Man-In-The_Middle attack (MITM); this algorithm is vulnerable to this kind of attack in which Eve can attack (active attack, she can modify) all communications between Alice and Bob [4]. Because of this type of attack, researchers tried to find the best defense for that, as a result they came out with several answers. The most famous solutions that the researchers have found are: Digital Signatures (DS) and Message Authentication Code (MAC). Even though they are convenient for many systems, but they still have some weaknesses we need to think about.

Digital Signatures (DS): Generally, DS used the private key to sign the message and uses the public key to verify the message in the other party. DS faces difficulties with securing the private keys, verification and secrecy. In terms of securing the private keys, if they are exposed, all security promises are gone [5]. Therefore, most of developers and cryptosystem designers believe that we should keep the private keys in a safe way (like never send or store the private keys in plain text). If we lose them, we could have so many damages. Subsequently, anyone who gets the private key can sign the message and send to the other party (the person who has the public key, Bob), he will recognize it as a valid message (the private key holder signed the message, Alice), hence Bob will believe that the message was sent by Alice.

Every person who holds the private key can sign a document. In other words, the DS does not automatically ensure that Alice signed the document. It does ensure the Alice document was signed by someone who had access to the private key [6]. So it is like stamps, it can be stolen and utilize by other people. In relation to verification and secrecy, when the DS failed to be verified by Bob (the PuK holder) the system will flag the message as invalid because it cannot be determined whether the message was corrupted by Eve or Alice used a false private key. This means that Alice has to be responsible for the security of each of her private keys. The DS can provide authenticity but cannot provide security. Therefore encryption and decryption are required to add security. Without this, DS cannot prevent the message from being intercepted changed by MITM [6].

Message Authentication Code (MAC): MACs differ from DS as in MAC both side (Alice and Bob) need to use the same key for generating the keyed MAC tag and verification.

Commonly, Alice generates MAC tag using MAC algorithm and sends the tag with the message to Bob. Bob uses the same MAC algorithm with the same key to generate the keyed MAC tag. Then Bob check his MAC tag with Alice MAC tag if they are identical, then he will mark the message as a valid message, else he will mark the message as an invalid message.

Therefore, both of sender and receiver must agree on the same shared key and MAC algorithm before they start their communications, just like the case with symmetric-key systems. Because of that, Bob cannot prove that the message coming from Alice (unlike digital signature), which mean MAC does not provide a non-repudiation property. Anyone who receives the message and can confirm a MAC, for example Bob, can produce MACs for different messages. Thus, 'MACs are symmetric-key schemes, and they do not provide a non-repudiation' [7].

As a result, Eve can attack the communication channels and record the messages that would be sent from Alice to Bob, and later Eve sends a copy of the messages to Bob. This will make Bob feels that those messages coming from Alice. Eve might send the message back to Alice, who would trust that it came from Bob [8].

III. PROPOSED WORK

The proposed work aims to distribute the keys between Alice and Bob without being compromised by Eve. Figure 1 explains the way that Alice and Bob communicates with the server and the data that would be sending through the channels to retrieve the shared key.

Assume that the password of Alice contains eight characters (password = abcdefgh (a character string), 64 bits). The binary representation of the ASCII code for "abcdefgh" is: a=97 (01100001) + b=98 (01100010) + c=99 (011000011) + d=100 (01100100) + e=101 (01100101) + f=102 (01100110) + g=103 (01100111) + h=104 (01101000). As a result we have obtained the password in binary numbers. The sequence will be: 0110000101100010011000110110010001100101011001100110011101101000 (64 bits)

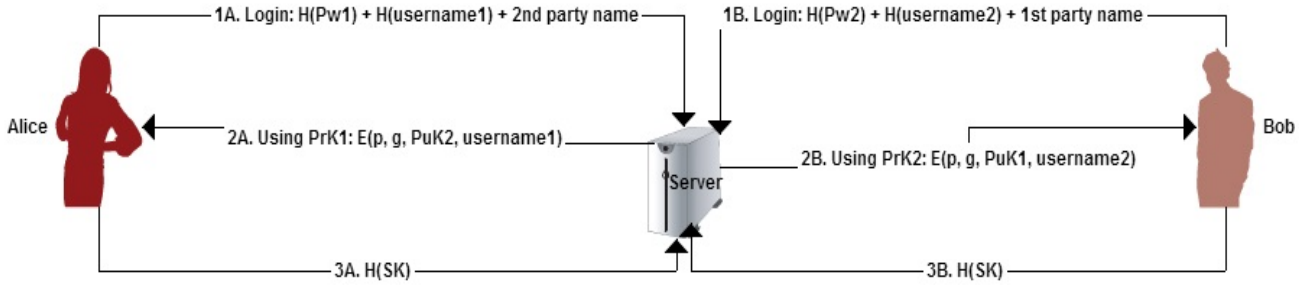


Fig. 1. Alice And Bob Communicate With The Server to Obtain The Shared Key

To make it more secure we used a pseudo random sequence generator to generate longer sequence.

A. Geffe Generator

Geffe generator is a pseudo random sequence generator. It uses three registers to generate one random sequence. The Geffe generator chose as it is in stream cipher cryptography to generate the new sequence. This generator used for several reasons: Firstly, the complexity of this generator could be superior in a different configuration of the steps. Secondly, the device does have some necessary features; for example, it has a balanced distribution of 0's and 1's in its output. It also gives the benefit of being useful as a module of a building of similar arrangements. The whole device could play the role of LFSR1 (linear feedback shift registers) in the same arrangement with like generators, and this complexity would escalate accordingly [9]. The Geffe generator needs three (LFSRs). The length of those LFSRs should be relatively prime, which mean the greatest common divisor (GCD) of the length of the initial values that would be input into the three registers LFSR1 (Len1), LFSR2 (Len2) and LFSR3 (Len3) is one:

$$GCD\ (Len1, Len2, Len3) = 1 \qquad (1)$$

Suppose the first one will have 20 bits as input for LFSR1, the second will have 21 bits as input for LFSR2 and the last one will have 23 bits as input for LFSR3, all together will be 64 bits (it is possible to choose any three numbers relatively primes and give us 64 bits).

20 bits 21 bits 23 bits

01100001011000100110, 001101100100011001010, 11001100110011101101000

And they should be connected as shown below:

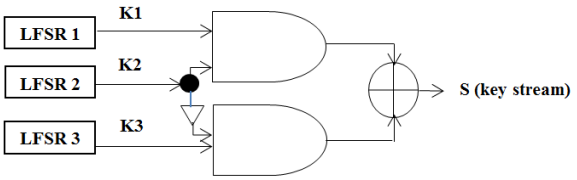


Fig. 1. Geffe Generator

The feedback functions are: K1= S10+S19, K2= S10+S20 and K3= S10+S22, successively. As an output of those registers, we will have three long binary sequences K1, K2 and K3, and each of them have the length 2^n-1 without repetition, which mean K1=1048575, K2=2097151 and K3=8388607. Then Geffe generator will mix them together and generate one binary sequence S have the length $(2^{K1}-1)(2^{K2}-1)(2^{K3}-1)$ = 18446715486418763775 without repetition. As shown in table I:

TABLE I. REGISTER'S PERIOD AND THEIR FEEDBACK FUNCTIONS

Bits	Feedback Function	Period
n		**2^n-1**
20	K1= $S_{10}+S_{19}$	1048575
21	K2= $S_{10}+S_{20}$	2097151
23	K3= $S_{10}+S_{22}$	8388607
64	S = $K_1\ K_2+K_2\ K_3+K_3$	$(2^{K1}-1)(2^{K2}-1)(2^{K3}-1)$ =18446715486418763775

Definitely if we generate this extreme large number (S) our system will slow down if not stop. So, we assume that we can generate sequences (S) with 1024, 512, 256 or 128 bits long, and that would be convenient to deal with. And after obtaining the binary sequence from Geffe generator, we will use three basic kinds of statistical test to check whether the randomness of the sequence is good or not. If the test successes we will continue with the next steps, else the system will ask the user to change his/her password. Finally, when the tests finished will decide which length could give us better results (1024, 512, 256 or 128).

B. Statistical tests

- Frequency test

For every binary random sequence, we expect that half of the sequence is 0's, and the other half is 1's, the purpose of this test relies on the number of 0's (n0) and the number of 1's (n1) in the sequence (n), which is we need to test. The static used is:

$$X1 = \frac{(n0-n1)^2}{n} \qquad (2)$$

To check whether the sequence passes this test or not, for one degree of freedom, the value of X_1 should be less than the acceptance threshold values of the test (X1 < 3.8415) [10].

- Serial test

This test depends upon the repetition of (n00, n01, n10, and n11) which denotes the numbers (00, 01, 10, and 11) in S respectively. And we expect that each of them represents nearly a quarter of n. The static used is:

$$X2 = \frac{4}{n-1}\sum_{i=0}^{1}\sum_{j=0}^{1}(\mathrm{nij})^2 - \frac{2}{n}\sum_{i=0}^{1}(ni)^2 + 1 \qquad (3)$$

To check whether the sequence passes this test or not, for two degree of freedom, the value of X_2 should be less than the acceptance threshold values of the test (X2 < 5.9915) [10].

- Poker test

This test is divide the sequence S into a number of blocks (K) have the length (M), and then check the repetition of those blocks and determine whether they appear approximately the same number of times as would be expected for a random sequence, the number of block's K=n/M (without fractions). The static used is:

$$X3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} ni^2\right) - k \qquad (4)$$

To check whether the sequence passes this test or not, for 2^m-1 degree of freedom, the value of X_3 should be less than the acceptance threshold values of the test (X3 < 14.0671) [10]. The poker test is a generalization of frequency test when we use m=1 we obtain results as frequency tests results.

In our work, we used those three tests to check the sequence S with 1024, 512, 256 and 128 bits long (with the current password), and the results are shown in the table below:

TABLE II. STATISTICAL RESULTS

Length of Pseudo	Statistical test	Value of statistical test	D. of freedom	Acceptance threshold
1024 its	Frequency test	0.03515625	1	3.8415
	Serial test	2.04011256720423	2	5.9915
	Poker test	9.19354838709677	7	14.0671
512 bits	Frequency test	0.0703125	1	3.8415
	Serial test	0.452192392367863	2	5.9915
	Poker test	7.2235294117647	7	14.0671
256 bits	Frequency test	1	1	3.8415
	Serial test	1.42745098039217	2	5.9915
	Poker test	7.51764705882353	7	14.0671
128 bits	Frequency test	5.28125	1	3.8415
	Serial test	4.56914370078741	2	5.9915
	Poker test	7.90476190476191	7	14.0671

As shown in Table II, three statistical tests have been used to check the randomness os the sequences. When the sequence S was 128 bits, it failed with frequency test (the sequence generated by password = abcdefgh does not pass the test). S equal to a 256 bit have been chosen, because it's the shortest successful length, and still can ensure that the sequence has a good randomness and small size. As illustrated in Table II, the frequency test increased every time when the bits' size decreased, and that was the reason behind stopping until 128 bits and chose 256 bits. The 256 bits sequence we got from the password is:
0100010011100100011001000001001000000101010000011111110111010101010101000011000100110001000111100111000000011010101001000010001111110011011001101010001000110101011100101001011011010000100101110000101011011111001001001001010110001111101110011101111100011101 0

This sequence derived from Alice password and it has passed all the tests. Next, convert every 8 bits into decimal number using ASCII code. As a result we got those numbers: 68 228 100 18 5 65 253 213 84 49 49 30 112 26 169 8 252 217 168 141 92 165 180 37 194 183 201 37 143 115 190 58, then modular each of them by 10 to ensure that all the numbers will be represented by one digit (1, 2, 3, 4, 5, 6, 7, 8 or 9). Merge them all together to get one number with 32 digits 88085533499026982781250743173508. Then assumed that the user's private key will be eight digits (the private key should be between 2 and P-2 [7], P will be provided by the server), therefore will divide the 32 digits number into eight blocks and take one number from each block, and those eight digits will be the private key (take the first number from each block (randomly), the private number will be PrK=85422243). In this stage, every user has his/her own private key and the server has all of them together in one table.

