A MUTUAL KEY AGREEMENT PROTOCOL TO MITIGATE REPLAYING ATTACK IN EXPRESSIVE INTERNET ARCHITECTURE (XIA)

Beny Nugraha[1], Rahamatullah Khondoker[2], Ronald Marx[2], Kpatcha Bayarou[2]

[1]Department of Electrical Engineering, Mercu Buana University, Jakarta, Indonesia
[2]Fraunhofer SIT, Rheinstr. 75, Darmstadt, Germany

ABSTRACT

Several Future Internet (FI) architectures have been proposed to address the problems of the Internet including flexibility (so called IP bottleneck), host-based addressing (addressing a host rather than the content itself), and security. In this paper, we focus on eXpressive Internet Architecture (XIA) as it is the most secure and open-source Content-Centric Network (CCN). CCN is claimed by the Future Content Networks (FCN) Group to be the Future Internet (FI). However, XIA does not have any mechanisms to mitigate the replaying attack, thus, this paper proposes and implements a solution to mitigate it. Several existing solutions have been analyzed to derive the requirements for the proposed solution. By implementing the proposed protocol, XIA is now able to mitigate all of the reviewed network attacks. The evaluation shows that the proposed solution is more secure and less complex over the existing solutions.

Keywords— Replaying Attack, Session Key, eXpressive Internet Architecture (XIA), Future Internet (FI), CCN

1. INTRODUCTION

Current Internet faces challenges such as inability to provide flexibility - changing of protocol in one layer requires another changing of protocol in another layer, and inability to provide intrinsic security - a security mechanism is added to counter a new threat, it is not integrated. The problems arise mainly because of the design principles of the Internet that are hard to be changed (cannot provide flexibility) [1]. Several Future Internet (FI) architectures have been developed to solve these problems. However, a security analysis for FI architectures is necessary to ensure that they have fulfilled the security goals.

A security analysis for seven FI architectures, namely, eXpressive Internet Architecture (XIA), Recursive Inter-Network Architecture (RINA), Service Oriented Network Architecture (SONATE), Netlet-based Node Architecture (NENA), MobilityFirst, NEBULA, and Named Data Networking (NDN) has been done in [2]. The conclusion of the security analysis is, all of them cannot tackle all of the reviewed attacks. The conclusion of the security analysis can be seen in Figure 1.

It can be seen from Figure 1 that no architecture is robust against all of the reviewed attacks. Each architecture must have at least one security mechanism to prevent an attack. For example, to prevent the snooping attack, the architecture must have an encryption mechanism. However, there are security mechanisms that can handle several attacks, for example the digital signature. It can be used to mitigate man-in-the-middle, and reflection attacks.

Security Goals	Security Attacks	SONATE	NENA	XIA	RINA	MobilityFirst	NDN	NEBULA
Confidentiality	Snooping	√	√	√	√	√	√	X
	Traffic Analysis	X	X	√	√	X	X	X
Integrity	Modification	√	√	√	√	√	√	√
	Repudiation	X	X	√	X	√	√	√
Availability	Denial of Service	√	√	√	X	√	X	√
Authentication	Man-In-The-Middle	√	√	√	√	√	√	√
	Reflection	√	√	√	√	√	√	√
	Masquerading	X	X	√	√	√	√	X
	Replaying	√	√	X	√	X	√	√

Legend:
√: Has (by design) mitigation mechanism(s)
X: Has no (by design) mitigation mechanism(s)

Figure 1. Comparison Of Several FI Architectures In Terms Of Handling Attacks

Criteria	Future Internet Architectures						
	XIA	RINA	SONATE	NENA	MobilityFirst	NEBULA	NDN
Approach	Content-Centric	Content - Centric	Protocol Graph	Protocol Graph	Content-Centric	Supports Cloud Computing	Content - Centric
Project Started	In 2010	In 2010	In 2009	In 2009	In 2010	In 2010	In 2010
Demo	√	√	√	√	√	√	X
Prototype	√	√	√	√	√(Not released in public yet)	X	√
Amount of Mitigated Attacks (Out of 9)	8	7	6	6	7	6	7

Legend:
√: Available
X: Not Available

Figure 2. Criteria For Choosing The Most Potential FI Architecture

In this paper, we select the most potential FI Architecture in term of several criteria such as the maturity of the projects, the availability of a demonstration and prototype, and the number of attacks that could be mitigated (by using existing security mechanisms). The criteria for choosing the potential architecture can be seen in Figure 2. We choose to focus more on XIA because of the following reasons:

- XIA is Content-Centric Network (CCN). CCN is claimed by the Future Content Networks (FCN) Group

as the Future Internet (FI). [3]

- XIA is the most secure one. XIA is able to mitigate 8 out of 9 attacks.
- Demonstration is already presented, so that means XIA has been successful to show the proof-of-concept.
- Prototype for XIA is available, and it is open source on github [4]. That means, XIA can be tested in the industrial context.

XIA is discussed in more details in the following subsection.