Alice and Bob get two numbers g and p, where p is a prime and g (generator) is a primitive root Modulo p, and those two numbers are coming from the server with the public key, Alice will receive p, g, Bob public key and username, Bob will receive p, g, Alice public key and username. They will be encrypted using the receiver private key. As mentioned above, the private key should be less than the prime p [7]. Hence, our private key less than nine digits will satisfy the requirements. For our example let p=100000007 and g=5. The public key will be PuK=15071649. If Alice pw=abcdefgh and Bob pw=12345678 (a character string), then the private key of Alice PrK=85422243 and Bob PrK = 68203955. After mix them with p and g will get the PuK of Alice PuK=15071649 and Bob PuK = 6629794. As a result the shared key SK = 68202249 (for both Alice and Bob).

Noticeably, the central server would not use the existing identification technique such as X.509 for some reasons: X.509 certificate require digital signature to be used for integrity, thus it has the same problems of using DS [11]. In addition X.509 certificate has problem with certificates expire, sometimes no one knows that the certificate has been expired until the website or sever goes down. It is really hard to figure out what is going on.

C. Algorithm

- Registration time

1. The system converts the eight character password into a 64 bit binary sequence using ASCII code.
2. The binary sequence is divided into three registers with lengths of 20, 21, and 23 respectively as indicated in section III. The Geffe pseudo random sequence generator is used to generate one sequence with 256 bit.
3. The resulting sequence, S is tested using the three basic statistical tests frequency, serial and poker:
    a. If the sequence passes all the tests, then continue,
    b. Else, go back to step 1 and ask the user to try another password.
4. Convert the successful sequence into a decimal number. This is then divided into eight blocks to obtain the eight digit private key using a digit from each of the block.
5. The server uses the same method to calculate PrK1 and PrK2.
6. The administrator sends an activation message to Alice and Bob asking them to activate their passwords:
    a. Alice and Bob receive the activation message, and then continue,
    b. Else, Pw1 and Pw2 have changed by Eve. Go to 15
7. The server hashs Pw1and Pw2, encrypts PrK1 and PrK2 using the admin key, and add salt (random number) then saves these in one table: the user info table.

- Log in time

8. The server redirects Alice and Bob to the second page and provides them with p, g, PuKs and usernames. These will be encrypted using the received private key where p and g will change for every session. Alice and Bob will need their password to access the information. The public keys are calculated using the following equations [3]:

$$PuK_1 = g^{PrK1} \bmod p \quad (5)$$

$$PuK_2 = g^{PrK2} \bmod p \quad (6)$$

9. The server redirects Alice and Bob to the second page and provides them with p, g, PuKs and usernames. These will be encrypted using the receiver private key. Alice and Bob have to use their password to access the information.
10. Alice and Bob compute their shared key (SK), and the server also calculates shared key (SK). The following equations are used to calculate the shared keys [3].

$$SK_{Alice} = PuK_2^{PrK1} \bmod p \quad (7)$$

$$SK_{Bob} = PuK_1^{PrK2} \bmod p \quad (8)$$

11. The server hashes the SK and saves it in the user info table.
12. Alice and Bob hash their shared keys (Sk) and send H(Sk) to the server.
13. The server checks the resulted hashed shared key H(Sk) with the hash table:
    a. If H(Sk) matches with one of the hashes in the hash table, then continue.
    b. Else, Sk has come from Eve. Go to 15
14. Alice and Bob are ready to share their messages encrypted with the shared key. Go to 16
15. Warning message: "Your data were compromised".
16. End.

TABLE III. OBTAINED DATA

	Alice	**Server**	**Eve**	**Bob**
1	Alice-username	Alice-username & Bob-username	H(Alice, Bob) username	Bob-username
2	Pw1, PrK1	Pw1, Pw2, PrK1 & PrK2	H(Pw)	Pw2, PrK2
3	P, g	P, g	E(P, g, PuKs, username)	P, g
4	PuK1	PuK1, PuK2	--	PuK2
5	SK	SK	--	SK

As shown in Table III, Eve can only obtain hashed and encrypted data and she cannot use them to intercept the communication channels.

To sum up, the MITM attacks occur when we share our keys in plain text. Eve can sit in the middle and pretend she is the intended destination for both Alice and Bob, Alice and Bob have no way of knowing Eve is there and believe they are communicating directly with each other. The proposed method proved that Alice and Bob could generate the keys without sending them in plain text using the data obtained from the server, so that will be controlled by the server. Even though some may argue that Eve can compromise all the data on the channels, but that will be not enough for her to get the shared key, all the data will be hashed and encrypted. In addition, the private key will never be sent to any party. If an attacker steal the private key from the server it would be not useful for him/her due to the entire private keys in the server are encrypted using the admin key. This offers high levels of security and helps prevent MITM attacks in DH. There are many applications used DH as a key exchange like SSL (Secure Sockets Layer), Secure Shell (SSH) and IP Security (IPSec) [12]. The proposed infrastructure could distribute the keys in secure manner and these applications would benefit from the proposed infrastructure.

IV. CONCLUSION

An efficient method to harden the Diffie-Hellman protocol against man-in-the-middle attack was described within this paper. Geffe generator was used to generate a binary sequence with a high level of randomness. Furthermore, statistical tests are used to check these sequences, before calculating the private key and the shared key. The proposed method ensures that the private keys will not be sent through the channels, and will be saved as hashes in the server. Additionally, it provides a non-repudiation property, as it can identify the sender and the receiver from their user information. As a result, this method provides more security properties than existing methods and prevents MITM attack. In the future, we will employ this method over other encryption methods to provide a secure cryptosystem for sharing our messages securely. The researchers are planning to employ the proposed algorithm in real cloud cryptosystem.

REFERENCES

[1] N. Kumar, P. Gupta, M. Sahu, and M. Rizvi, "Boolean Algebra based effective and efficient asymmetric key cryptography algorithm: BAC algorithm," in Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on, 2013, pp. 250-254.

[2] W. Diffie and M. E. Hellman, "New directions in cryptography," Information Theory, IEEE Transactions on, vol. 22, pp. 644-654, 1976.

[3] C. K. Kumar, G. J. A. Jose, C. Sajeev, and C. Suyambulingom, "Safety measures against man-in-the-middle attack in key exchange," Asia Research Publishing Network (ARPN) Journal of Engineering and Applied Sciences, vol. 7, pp. 243-6, 2006.

[4] M. K. Ibrahem, "Modification of Diffie-Hellman key exchange algorithm for Zero knowledge proof," in Future Communication Networks (ICFCN), 2012 International Conference on, 2012, pp. 147-152.

[5] J.-h. Chen, Y. Long, K.-f. Chen, Y.-t. Wang, and X.-x. Li, "An efficient threshold key-insulated signature scheme," Journal of Shanghai Jiaotong University (Science), vol. 16, pp. 658-662, 2011.

[6] F. Boudrez, "Digital signatures and electronic records," Archival Science, vol. 7, pp. 179-193, 2007.

[7] B. Preneel, C. Paar, and J. Pelzl, Understanding cryptography: a textbook for students and practitioners. London: Springer, 2009.

[8] N. Ferguson and B. Schneier, Practical cryptography vol. 141: Wiley New York, Indianapolis, 2003.

[9] S. Wei, "On Generalization of Geffe's Generator," IJCSNS August, vol. 6, pp. 161-5, 2006.

[10] S. M. Hosseini, H. Karimi, and M. V. Jahan, "Generating pseudo-random numbers by combining two systems with complex behaviors," Journal of Information Security and Applications, pp. 149–62, 2014.

[11] T. Cristian, "Security Issues of the Digital Certificates within Public Key Infrastructures," Informatica Economica, vol. 13, pp. 16-28, 2009.

[12] M. Ahmed, B. Sanjabi, D. Aldiaz, A. Rezaei, and H. Omotunde, "Diffie-Hellman and Its Application in Security Protocols," International Journal of Engineering Science and Innovative Technology (IJESIT), vol. 1, pp. 69-73, 2012.

An Efficient Hash Based Parallel Block Cipher Mode of Operation

Aqeel Sahi*
Computer Centre
Thi-Qar University
Thi-Qar, Iraq
e-mail: akeel_sahy@yahoo.co.uk

David Lai, and Yan Li
*Math and Computing Department
University of Southern Queensland
Toowoomba, Australia
e-mail: david.lai@usq.edu.au, yand.li@usq.edu.au

***Abstract*—Block cipher encryption works on fixed length blocks, usually 128bits. The blocks of data are transformed into encrypted data blocks of identical size using a shared session key. A common feature of some modes of block cipher, such as Cipher Block Chaining mode (CBC), Cipher Feedback mode (CFB), Output Feedback mode (OFB), is the sequential processing. The ciphering process of a block cannot begin until the processing of the preceding block is completed. This feature does not make full use of the processing power in multiple processor systems. In this paper, we proposed a Parallel Block Cipher (PBC) mode, in which blocks of cipher can be processed in parallel. Results of speed performance tests of the PBC mode using various settings are presented and compared with the standard CBC mode. The PBC mode was shown to save 60% of execution time when compared with the CBC mode. Furthermore, the hash value of the data file might be utilized to provide integrity check in addition to encryption using AES128. As a result, the PBC mode has a better speed performance on top of the confidentiality and security provided by the CBC mode.**

Keywords-block cipher; cryptography; data encryption; mode of operation; parallel processing

I. INTRODUCTION

Block cipher is a generic process in which data are handled in fixed size blocks. It can be used to handle a single chunk as well as a stream of data. Different ways to handle the data blocks will end up in different modes of operation for the block cipher. There are five modes of operation for block cipher that have been approved by the National Institute of Standards and Technology (NIST of USA) [1] to be used for encryption: Electronic Codebook mode (ECB), Cipher Block Chaining mode (CBC), Cipher Feedback mode (CFB), Output Feedback mode (OFB) and Counter mode (CTR).

The Electronic Codebook (ECB) mode is the simplest mode among the encryption modes. The plaintext is split into blocks, and each block is processed independently [2]. The mathematical formulas for the ECB are:

$$C_i = E_k(P_i) \quad (1)$$

$$P_i = D_k(C_i) \quad (2)$$

The Cipher Block Chaining (CBC) mode was brought to us in 1978 by Ehrsam, Meyer, Smith and Tuchman [3]. In the CBC mode, each plaintext block is XOR with the previous ciphertext block before being encrypted. To be probabilistic, an initialization vector (IV) should be utilized in the first block [4]. The mathematical formulas for the CBC are:

$$C_i = E_k(P_i \oplus C_{(i-1)}) \quad (3)$$

$$P_i = D_k(C_i) \oplus C_{(i-1)} \quad (4)$$

where, C_0= IV

The Cipher Feedback (CFB) mode uses the previous ciphertext as input for encryption. The encryption output is XOR with the current plaintext block to produce the ciphertext of the current block. The mathematical formulas for the CFB are:

$$C_i = E_k(C_{(i-1)}) \oplus P_i \quad (5)$$

$$P_i = D_k(C_{(i-1)}) \oplus C_i \quad (6)$$

where, C_0= IV

The Output Feedback (OFB) mode uses block cipher as a key stream generator. The OFB is XOR the plaintext blocks with the generated stream to obtain the ciphertext. Since the XOR operation results in symmetry, encryption and decryption are exactly the same. The mathematical formulas for the OFB are:

$$C_i = P_i \oplus O_i \quad (7)$$

$$P_i = C_i \oplus O_i \quad (8)$$

$$O_i = E_k(I_i) \quad (9)$$

where, $I_i = O_{(i-1)}$ & I_0=IV

The counter (CTR) mode was introduced in 1979 by Whitfield Diffie and Martin Hellman [5], it is similar to the OFB, and suitable to be implemented on multiprocessor devices where many blocks can be encrypted in the same time. Furthermore, successive values of a counter are applied to a block cipher encryption process to generate the key stream.

Another mode of encryption was proposed by Institute of Electrical and Electronics Engineers (IEEE) [6] and recommended by NIST of USA [7] called XTS. XTS mode is XEX-based tweaked-codebook mode with ciphertext stealing.

Generally, cryptography could be the best solution for data security [7]. In fact, cryptography can help improve data security from different aspects: firstly, the provider of services cannot acquire any information about encrypted user data. Secondly, if user data are corrupted by an attacker, the corruption can be easily identified by the user himself/herself [8]. Also, cryptography applications can easily be installed in computers, smart phones and other mobile devices so users can conveniently protect and share their data with trusted parties. Thus, cryptosystem can provide authentication,

978-1-5386-6349-3/18/$31.00 ©2018 IEEE

confidentiality and integrity, as well as trusted data sharing [8].

There are two main cryptographic schemes, deterministic encryption and probabilistic encryption. For deterministic encryption, encrypting identical plaintext multiple times yield identical ciphertext every time. On the other hand, encrypting identical plaintext many times using probabilistic encryption yield different ciphertext every time even if the same encryption key is used (and different Initial Vectors (IV) are used). For example, the ECB mode uses deterministic encryption while the CBC mode uses probabilistic encryption.

Deterministic encryption is vulnerable to substitution attack, which means it is easy to attack the ciphertext if the same message is sent twice [7]. Therefore, Goldwasser and Micali brought in probabilistic encryption in 1984 [9]. This is a scheme for encryption where a plaintext is encrypted into one of many possible ciphertext (not only a single ciphertext as in deterministic scheme) and is free from substitution attack [10].

Some people have concerns about putting all of their data and files in online servers. If a hacker can launch a successful attack to an exposed service, he/she may be able to get all the user's information. Furthermore, if the hacker tries to use or resell the stolen information, it would be a privacy and security catastrophe. To overcome these issues, service providers often use secure and reliable encryption techniques to secure their systems. While cryptography can offer a high level of data security, there are security concerns [11]. Cryptanalysis also shows some weaknesses in various block cipher modes, such as the sequential nature of the CBC mode [12]. In response, a parallel block cipher mode was developed [13].

II. Related Works

The five approved block cipher modes (ECB, CBC, CFB OFB and CTR) are divided into two types. The ECB and CBC modes use block cipher for encryption, while the CFB, OFB and CTR use block cipher as a key stream generator.

Each of these five modes has its own shortcomings. Deterministic encryption is the biggest issue in the ECB mode which leads to a substitution attack [7], as shown in the following example.

Assume the data shown in Figure 1 is to be transferred between banks. Let us assume that each of the fields has exactly the same size of a block used in the block cipher (16 bytes in the case of AES). And the encryption key shared between the two banks does not change too frequently. Due to the nature of the ECB mode and deterministic scheme, an attacker can easily substitute block number 4 (which contains the receiving account number) without deciphering with the ciphertext block with his own account number. As a consequence, all transfers from bank A to bank B will be redirected to the attacker's account. Interesting enough, the attack works without having to attack the block cipher (e.g. AES) itself.

The CBC mode does not allow the encryption process to be parallelized to make it work faster as there is reliance on the previous ciphertext block [12]. Furthermore, in the CBC mode, chaining between the blocks means one corrupted cipher block will result in two corrupted blocks [14]. Example of breaking the CBC chain is shown in table I.

The CFB and OFB modes have similar issues as the CBC mode, in that they cannot parallelize their encryption processes to make them faster [7]. Therefore, they will struggle in terms of speed, especially with big data. In addition, there is a common problem for the CFB, OFB and CTR stream modes, it is the flipping bits. If a bit flip error arises in ciphertext in a particular block, then after decryption, the error is limited to the exact bit of the exact block of plaintext [15]. Flipping a ciphertext bit flips the corresponding plaintext bit. The attack aims to change a bit in the decrypted plaintext by flipping a bit in the ciphertext thereby changing the decrypted plaintext.

For example, assume an attacker knows the ciphertext for an electronic fund transfer, which contains the ASCII string "$1000.00" for the transfer amount. The attacker can alter the amount to "$9000.00" by XOR the part of cipher text for the string with the result of ("$1000.00" ⊕ "$9000.00"):

"$1000.00" ⊕ ("$1000.00" ⊕"$9000.00")
= "$1000.00" ⊕ "$1000.00" ⊕ "$9000.00"
= "$9000.00"

Note that:
"$1000.00" ⊕ "$1000.00" = all zeros
All zeros ⊕ "$9000.00" = "$9000.00"

In addition, there is no chaining between blocks in the CTR mode. Thus, if Eve (an attacker) modifies any block, Bob (a receiver) will never know about that. Moreover, the CTR requires synchronization of the initial counter value between the encryption party and decryption party. A policy to ensure that the next communication's initial counter value is unique and correct should be in place and enforced. This might require a private communication channel [16].

XEX based Tweaked-codebook mode with ciphertext Stealing (XTS) mode is developed based on the XOR Encrypt XOR mode (XEX) [17] with Ciphertext Stealing mode (CTS) feature. However, it is slower than the CBC mode due to more mathematical operations (such as XOR and multiplication) [18]. XTS mode uses XOR function.

Blocks

1	2	3	4	5
Sending Bank A ***ID***	Sending Account ***#***	Receiving Bank B ***ID***	Receiving Account ***#***	Amount ***$***

Figure 1. Example for a substitution attack against the ECB mode

TABLE I. Example of Breaking the CBC Chain

CBC Encryption:	**CBC Decryption:**	**If C_1 is corrupted to X then:**
$C_0 = E_k (P_0 \oplus IV)$ **$C_1 = E_k (P_1 \oplus C_0)$** **$C_2 = E_k (P_2 \oplus C_1)$** **$C_3 = E_k (P_3 \oplus C_2)$** **$C_4 = E_k (P_4 \oplus C_3)$**	$P_0 = D_k (C_0) \oplus IV$ $P_1 = D_k (C_1) \oplus C_0$ $P_2 = D_k (C_2) \oplus C_1$ $P_3 = D_k (C_3) \oplus C_2$ $P_4 = D_k (C_4) \oplus C_3$	$P_0 = D_k (C_0) \oplus IV$ $P_1' = D_k (X) \oplus C_0 \neq P_1$ $P_2' = D_k (C_2) \oplus X \neq P_2$ However: $P_3 = D_k (C_3) \oplus C_2$ $P_4 = D_k (C_4) \oplus C_3$

Before and after each encryption process (like XEX), and uses two keys instead of one key [19]. Since there is no built-in mechanism to detect alterations (an active attack) [4], any altered ciphertext (by attacker) will be decrypted to some legitimate plaintext.

Some may argue that the CTR_CBC MAC mode (CCM) can provide an integrity check in addition to performing encryption [20]. However, the inherently sequential nature of the CBC mode, and the need for two passes with AES, can make CCM AES a slower scheme [21]. CCM mode requires a longer processing time than the CBC mode (CBC is a special case of CCM).

A Galois/Counter mode of operation (GCM) was proposed by McGrew and Viega in 2004 [22]. The GCM mode has been approved by the NIST of USA as an authenticated mode of operation [23]. The GCM mode ensures data confidentiality using the CTR mode. It delivers guarantee of the authenticity of the confidential data using a hash function which is defined over a binary Galois field [23]. However, while the CTR mode is a part of the GCM mode, therefore it could be vulnerable to the flipping bits attack which we mentioned above. In addition, the GCM mode is vulnerable to replay attack [23], in other words, similar to numerous authentication modes, GCM does not naturally stop an attacker from interrupting the message and replaying it for authenticated decryption at a later time, for example, in an attempt to imitate a party that has access to the key.

The Extended Codebook (XCB) Mode of Operation was proposed by McGrew and Fluhrer in 2004 [24]. The XCB was the first mode that constructed from the hash CTR hash. The XCB used the CTR mode of operation in order to ease tackling variable length messages. However, Chakraborty and Sarkar reported that the XCB has limitation [25]. While the XCB mode adopt the ciphertext block as a key, therefore, the key length and the length of the block must be identical [25]. Thus, it is impossible to use the XCB mode once the length of the key is different from the length of the block, as for example in AES with 192bit key and 128bit block.

Parallel ciphering can improve the speed performance of the encryption. Big data usually stored in distributed locations, therefore parallel encryption is needed in order to encrypt the large number of blocks or chunks of data separately, which takes much longer when using sequential encryption.

III. PARALLEL BLOCK CIPHER MODE

Data encryption is one of the most common practical choices for computing developers and database designers [26] to protect data. The biggest database management systems like Microsoft's SQL Server and Oracle adopted the CBC mode in block cipher algorithms [27], as well as most of cryptosystems nowadays adopting the CBC in their block cipher. Because of its popularity, we choose the CBC as the sequential block cipher mode to compare with Parallel Block Cipher (PBC) mode, was briefly presented in 2015 [13]. Before we present the comparison, we introduce the PBC briefly Below.

The PBC mode parallelizes the encryption process. It converts plaintext blocks to ciphertext blocks without any chaining occurring between data or outputs from another block. The hash of the file is utilized to generate encryption and decryption keys. Multiple blocks can be encrypted in one round and each plain text block does not need to wait for outputs from a previous block to start or complete the encryption process, making parallelization possible. The proposed PBC mode makes use of 128 bit key AES and hash function [28]. Encryption diagram is shown in Figure 2.