1.1. Background: eXpressive Internet Architecture (XIA)

XIA is an FI Architecture that uses content-centric approach. Content-centric means if a user wants to retrieve a content in XIA network, he only needs to know the ID of the content not the host address.

In XIA, there are multiple principles to retrieve the data: Content ID (CID), Host ID (HID), and Service ID (SID). CID is used to retrieve the content and to ensure that the content is correct because CID is a hash of the content (e.g., using SHA 256). HID is used to ensure that the content is coming from the right host because HID is the hash of the public key of the Host. SID is used to ensure that the content is provided by the right service because SID is the hash of the public key of the service [5].

There are four security mechanisms in XIA:

First, XIA does not use well-known ports, it uses Content/Hash/Service ID (CID/HID/XID) to retrieve the content. These IDs are used by the receiver to ensure that the requested content/host/service is correct.

Second, Lightweight Anonymity and Privacy (LAP), which enables anonymous communication to prevent remote tracking [6].

Third, Sanctuary Trail-Refuge from Internet DDoS Entrapment (STRIDE) defense mechanism. STRIDE is provided by SCION and it is robust against the DoS attack. It allocates the available bandwidth in a tree-based topology so that the bandwidth can be split from the TD Core to the end-host AD [7].

Fourth, Accountable Key Infrastructure (AKI) is a public-key validation infrastructure used for authentication [8].

However, all of the above security mechanisms cannot prevent the replaying attack [2], thus, the replaying attack solution for XIA is required.

The description of replaying attack is given in the next subsection.

1.2. Background: Replaying Attack

Replaying attack inhibits the authentication goal (i.e., only the authorized user is able to send a message and the receiver is able to proof the sender's identity). This attack occurs when a packet is captured and then replayed later in a different session in order, for example, to gain others trust. For example, suppose Alice wants to prove her identity to Bob. Bob requests Alice's password as a proof of identity, meanwhile, Trudy is watching on the conversation and captures the password that was sent by Alice. Trudy waits until Alice and Bob open a new session before she connects to Bob. When asked for a proof of identity, Trudy sends the password of Alice that she captured from the last session, which Bob accepts [9]. Replaying attacks can also happen when an adversary first intercepts some communication data in the current key agreement protocol run, and then the adversary replays the intercepted data with the receiver in a future key agreement protocol run [10]. The replaying attack can be mitigated by having a marker in each session (e.g., a random nonce, a session key, a timestamp). This marker is used to distinguish between the messages in different sessions.

In this paper, we provide a replaying attack solution which is more secure and less complex than the existing solutions. The rest of this paper is organized as follows: In Section 2, the existing solutions for replaying attack are analyzed to derive the requirements for the proposed solution which is described in Section 3. Based on the derived requirements, the solution is proposed in Section 4. The implementation and evalution of the proposed solution are described in Section 5 and 6 respectively. Finally, the conclusion and the future work are discussed in Section 7.

2. ANALYSIS OF THE EXISTING SOLUTIONS FOR REPLAYING ATTACK

In order to propose a solution to be implemented in XIA, we reviewed nine existing solutions that have the mechanisms to prevent the replaying attack (e.g., session keys, random nonce, or timestamp). They are Diffie-Hellmann [11], Lamport's Password Authentication [12], S/Key One Time Password [13], Keung-Siu Protocol [14], Message Binding [15], Timestamp [16], Luo-Shieh-Shien Authentication Protocol [17], Yoon-Jeon Protocol [10], and Tseng-Jou Protocol [18]. These solutions are analyzed to derive the requirements for the proposed solution.

2.1. Diffie-Hellmann

Diffie-Hellman is a method to compute a unique session key. In order to compute a session key, the sender and the receiver choose two public parameters and generate a new private value in every session.

Diffie-Hellman was developed by Whitfield Diffie and Martin E. Hellman and it was published in 1976 [11].

Advantage: This method is considered to be secure if the value of the public parameters, p and g, are chosen properly. Therefore, it is not likely for an attacker to calculate the secret key $s = g^{ab} \bmod p$. The secret key can be used to prevent replaying attack because only with the correct secret key Alice and Bob can encrypt and decrypt their messages [11].

Disadvantage: Original Diffie-Hellman scheme does not authenticate the communicating users, thus, it is vulnerable to the man-in-the-middle attack [18]. A person in the middle may establish two distinct Diffie-Hellman key exchanges, one with Alice and the other with Bob, effectively masquerading as Alice to Bob, and vice versa, allowing the attacker to decrypt (and read or store) then re-encrypts the messages passed between them.

2.2. Lamport's Password Authentication

Lamport's password authentication is a secure one-time password authentication method that was published by Leslie Lamport in 1981 [12] [19].

This method implements a one-time password to protect against eavesdropping. The authentication process is between the user (A) and the server (S).

Advantage: This method is robust against the replaying attack since one session is bound by one password. Furthermore, a system that uses this method will never used a same password even though the system is crash. The system does not require back up to a point where a password already have been used, the system will continue from the point when the system crashed.