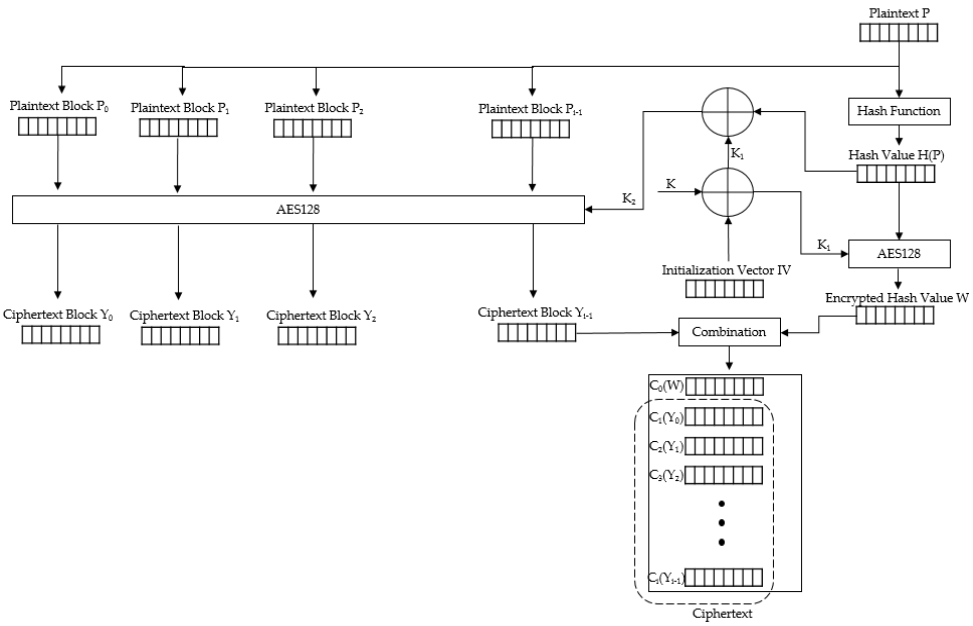


Figure 2. The PBC encryption diagram.

The following steps are used to encrypt data using the PBC mode:

Step 1. $P \xrightarrow{hash(P)} H(P)$

Step 2. First of all, the plaintext *P* will be hashed to obtain the hash value of the data *H(P)* in order to mix it with the key. $H(P) \xrightarrow{E_{K1}(H(P)),\ AES128} W$

$$P_i \xrightarrow{E_{K2}(P_i),\ AES128} Y_i$$

In the second step, we encrypt the hash value *H(P)* using *K1 = IV ⊕ K* (where *K* is the session key) to obtain W. then, encrypt the plaintext P_i using *K2 = ((IV⊕ K) ⊕ H(P))* to yield the ciphertext of the data Y_i.

Step 3. $W + Y_i \xrightarrow{Combination} C_i$

Finally, we combine *W* and Y_i to produce the complete ciphertext C_i which involves the ciphertext of the data and the hash value.

The following steps are used to decrypt data using the PBC mode:

Step 1. $C_i \xrightarrow{Extraction} W + Y_i$

To decrypt the ciphertext C_i using the PBC mode firstly we have to extract the ciphertext of the hash value *W* which is located in the first block of the ciphertext, the rest of the ciphertext is Y_i.

Step 2. $W \xrightarrow{D_{K1}(W),\ AES^{-1}128} H(P)$

Secondly, we will decrypt the ciphertext of the hash value *W* to retrieve a copy of the original hash value *H(P)*.

Step 3. $Y_i \xrightarrow{D_{K2}(Y_i),\ AES^{-1}128} P_i$

Then decrypt using K_2 the rest of the ciphertext Y_i to retrieve P_i which represent the original data.

Step 4. *Integrity check*, $H_2(P) = H(P)$, (optinal)

Furthermore, the PBC mode facilitates a message integrity check (step 4). The PBC mode can check the integrity of the message by comparing the hash of the copy of the original file which derived from Ci with H(P). This step is optional, because the proposed PBC mode is an encryption mode and not an authenticated encryption mode.

Decryption diagram is shown in Figure 3.

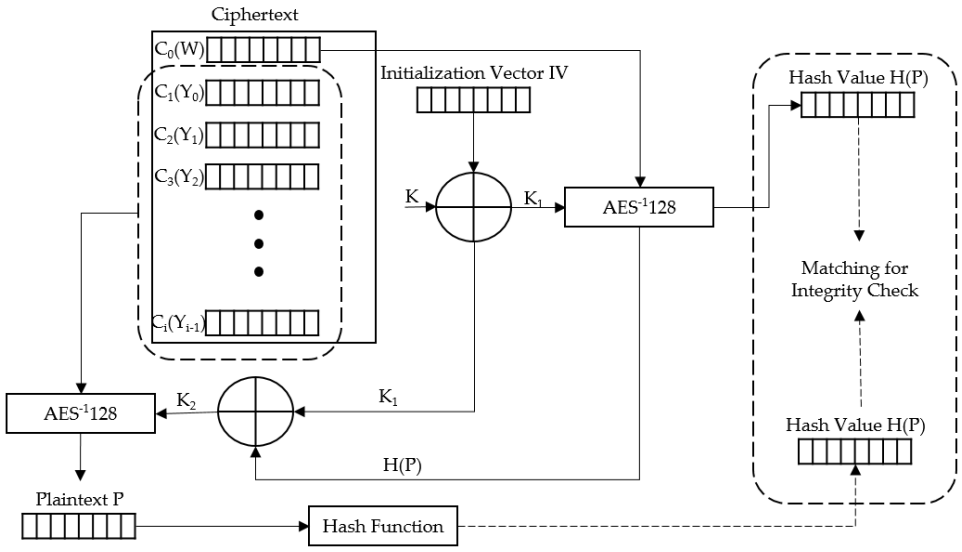


Figure 3. The PBC decryption diagram.

IV. IMPLEMENTATION

Microsoft Visual Studio (MVS) 2010 and Dot Net Framework 4.5 were used to implement the CBC mode and the PBC mode. Two console applications were developed. The first one for the execution of the CBC mode and another one for the execution of the PBC mode. We have used System.Security.Cryptography package to utilize cryptographic services in MVS such as secure encryption and decryption of data using AES, as well as calculating hash value hash function.

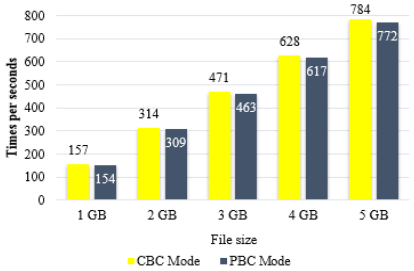
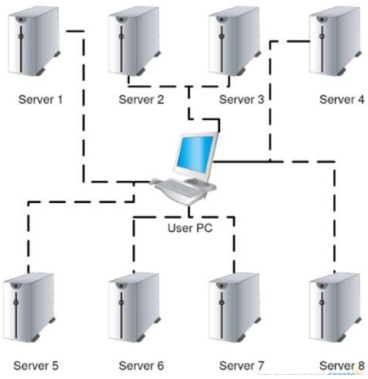
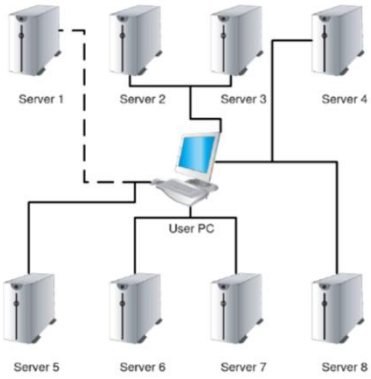


Figure 4. Execution times of the PBC and the CBC modes using single process.



(a) The PBC servers' box (multiple servers are running)



(b) The CBC servers' box (single server is running)

Figure 5. PBC and CBC servers' boxes.

To parallelize the PBC mode, a Task Parallel Library (TPL) was also used. The TPL is a set of public types and Application Programming Interfaces in the System.Threading and System.Threading.Tasks packages. The TPL aims to simplify the process of adding parallelism to applications and efficiently use all the CPUs that are available. In our case, the TPL can let multiple cores computer to run eight CPUs at the same time to allow the program to run in parallel.

The executions of the sequential the CBC mode and the parallel PBC mode have been conducted using thirteen data samples with different sizes (100MB, 200MB, 300MB, 400MB, 500MB, 600MB, 700MB, 800MB, 900MB, 1GB, 1.1GB, 1.2GB and 1.3GB).

V. Results and Evaluation

A. Experimental Results

The PBC mode has been tested and compared with the CBC mode in three different scenarios. A single data file was used as input for the PBC mode which uses only one process. In the second scenario, the data file was manually split into data blocks and used as input for the PBC mode using multiple processes. Finally, a single data file is used as input for the PBC mode using multiple processes. In the last scenario, the PBC mode automatically splits the data file, processes the blocks, and recombines encrypted blocks to an encrypted data file.

1) Scenario 1

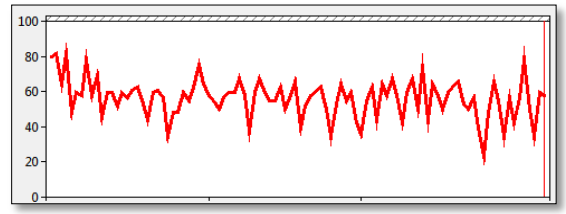
The PBC mode has been tested using a single process and compared with the CBC mode [13] in computers using Windows 7 and an Intel(R) Core(TM) i7 2600 CPU. CrypTool 2.1 and AES128 algorithm were used. The CBC_AES parameters were: action is encryption, key size is 128 bits, block size is 128 bites, chaining mode is the CBC, and padding mode is zeros. The PBC and CBC modes were running with a single process and a single input file was not split into blocks. The PBC mode was running like the CBC mode in a sequential manor. The results in Figure 4 indicate that the PBC mode is faster than the CBC mode even when both run on a single process. However, the time difference between the CBC mode and the PBC mode is quite small, it is only 1.61% on average using single process.

2) Scenario 2

When a single process is used, parallel processing for the data blocks were in fact not tested. So we proceed to test the PBC mode with input data being broken manually into 8 sub files. Each sub file is fed to a single process. Outputs from all eight processes were manually combined to generate the encrypted file. The time difference between the CBC mode and the PBC mode now increased to 19.61% on average. The PBC can save approximately one fifth of the execution time as compared to the CBC mode.

3) Scenario 3

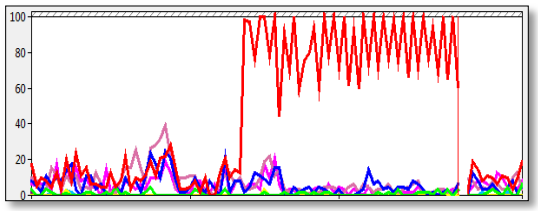
Nowadays, most of the high performance computers have more than one processor (CPU). We can treat those computers as multiple computers in a single box. This idea was adopted in this test.



(a). Single standalone server is running

(b) The execution result

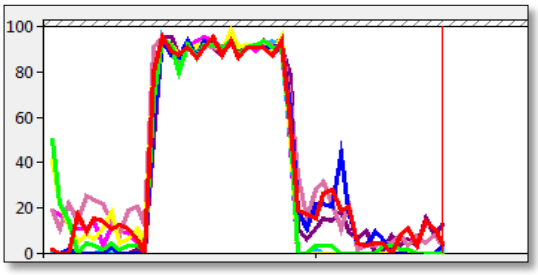
Figure 6. Running the CBC mode in virtual machine (CPU usage of the server).



(a) Single server running among eight servers

(b) The execution result

Figure 7. Running the CBC mode using Affinity Mask (CPU usage of the server).



(a) Eight servers running in parallel

(b) The execution result

Figure 8. Running the PBC mode on multiple servers (CPU usage of the server).

One computer with Intel(R) Core(TM) i7 2600 CPU (8 CPUs) and 16GB RAM adopted to perform this test.

Assumption 1. Assume that this computer has multiple processors (can be multiple servers) in a single box simulating eight servers, as shown in figs. 5a and 5b.

Assumption 2. Suppose every server mentioned in this paper has 1 CPU and 2 gigabyte RAM.

Limitation 1. Tests executed using data files with size ranging from 100MB to 1.3GB due to computer hardware limitation.

While the CBC cannot run in parallel [29] due to the block chaining feature, it will be run on a single server, as shown in Figure 5b. We have two options when using only a single server for a multiprocessor computer. The first option is to use virtual machine (VM) that runs on one server. Figure 6a shows the CPU usage when the CBC mode is run using of VM which uses one server. Encrypting 1.3GB file takes about 99.49 seconds (1:39.49), as shown in Figure 6b.

The second option is to use CPU Affinity Mask to force the multiprocessor computer to use a single CPU and 16GB RAM. As a result, encrypting 1.3GB file using the CBC mode takes about 92.84 seconds (1:32.84), as shown in Figure 7b. Which is very close to the previous result and the improvement may be attributed to the larger RAM used.

The PBC can be run processes in parallel using all the servers in the box (eight servers in this test), as shown in Figure 8a. The whole duration of hashing, breaking the file into parts, encrypting and combining the encrypted parts into one encrypted file using the PBC mode takes the average 41.4 seconds for a 1.3GB data file. The result is shown in Figure 8b, which is significantly better than the previous results. It shows that the PBC mode is more than twice as fast as the CBC mode.

Note that when we use 8 servers, the expected encryption time is around one eighth of that used by the CBC mode (approximately 13 s). The actual time used is around 17.42 s. With overheads for splitting the data file (14.68 s) and joining the cipher blocks (9.30 s), the total time span adds up to 41.40 s.

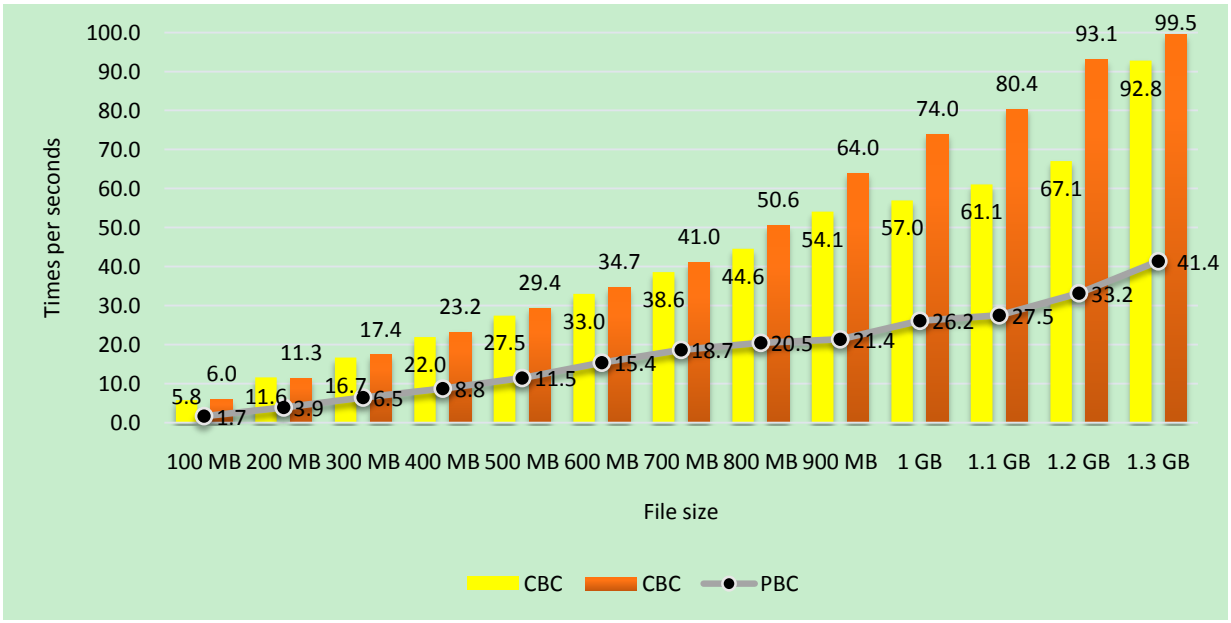


Figure 9. Execution times of the PBC and the CBC modes using multiple processes.

Multiple servers' tests were executed using data files from 100MB till 1.3GB, with an increment of 100MB for each data file. Execution time was measured in seconds. The percentage time saving between the PBC mode and the CBC mode is about 60% on average for all data file sizes. In other words, the PBC mode takes only 40% of the CBC mode execution time to encrypt the same amount of data. The test results were shown in Figure 9.

B. Discusions

The PBC mode is faster than the standard CBC mode in terms of execution time. As mentioned in the experimental results section, the proposed mode proves to be superior to the original CBC mode in terms of execution time under multiple servers' environment. As shown in Figure 9, the PBC can save up to 60% of the CBC mode encryption time.

Furthermore, the proposed PBC mode provides confidentiality. The PBC mode essentially has a similar concept of chaining blocks as in the CBC block cipher in terms of security performance. The PBC mode chains all the blocks together using the hash value of the plaintext. The key used to encrypt the plaintext blocks is generated using the hash value, therefore the hash value is used in encrypting every single block. If one cipher block is corrupted, then the corresponding plain text block will also be corrupted. The hash of the decrypted plaintext will differ from the original hash sent to the receiver. This will provide an integrity check on top of the confidentiality provided by encryption. Accordingly, the security thoughts of the CBC can be applied straight away to those of the PBC mode. Since the CBC mode of operation is revealed to be safe against numerous attacks such as chosen ciphertext attacks, ciphertext only and, known plaintext, we can claim that the proposed PBC mode is also safe against these attacks. As well as, the key stream of the CBC mode involves two inputs: key and IV. While the PBC mode key stream involves key, IV, and the hash value of all blocks, which add another degree of randomness.

Moreover, the proposed PBC mode has faster error recovery than the CBC mode. As mentioned in table I, in the CBC mode one ciphertext block corrupted will affect two plaintext blocks as a result. However, in the PBC mode one ciphertext block corrupted will affect the corresponding block only. As a result, in case of AES, corrupting 16 bytes block in the CBC mode affect 32 bytes blocks, and corrupting 16 bytes block in the PBC mode affect only 16 bytes block.

In relation to the CTR mode which is intended for parallelized processes and has been commonly utilized in high speed network standards. However, the PBC mode does not require counter synchronization. The unique values of IV and the hash deliver better security than the CTR mode.

VI. Conclusion

The new PBC mode considerably increases the speed of encryption process. In this mode, each cipher block makes use of the characteristics of the entire file (hash value of the file) instead of just the previous cipher block during encryption as in the CBC to improve the randomness of the key stream. This mode offers high levels of security, and better speed performance as compared to the standard CBC mode. In the future, the PBC mode can be tested in a real network and using a cloud environment with larger files.

Acknowledgment

Dr. Barbara Harmes is gratefully acknowledged for her help and support.

References

[1] W. Stallings, "NIST block cipher modes of operation for confidentiality," *Cryptologia*, vol. 34, pp. 163-175, 2010.

[2] M. Dworkin, "Recommendation for block cipher modes of operation. methods and techniques," NIST DTIC Document 2001.

[3] W. F. Ehrsam, C. H. Meyer, J. L. Smith, and W. L. Tuchman, "Message verification and transmission error detection by block chaining," ed: Google Patents, 1978.

[4] M. J. Dworkin, "Recommendation for block cipher modes of operation: The XTS-AES mode for confidentiality on storage devices," NIST Special Publication 2010.

[5] H. Lipmaa, P. Rogaway, and D. Wagner, "CTR-mode encryption," in *First NIST Workshop on Modes of Operation*, 2000.

[6] I. Std, "The XTS-AES tweakable block cipher. Institute of Electrical and Electronics Engineers," Inc. 2008.

[7] C. Paar and J. Pelzl, Understanding cryptography: a textbook for students and practitioners: Springer Science & Business Media, 2009.

[8] S. Kamara and K. Lauter, "Cryptographic Cloud Storage Financial Cryptography and Data Security," *Financial Cryptography and Data Security*, pp. 136-149, 2010.

[9] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of computer and system sciences*, vol. 28, pp. 270-299, 1984.

[10] G. J. Fuchsbauer, "An Introduction to Probabilistic Encryption," *Osječki matematički list,* vol. 6, pp. 37-44, 2006.

[11] N. Gonzalez, C. Miers, F. Redigolo, M. Simplicio, T. Carvalho, M. Näslund, *et al.*, "A quantitative analysis of current security concerns and solutions for cloud computing," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 1, p. 11, 2012.

[12] A. K. Beeputh, M. R. Doomun, and P. Dookee, "Energy-Security Adaptation Scheme of Block Cipher Mode of Operations," in *Innovations and Advances in Computer Sciences and Engineering*, ed: Springer, 2010, pp. 73-78.

[13] A. Sahi, D. Lai, and Y. Li, "Parallel encryption mode for probabilistic scheme to secure data in the cloud," in *Proc. 10th Int. Conf. Inf. Technol. Appl.(ICITA)*, 2015.

[14] M. Stamp, Information security: principles and practice: John Wiley & Sons, 2011.

[15] H. Lipmaa, D. Wagner, and P. Rogaway, "Comments to NIST concerning AES modes of operation: CTR-mode encryption," 2000.

[16] M. Tarhuni, S. Ng, A. Samsudin, and W. Ng, "Enhanced counter mode," in *Communications, 2003. APCC 2003. The 9th Asia-Pacific Conference on*, 2003, pp. 701-705.

[17] P. Rogaway, "Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC," in *International Conference on the Theory and Application of Cryptology and Information Security*, 2004, pp. 16-31.

[18] M. A. Alomari, K. Samsudin, and A. R. Ramli, "A parallel XTS encryption mode of operation," in *Research and Development (SCOReD), 2009 IEEE Student* Conference on, 2009, pp. 172-175.

[19] M. Liskov and K. Minematsu, "Comments on xts-aes," *Comments to NIST, available from their web page*, 2008.

[20] Z. A. Zukarnain, "Increase Throughput of CCM Security Mode Using MKP," *Applied Mathematics*, vol. 5, p. 581, 2014.

[21] P. Rogaway, "Evaluation of some blockcipher modes of operation," Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan, 2011.

[22] D. McGrew and J. Viega, "The Galois/counter mode of operation (GCM)," *Submission to NIST Modes of Operation Process*, vol. 20, 2004.

[23] M. J. Dworkin, "Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC," 2007.

[24] D. A. McGrew and S. R. Fluhrer, "The Extended Codebook (XCB) Mode of Operation," *IACR Cryptology ePrint Archive*, vol. 2004, p. 278, 2004.

[25] D. Chakraborty and P. Sarkar, "HCH: A new tweakable enciphering scheme using the hash-counter-hash approach," *IEEE Transactions on Information Theory*, vol. 54, pp. 1683-1699, 2008.

[26] V. Gampala, S. Inuganti, and S. Muppidi, "Data security in cloud computing with elliptic curve cryptography," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 2, pp. 138-141, 2012.

[27] S. Fazackerley, S. M. McAvoy, and R. Lawrence, "Gpu accelerated aes-cbc for database applications," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, 2012, pp. 873-878.

[28] Z. Wang, J. Graham, N. Ajam, and H. Jiang, "Design and optimization of hybrid MD5-blowfish encryption on GPUs," in *Proc. of International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, 2011, pp. 18-21.

[29] Y.-S. Yeh, T.-Y. Huang, and H.-Y. Lin, "Structural Binary CBC Encryption Mode," *Journal of Information Science and Engineering*, vol. 25, pp. 937-944, 2009.

Parallel Encryption Mode for Probabilistic Scheme to Secure Data in the Cloud

Aqeel Sahi, David Lai, and Yan Li

Abstract-- **Cloud computing is a model for using computer resources and technologies to provide services such as storage and applications. Users can access and use the cloud computing services without the need to acquire knowledge, expertise or even administration of infrastructures that support these services. Cloud computing can be a general concept which includes software services and other modern technological functionality in the information technology world. Since cloud computing shares distributed resources through the Internet and intranet in the exposed environment, security is therefore an important issue. Cryptography is one of the common practical choices for cloud computing developers. It can offer a high level of security. However, cryptanalysis shows some weaknesses in block cipher modes, such as parallelization for Cipher Block Chaining (CBC) mode. This paper will explore the security issues of the standard block cipher modes of operation as well as in cloud computing. In addition, a new mode of encryption called the parallel encryption mode will be presented. In this mode each cipher block makes use of the characteristics of the entire file instead of just the previous cipher block. Therefore, if any single plaintext block is changed, all the cipher blocks will be changed. This mode offers high levels of security, integrity and enhances the performance of the block cipher in terms of speed as compared to standard CBC mode.**

Index Terms-- **Cloud Computing, Data Security, Modes of Encryption and Encryption Schemes.**

I. INTRODUCTION

First of all, we have to clarify that cloud computing is not a new technology but a new form of service. To understand this we must put forward a simplified definition of cloud computing which states that cloud computing is a model for using computing resources (Hardware and Software) via the Internet to introduce to us as a service. Users do not have to know or master the concept of how this service operates, runs, gets connected, and what software is installed on it. There are three main types of services offered by the cloud: Software as a Service (SaaS); Platform as a Service (PaaS); and Infrastructure as a Service (IaaS) [1]. SaaS means using software which is already stored in the cloud such as YouTube. PaaS is using the cloud as a platform to install applications and operating systems in a platform to ensure homogeneity between applications such as the Google app Engine. IaaS is using the cloud as infrastructure (Processor, memory, hard drive and limited number of users). Users are free to use these components in any way such as the Amazon web service. Fig. 1 shows the Cloud Computing Service Models with their examples. In addition, four deployment models have been identified for cloud architecture solutions, namely Private cloud; Community cloud; Public cloud and Hybrid cloud [2]. The cloud infrastructure of the Private cloud is functioned for a private organization. Private cloud can be managed by the same organization (organization that owns the cloud) or by external organization. The cloud infrastructure of the Community cloud provides services to a number of organizations and supports a specific community that has shared interests. Community cloud can be managed by the same organization (organization that owns the cloud) or by a third party. The cloud infrastructure of the Public cloud is made accessible to the universal public or an enormous organizations group and is managed by an organization offering cloud services. The cloud infrastructure of the Hybrid cloud is a mixture of two or more clouds (private, community, or public). Hybrid cloud is usually used by organizations that use small amounts of data or need special applications.

Cloud computing has many advantages over traditional computing such as: easy access to most of the applications we need regardless of time or place. Reducing the expense of buying new high performance computers and hardware; ensuring continuity of service. Cloud computing organizations work 24/7. However, there are security concerns. Some people have concerns about putting all of their information and files in the cloud. If a hacker can launch a successful attack to an exposed service, he/she may be able to get all the user information.

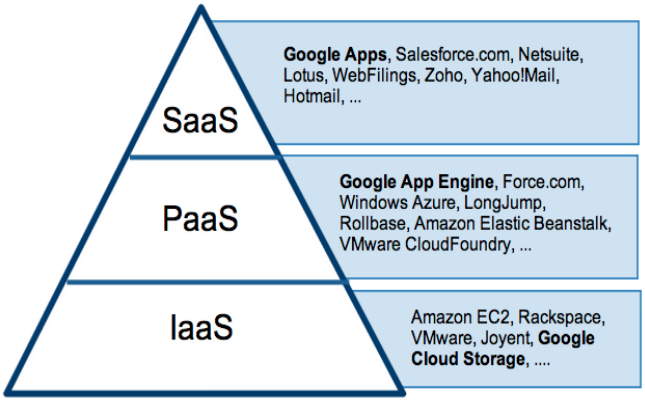


Fig.1. Cloud computing service models [3].

ISBN: 978-0-9803267-7-2

Also, if the hacker tries to use or re-sell the stolen information, it would be a privacy and security catastrophe. To overcome these issues, cloud service providers often use secure and reliable encryption techniques to secure their systems.

The two main encryption schemes are Deterministic encryption and Probabilistic encryption. For Deterministic encryption, encrypting identical plaintext multiple times will yield identical ciphertext every time. On the other hand, encrypting identical plaintext many times using Probabilistic encryption yield different ciphertext every time if the same encryption key is used in all cases. These two encryption schemes are used by many types of modes of encryption in different situations. For example, Electronic Code Book mode uses Deterministic encryption while Cipher Block Chaining mode uses Probabilistic encryption. This paper presents a new cipher block mode for Probabilistic encryption to secure data in the cloud.

II. SECURITY ISSUES

A. Standard Modes of Encryption

There are five modes of encryption for block cipher that have been approved by the National Institute of Standards and Technology (NIST) [4]: electronic codebook mode (ECB), cipher block chaining mode (CBC), cipher feedback mode (CFB), output feedback mode (OFB) and counter mode (CTR). Each of these five modes has its own shortcomings. Firstly, deterministic encryption is the big issue of ECB mode, which means if you encrypt the same plaintext blocks more than once you will have identical ciphertext blocks, while the key is not changed. Subsequently, this will lead to substitution attack against ECB mode [5]. Secondly, the encryption process for CBC mode cannot be parallelized to make it faster, as there is reliance on the previous ciphertext block [6]. The ECB and CBC modes use block cipher for encryption, while CFB, OFB and CTR use block cipher as a key stream generator. CFB and OFB modes have similar issues to CBC mode, in that they cannot parallelize their encryption processes to make them faster. Therefore, they will struggle in terms of speed, especially with big data. In addition, there is a shared problem between CFB, OFB and CTR stream modes which called flipping bits. If a bit flip error arises in ciphertext in particular block, then after decryption, the error is limited to the exact bit of the exact block of plaintext [7]. Flipping a ciphertext bit flips the corresponding plaintext bit. In addition, in CTR mode, there is no chaining between the blocks. Thus, if Eve modifies any block, Bob will never know about that. Moreover, the difficulty in the execution of CTR is in the deflection of the counter value between the encryption party and decryption party, both of them needing a policy for counter synchronization to ensure that the next communication's initial counter value is unique and correct this might require a private communication channel [8]. In cryptosystems, the integrity of the message is critical for satisfactory security, yet all five standard modes do not provide this feature.

B. Cloud Computing

Nowadays, cloud computing is used by millions of people around the world. Cloud computing gives users the opportunity to store data in the cloud for easy access anytime and anywhere. However, cloud computing poses many security issues with regard to confidentiality and integrity. For data distribution in the cloud, both confidentiality and integrity are vital in order to avoid data from being vulnerable to unauthorized attacks [9]. In terms of confidentiality, in the cloud environment, user's data is controlled by service providers and not by users themselves. The potential for data leaks is real, either intentionally or accidentally, which is unacceptable [10, 11]. Furthermore, data in the cloud is stored in geographically diverse locations; therefore confidentiality becomes a big concern [12]. Once users use cloud services, they perhaps won't know precisely where their information is stored [13]. In this situation, it is better for the cloud service providers to offer high levels of encryption techniques to secure the data wherever it is. In relation to integrity, confidentiality is not enough to guarantee cloud computing security. Users want to ensure that their data has not being improperly modified or compromised by a third party. Cloud service suppliers should apply methods to guarantee data integrity [1]. In addition, as the data is stored in diverse locations in a cloud, the access control methods need to be completely safe and every user needs to be verified as an authentic user [12].

III. PROPOSED WORK

It is well known that there are several ways to protect files in the cloud. Data encryption could be the most common practical choice for cloud computing developers and database designers[13].The biggest database management systems like Microsoft's SQL Server and Oracle use CBC mode in block cipher algorithms [14, 15]. However, in CBC mode chaining between the blocks necessarily means that one corrupted cipher block will result in two corrupted blocks [16]. Furthermore, CBC cannot parallelize the encryption process to make it any faster. Thus, we propose a parallel encryption mode for encrypting files in the cloud. This parallel encryption mode parallelizes the encryption process and provides integrity. It is encrypting plaintext blocks to produce ciphertext blocks without any chaining occurring between plaintext and ciphertext. The hash of the file is used to encrypt and decrypt the blocks, thus changing one block will affect the whole file. Each plaintext block does not need to wait for the previous ciphertext block to start the encryption, making parallelization possible. The proposed parallel encryption mode has utilized AES128bit key and MD5 (HF) together.

We chose AES128 to match with the output of MD5. MD5 produces a 128bit message digest [17] and the proposed parallel encryption mode uses 16 byte block. The Initial Vector IV should be known by both parties (Alice and Bob) but unpredictable for a third party (Eve). In particular, the IV should be unpredictable and not related to the plaintext.

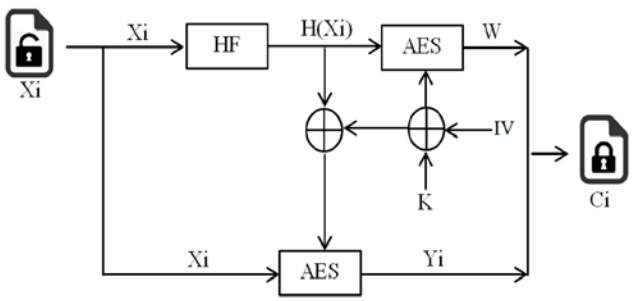


Fig. 2. Encryption process.

$$W = E_{(K \oplus IV)}(H(X_i)) \quad (1)$$
$$Y_i = E_{(K \oplus IV) \oplus H(X_i)}(X_i) \quad (2)$$
$$C_i = W + Y_i \quad (3)$$

Thus, encrypting files using the Parallel Encryption mode requires the following steps. Firstly, file X will be hashed using MD5 to obtain the hash value H(Xi). Secondly, IV will XORed (Exclusive or) with the key (K) and the result will XORed with the hash value H(Xi), ((IV⊕ K) ⊕ H(Xi)).The result of these two XOR functions will be used as a key input to the AES algorithm (shown as 'E' in equations 1 and 2) to encrypt the file and obtain the ciphertext (Yi) (shown in equation 2). At the same time, the hash value H(Xi) will be encrypted using the AES algorithm and using IV ⊕K as a key input to obtain an encrypted form of the H(Xi) (shown as 'W' in the diagram and equation 1). Finally, W will be sent together with the rest of cipher blocks (Yi) to the destination (shown as 'Ci' in the diagram and equation 3).The cipher block (W) will be the first block of the final ciphertext (Ci) sent, so that the decryption process can commence after receiving only two blocks. Encryption logic is shown above in Fig. 2.

Furthermore, the Parallel Encryption mode facilitates a message integrity check. The decryption process is actually a reverse of the encryption process. We split the first cipher block (W) from Ci and decrypt it using AES-1 and the decryption key IV ⊕K to obtain the hash value H(Xi). Then decrypt the rest of cipher blocks (Yi) using AES-1 (Shown as 'D' in equations 4 and 5) and the decryption key H(Xi) ⊕(IV ⊕K) to obtain a copy of the original file X.

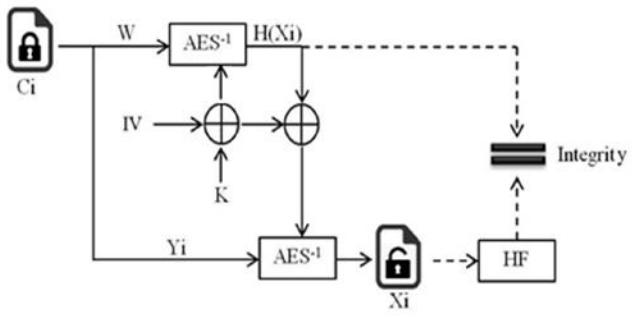


Fig.3. Decryption process.

$$H(X_i) = D_{(K \oplus IV)}(W) \quad (4)$$
$$X_i = D_{H(X_i) \oplus (K \oplus IV)}(Y_i) \quad (5)$$

In relation to integrity, this mode provides integrity without the need for a Message Authentication Code (MAC). We can check the integrity of the message by comparing the hash of the copy of the original file we derived from Ci and H(Xi).This extra step (integrity step) is optional. Decryption logic is shown above in Fig. 3.

Some may argue that CTR (Counter mode) plus CBC-MAC mode (CCM) can provide an integrity check in addition to performing encryption. However, the inherently sequential nature of CBC mode, and the need for two passes with AES, can make CCM-AES a slower scheme [18]. CCM mode requires a longer time than CBC mode (CBC is part of CCM). As a result, the Parallel Encryption mode would be better than CCM in terms of speed and provides message integrity in addition to encryption.

IV. EXPERIMENTAL RESULTS

The Parallel Encryption mode has been tested in computers using Windows 7 and an Intel(R) Core(TM) i7-2600 CPU. CrypTool 2.1 [19] used to compare the results between CBC mode and the Parallel Encryption mode using an AES algorithm. The results indicate that to encrypt a one gigabyte file using 128bits key input, CBC mode took 156889 ms and Parallel Encryption mode took 154362 ms. Noticeably, the gap between the CBC mode encryption time and Parallel Encryption mode encryption time would be increased with increasing file size, as shown in table I:

Table I: Comparison between Existing CBC mode and proposed mode.

	Modes of encryption	Size	Total ms
1	CBC mode	1 gigabyte	156889
	Parallel Encryption mode		154362
2	CBC mode	2 gigabyte	313778
	Parallel Encryption mode		308724
3	CBC mode	3 gigabyte	470666
	Parallel Encryption mode		463086
4	CBC mode	4 gigabyte	627555
	Parallel Encryption mode		617448
5	CBC mode	5 gigabyte	78444
	Parallel Encryption mode		771810

The time difference between CBC mode and the Parallel Encryption mode to encrypt one gigabyte of data is 2527ms.

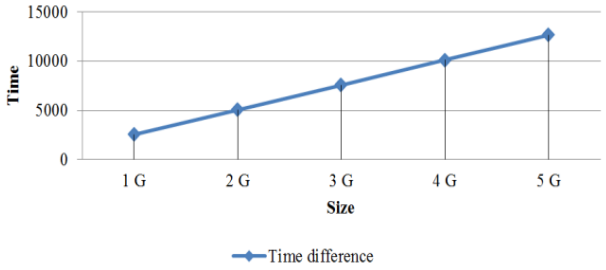


Fig .4. The time difference between the two modes of encryption.

Fig. 4 shows the time difference between the two modes of encryption for files up to five gigabytes, it is states that the time difference between the two modes increases linearly with file size.

V. CONCLUSION

A new Parallel Encryption mode with faster speed and an integrity check was described in this paper. MD5 hash function was used to produce the 128bit hash values. The hash value of the original file was used as part of the encryption and decryption keys generation process. This hash function ensured that any change in the plaintext blocks will affect all cipher blocks. Furthermore, the CrypTool 2.1 platform for cryptography and cryptanalysis was utilized to measure the encryption times of both CBC and Parallel Encryption mode. Then, we compare the encryption times and the result indicates that Parallel Encryption mode is faster than CBC. In addition, the file hash value in Parallel Encryption mode can be used as an integrity check. To sum up, Parallel Encryption mode provides a faster encryption process and integrity check as compared to CBC mode. In the future, we will use the proposed mode to develop new cloud cryptosystems and infrastructure to secure users' data and information.

REFERENCES

[1] M. Sugumaran, B. B. Murugan, and D. Kamalraj, "An Architecture for Data Security in Cloud Computing," in *Computing and Communication Technologies (WCCCT), 2014 World Congress on*, 2014, pp. 252-255.

[2] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," *Future Generation computer systems*, vol. 28, pp. 583-592, 2012.

[3] A. Oludele, E. C. Ogu, and U. C. Kuyoro'Shade, "On the Evolution of Virtualization and Cloud Computing: A Review," *Journal of Computer Sciences and Applications*, vol. 2, pp. 40-43, 2014.

[4] W. Stallings, "NIST Block Cipher Modes of Operation for Confidentiality," *Cryptologia*, vol. 34, pp. 163-175, 2010.

[5] C. Paar and J. Pelzl, *Understanding cryptography: a textbook for students and practitioners*: Springer Science & Business Media, 2009.

[6] A. K. Beeputh, M. R. Doomun, and P. Dookee, "Energy-Security Adaptation Scheme of Block Cipher Mode of Operations," in *Innovations and Advances in Computer Sciences and Engineering*, ed: Springer, 2010, pp. 73-78.

[7] H. Lipmaa, D. Wagner, and P. Rogaway, "Comments to NIST concerning AES modes of operation: CTR-mode encryption," 2000.

[8] M. A. Tarhuni, S. H. Ng, A. Samsudin, and W. P. Ng, "Enhanced counter mode," in *Communications, 2003.APCC 2003. The 9th Asia-Pacific Conference on*, 2003, pp. 701-705.

[9] D. Chen and H. Zhao, "Data security and privacy protection issues in cloud computing," in International Conference onComputer Science and Electronics Engineering (ICCSEE), 2012, pp. 647-651.

[10] X. Ma, "Security concerns in cloud computing," in *Proceedings of the 2012 fourth international conference on computational and information sciences (ICCIS 2012), Chongqing*, 2012, pp. 17-19.

[11] N. Gonzalez, C. Miers, F. Redígolo, M. Simplicio, T. Carvalho, M. Näslund, *et al.*, "A quantitative analysis of current security concerns and solutions for cloud computing," *Journal of Cloud Computing*, vol. 1, pp. 1-18, 2012.

[12] I. Hussain and I. Ashraf, "Security Issues in Cloud Computing-A Review," *Int. J. Advanced Networking and Applications*, vol. 6, pp. 2240-2243, 2014.

[13] V. Gampala, S. Inuganti, and S. Muppidi, "Data security in cloud computing with elliptic curve cryptography," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 2, pp. 138-141, 2012.

[14] R. M. Dansereau, S. Jin, and R. A. Goubran, "Reducing Packet Loss in CBC Secured VoIP using Interleaved Encryption," in *Electrical and Computer Engineering, 2006. CCECE'06. Canadian Conference on*, 2006, pp. 1320-1324.

[15] S. Fazackerley, S. M. McAvoy, and R. Lawrence, "GPU accelerated AES-CBC for database applications," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, 2012, pp. 873-878.

[16] M. Stamp, *Information security: principles and practice*: John Wiley & Sons, 2011.

[17] Z. Wang, J. Graham, N. Ajam, and H. Jiang, "Design and optimization of hybrid MD5-blowfish encryption on GPUs," in Proceedings of 2011 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA), Las Vegas, Nevada, USA, 2011.

[18] P. Rogaway, "Evaluation of some blockcipher modes of operation," *Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan*, 2011.

[19] Platform for cryptography and cryptanalysis (https://www.cryptool.org/en/cryptool2-en).

An Energy Efficient TCP DoS Attacks Mitigation Method in Cloud Computing

Aqeel Sahi[1, 2], David Lai[1], and Yan Li[1]

[1]University of Southern Queensland, Toowoomba, Australia
[2Thi]-Qar University, Thi-Qar, Iraq

Keywords: Cloud computing, TCP DoS attacks, energy consumption.

Cloud computing is a model which provides an easy, cheap, and flexible technological services. However, it poses some security problems. One of the most common security problems is the TCP DoS attack. This attack threatens any cloud in terms of energy consumption and resources exhaustion. In this paper, we propose a method to mitigate the DoS attacks in a cloud by reducing excessive energy consumption via limiting the number of packets. Instead of system shutdown, the proposed method ensures the availability of service. The proposed method can better manage the incoming packets by dropping packets from the most frequent requesting sources. This method shows that it can process 98.4% of the accepted packets during an attack. Furthermore, it is proved that dropping the most frequent requesting sources will always save more energy than not dropping when under attacks.

1. Introduction

The importance of the cloud computing has been significantly increasing within the last decade due to the ever increasing services and facilities that cloud can deliver. In addition, cloud computing services are in extremely high demand from organizations and individuals as a result of benefits the services can offer, such as better computing power, inexpensive operation costs, good performance, and high availability. On the other hand, cloud computing projects have some security issues that need to be addressed [1]. One of the most concerning issues is the Denial of Service (DoS) attacks [2]. There are many types of DoS and distributed DoS attacks. Many of them were mentioned in [3]. One of those types is the Transmission Control Protocol (TCP) DoS flood attack, which we will discuss in this paper. The TCP DoS attacks can flood the victim's network with extremely large numbers of packets to shut the network down and prevent legitimate users from accessing the services, to exhaust the resources and cause overheating, and to consume more energy than expected. Historically, in 2013 there was an incident of overheating Microsoft's data center which led to 16 hours' shutdown in Microsoft's cloud services, such as SkyDrive, Hotmail and Outlook [4]. Therefore, developing an efficient DoS mitigation method that can keep the services available 24/7 and reduce the excessive energy consumption is an important research topic. In this paper, we propose an energy efficient TCP DoS attack mitigation method that can be implemented in the cloud environment. The proposed method is designed to keep the incoming packets number under the number of packets the server can handle. Thus, this method can improve the services for cloud projects with protecting them against the packets flood, as well as reducing cost and energy consumption.

The rest of this paper is structured as follows: Section 2 briefly describes the proposed method. In Section 3, we evaluate the performance of the proposed method. In Section 4 we sum up this paper with conclusions.

2. The Proposed Method

In this section, we discuss cloud architecture as well as the mitigation method. Our cloud includes: (1) data storage, a single server point (*Server*); (2) several other points that represent the legitimate points (*Client*); (3) spurious points (*Attacker*); and (4) TCP packets ($TCP_{Packets}$). Assume that the *Attacker* uses a real, not faked or spoofed IP. We also assume that $T_{Packets}$ is the rate of $TCP_{Packets}$, which *Server* can handle within a time frame *Time*, and $L_{Packets}$ is the total rate of the legitimate $TCP_{Packets}$, that the *Client* points send to the server. The following equation is a further assumption that the rate of the legitimate packets is at most the half of the handling capacity of the *Server* to allow for the bursts of requests:

$$L_{Packets} = T_{Packets} / 2 \quad (1)$$

In addition, we assume that together with the legitimate packets, the *Attacker* is able to send spurious to make $TCP_{Packets}$ more than what the *Server* can handle. We denote the *Attacker* spurious packets as $S_{Packets}$, such that:

$$L_{Packets} + S_{Packets} > T_{Packets} \quad (2)$$

The proposed method aims to keep the sum of $L_{Packets}$ and $S_{Packets}$ smaller than $T_{Packets}$ to ensure the availability of a cloud project, reduce energy consumption, and decrease the cost of the cloud maintenance:

$$L_{Packets} + S_{Packets} \leq T_{Packets} \quad (3)$$

In case of a normal traffic when no attack is occurred to the cloud,

$$S_{Packets} = 0 \quad (4)$$

From equation (2), $L_{Packets} = T_{Packets} / 2$, equation (3) is satisfied for the normal traffic scenario.

On the other hand, in the case of abnormal traffic when the attack is performed on the cloud, the proposed method ∏ will drop the $TCP_{Packets}$ of the most frequent requesting IP addresses in order to ensure that $L_{Packets} + S_{Packets}$ does not exceed $T_{Packets}$. Most frequent requesting IP addresses can be filtered using any packets filtering method, such as iptables [5].

For example, suppose that our server can handle 10 packets per *Time*, and there are six sources of {*attacker* | *Client*} initiating a connection with the *Server*. The number of packets that are sent by the sources are as follows: $source_1$=1 packet, $source_2$=2 packets, $source_3$=3 packets, $source_4$=4 packets, $source_5$=5 packets, and $source_6$=6 packets. Therefore, the number of packets that are sent from sources 1 to 6 are about 21 packets, while the server can handle a maximum of 10 packets within *Time*. In this case, ∏ will use a filter to determine which sources are the most frequent requesting IP addresses, and then ∏ will drop them. In our example $source_5$ and $source_6$ have 5+6=11 packets, and the rest of the packets are 1+2+3+4=10 packets. Intuitively, $source_5$ and $source_6$ that have relatively higher request rates are more likely the target of the TCP DoS attacker sources than the other less frequent requesting sources. Even though there is a possibility that $source_5$ and $source_6$ could be *Client* sources, ∏ can keep the request rate below $T_{Packets}$, and the service is still available. Furthermore, using ∏ will save energy during a DoS attack as shown in the performance evaluation section below.

3. Performance Evaluation

We tested the proposed method in our lab. The network contained one *Server*, one *Attacker*, and eight legitimate *Clients*.

We established TCP DoS attacks from the Attacker (IP_2) point against the Server (IP_1) point alongside with the legitimate TCP connection requests from *Clients* (IP_3 to P_{10}). According to Tables 1 and 2, the *Attacker* sent 47807 requests and received only 21130 responses, and the *Server* sent 47893 requests and made 21219 responses.

Wireshark Network Analyser 2.0.0 [6] was used to capture the packets traversing the experimental network and no missing packets were discovered, as shown in Table I. 99.7% of the traffic with 68997 packets were using in TCP protocol. In Tables 1 and 2, sources indicate the source of a packet and destinations present the destination of a packet.

		Sources	Destinations
N	Valid Packets	69235	69235
n	TCP Packets	68997	68997
	Missing	0	0

Table 1. Overall packets statistics

	Sources		Destinations	
	Rate number of packets	Percent	Rate number of packets	Percent
(IP_1) *Server*	21219	30.6	47893	69.2
(IP_2) *Attacker*	47807	69.1	21130	30.5
(IP_3) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_4) *Client*	< 60	< 0.08	< 60	< 0.08
(IP_5) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_6) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_7) *Client*	< 60	< 0.08	< 60	< 0.08
(IP_8) *Client*	< 30	< 0.04	< 30	< 0.04
(IP_9) *Client*	< 60	< 0.08	< 60	< 0.08
(IP_{10}) *Client*	< 30	< 0.04	< 30	< 0.04
Total	69235	100.0	69235	100.0

Table 2. Sources and destination statistics

Let the probability of an accepted $TCP_{Packets}$ within the time frame of *Time* be ε_1. The number of all the incoming packets during *Time* is denoted as *n*. The number of $TCP_{Packets}$ which are expected to be sent to the *Server* during *Time* must not exceed $T_{Packets} \cdot Time$. Thus, every single $TCP_{Packets}$ has the following probability to be accepted by the *Server*:

$$\varepsilon_1 . n \leq T_{Packets} . Time \quad (5)$$

$$\therefore \varepsilon_1 \leq \frac{T_{Packets} . Time}{n} \quad (6)$$

In case of normal traffic when no attack is performed against the cloud, all $TCP_{Packets}$ must be accepted by the *Server*. As a result, the *Server* accept the $TCP_{Packets}$ with the following probability:

$$\varepsilon_1 = \begin{cases} 1, & if\ n \leq T_{Packets}.Time \\ \frac{T_{Packets}.Time}{n}, & if\ n > T_{Packets}.Time \end{cases} \quad (7)$$

As a result, this can be expressed as follows:

$$n = 68997, \quad T_{Packets} = 21219, \quad and\ Time = 3.2$$

$$\because n > T_{Packets}.Time$$

$$\therefore \varepsilon_1 = \frac{T_{Packets}.Time}{n}, \text{ from (7)}$$

$$\varepsilon_1 = \frac{21219 \cdot 3.2}{68997} \simeq 0.984 \ \blacksquare$$

Thus, we can claim that the proposed method ∏ can process 98% of the accepted packets during an attack.

As mentioned above, that the $TCP_{Packets}$ of the most frequent requesting IP addresses will be dropped in order to ensure that $L_{Packets}$ + $S_{Packets}$ won't exceed $T_{Packets}$. However, this means that $L_{Packets}$ which represent legitimate users packets could be dropped as well, and not only $S_{Packets}$ has the potential to be dropped. Therefore, we must justify ∏ in terms of the energy consumption, and investigate whether the benefits of ∏ are sufficient enough when it is compared with the loss of $L_{Packets}$. If ∏ drops $L_{Packets}$ from the traffic, it costs energy of receiving (R_{energy}) these packets by the *Server*, denoted by $\prod cost$. Whereas, if ∏ drops $S_{Packets}$ from the traffic, it saves energy by not processing the packets (P_{energy}), as well as saving the energy S_{energy} by not sending packets from a *Server* to *Client*, denoted by $\prod benefit$. According to the authors of [7, 8], S_{energy} is more than R_{energy}. Consequently, we can formulate the relation between R_{energy} and P_{energy} + S_{energy} as follows:

$$P_{energy} + S_{energy} > R_{energy} \quad (8)$$

$$\prod_{cost} = n.(1-\varepsilon_1).\frac{L_{Packets}}{L_{Packets}+S_{Packets}}.(R_{energy}) \quad (9)$$

$$\prod_{benefit} = n.(1-\varepsilon_1).\frac{S_{Packets}}{L_{Packets}+S_{Packets}}.(P_{energy} + S_{energy}) \quad (10)$$

$$\prod_{profit} = \prod_{benefit} - \prod_{cost} \quad (11)$$

where $\prod profit$ is the net energy saved when ∏ is in action.

In case of no attacks $n \leq T_{Packets}.Time$, from (7) the probability of packets acceptance will be $\varepsilon 1 = 1$, therefore the profit of ∏ will be calculated as follows:

$$\prod_{cost} = n.(1-1).\frac{L_{Packets}}{L_{Packets} + S_{Packets}}.(R_{energy}) = 0$$

$$\prod_{benefit} = n.(1-1).\frac{S_{Packets}}{L_{Packets} + S_{Packets}}.(P_{energy} + S_{energy}) = 0$$

$$\therefore \prod_{profit} = \prod_{benefit} - \prod_{cost} = 0, \text{ from (11)}$$

On the other hand, in case of attacks $n > T_{Packets} \cdot Time$, from (7) the probability of packets acceptance will be $\varepsilon_1 = \frac{T_{Packets} \cdot Time}{n}$, therefore the profit of $\prod$ will be calculated as follows:

$$\prod_{cost} = n \cdot \left(1 - \frac{T_{Packets} \cdot Time}{n}\right) \cdot \frac{L_{Packets}}{L_{Packets} + S_{Packets}} \cdot (R_{energy}) \quad (12)$$

$$\therefore \prod_{benefit} = n \cdot \left(1 - \frac{T_{Packets} \cdot Time}{n}\right) \cdot \frac{S_{Packets}}{L_{Packets} + S_{Packets}} \cdot (P_{energy} + S_{energy})$$

$$\because P_{energy} + S_{energy} > R_{energy}, \text{ from (8)}$$

$$\therefore \prod_{benefit} = n \cdot \left(1 - \frac{T_{Packets} \cdot Time}{n}\right) \cdot \frac{S_{Packets}}{L_{Packets} + S_{Packets}} \cdot (R_{energy}) \cdot Z \quad (13)$$

$$where\ Z > 1$$

$$\therefore \prod_{profit} = \frac{n \cdot (S_{energy} + R_{energy})}{L_{Packets} + S_{Packets}} \cdot \left(1 - \frac{T_{Packets} \cdot Time}{n}\right) \cdot (Z - 1) \cdot R_{energy} \quad (14)$$

$$\because Z > 1$$

$$\therefore \prod_{profit} > 0 \ \blacksquare$$

As a result, dropping $TCP_{Packets}$ of the most frequent requesting IP addresses when under attacks will always save more energy than not dropping.

To recap, this paper handles the security problem in cloud computing and proposes a method to mitigate TCP DoS attacks by reducing excessive energy consumption via limiting the number of packets. Instead of system shutdown, the proposed method ensures the availability of service.

4. Conclusion

To sum up, in this paper we presented an energy efficient TCP based DoS attacks' mitigation method to enhance the security of the cloud on top of saving energy. The method maintains the availability of the service by controlling the number of the server processed TCP packets to stay below the number of packets which the server can handle. This method can reduce the energy consumption in case of a DoS attack thereby reducing the chance of shutting down services due to such attacks.

References

[1] A. Sahi, D. Lai, and Y. Li, "Security and privacy preserving approaches in the eHealth clouds with disaster recovery plan," Computers in biology and medicine, vol. 78, pp. 1-8, 2016.

[2] S. M. T. Nezhad, M. Nazari, and E. A. Gharavol, "A Novel DoS and DDoS Attacks Detection Algorithm Using ARIMA Time Series Model and Chaotic System in Computer Networks," IEEE Communications Letters, vol. 20, pp. 700-703, 2016.

[3] A. Shameli-Sendi, M. Pourzandi, M. Fekih-Ahmed, and M. Cheriet, "Taxonomy of Distributed Denial of Service mitigation approaches for cloud computing," Journal of Network and Computer Applications, vol. 58, pp. 165-179, 2015.

[4] Z. Anwar and A. W. Malik, "Can a DDoS attack meltdown my data center? A simulation study and defense strategies," IEEE Communications Letters, vol. 18, pp. 1175-1178, 2014.

[5] C. Diekmann, J. Michaelis, M. Haslbeck, and G. Carle, "Verified iptables Firewall Analysis," 2016.

[6] Wireshark Network Analyser, https://www.wireshark.org/

[7] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2001, pp. 1548-1557.

[8] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," Wireless networks, vol. 8, pp. 481-494, 2002.

Corresponding author's email: akeel_sahy@yahoo.co.uk

End of thesis