Disadvantage: This method is vulnerable to one type of man-in-the-middle attack, called the small *n* attack (e.g., the attacker impersonates the server).

2.3. S/Key One Time Password

S/KEY One Time Password is a method that only allows one password ever crosses the network. The secret of a user will never be shared, thus, it prevents from an eavesdropping. This method was published by Neil Haller in 1994 [13].

Advantage: The user's secret pass-phrase never crosses the network at any time, thus, this method is able to prevent an eavesdropper. Assuming that an attacker manages to get hold of a passwords that was used for a successful authentication. This password is already useless for subsequent authentications, because each password can only be used once in one session, thus, prevents replaying attack.

Disadvantage: This method is vulnerable to a dictionary attack where the attacker is using a list of most possibly used passwords to guess the secret [20].

2.4. Keung-Siu Protocol

This protocol was developed by Stephen Keung and Kai-Yeung Siu in 1995 [14]. Aims of this protocol are to establish a session key while protecting the weak passwords (easy to be guessed by using a list of commonly used passwords) and to prevent off-line password guessing attack (the attacker guesses a password by analyzing the pattern of legitimate user's password). This protocol provides authentication process by using challenge and response messages that allow both users to validate each other.

Advantage: The protocols are immune to the replaying attack because of the following properties:

1. It uses random number that is different in every session. This random number is used to ensure that both hosts are communicating in one session.

2. This method also uses session key that is different in every session. The session key can be used to prevent the replaying attack.

3. This method uses encryption mechanisms so that the attacker is unable to read the message.

Disadvantage: The source and the destination must know the public key server, meanwhile, there is a situation where the public key is difficult to obtain (e.g., in a mobile environment).

2.5. Message Binding

By binding the messages to their correct context (e.g., binding the message to its protocol run), the replaying attack can be prevented. One way of binding the message can be done by including an information in the messages, therefore they are recognized to belong to a certain state of a certain protocol run. Example of information that can be included in the message is a protocol identifier [15].

Advantage: Message binding is able to withstand replaying attack because it has an information that is tagged to the message to bind the message and the protocol run.

Disadvantage: Message binding cannot bind a message and a session, it only binds a message with a protocol run. That means, in a certain point the replaying attack cannot be prevented (e.g., where the same protocol is used in a different session).

2.6. Timestamp

Timestamp is a marker that is used in a message to ensure the freshness of the message [16].

Advantage: The replaying attack is prevented by the use of timestamps. For example, a developer sets the value of δ_{max}, a constant to limit the difference in timestamp, to 200 milliseconds. If the receiver gets the message and the value of $|T_{sender} - T_{receiver}|$ is higher than 200 milliseconds, then the receiver will detect the replaying attack and drops the message [16].

Disadvantage:

One disadvantage of timestamp is in term of clock synchronization of the two hosts. Synchronization is required to maintain the accuracy and precision of the timestamp. The other disadvantage is, maintaining a list of used timestamps within the current window has the drawback of potentially large storage requirement, and corresponding verification overhead [10].

2.7. Luo-Shieh-Shien Authentication Protocol

This is a protocol to generate session keys with the help of a third party authentication server. This protocol was developed by Jia-Ning Luo, Shiuhpyng Shieh, and Ji-Chiang Shen and was published in 2006 [17].

Advantage: This protocol uses random numbers and session keys. The replaying attack can be mitigated by using the session keys as marker to distinguish the messages in different sessions. Furthermore, there is a mechanism to ensure that both hosts has created the same session key.

Disadvantage: There is a redundant message that increases the complexity of the protocol.

2.8. Yoon-Jeon Protocol

This protocol was developed by Eun-Jun Yoon and Il-Soo Jeon and was published in 2010 [10]. The protocol generates session keys based on Chebyshev polynomial.

Advantage: This protocol is robust against the replaying attack by utilizing the session key to ensure that the messages in one session are different than the messages in another session. Additionally, secure mutual authentication between entities is achieved by using a MAC by each entity. MAC is created by hashing the identity of both users and the Chebyshev Polynomial that is received by each user.

Disadvantage: There is an unused random number *N*. User A selects large prime number *N* that is not used in any operation and it is also not used to detect the freshness of the message. The inclusion of an unused random number can increase the complexity of the protocol.

2.9. Tseng-Jou Protocol

This protocol is an improvement of Yoon-Jeon Protocol. Similar to Yoon-Jeon Protocol, Tseng-Jou Protocol uses Chebyshev polynomial as a base to generate session keys. The main improvement is, it provides anonymous identity of the hosts by generating a parameter pseudo identity (PID) on each host. This protocol was developed by Huei-Ru Tseng and Emery Jou and published in 2011 [18].

Advantage: This protocol can mitigate replaying attack by using the session key to ensure the messages is bound to a specific session. It also provides anonymous identity of the host by having parameter PID on each host.

Disadvantage: There is an unused random number N_i. User U_i selects a large prime number N_i that is not used in any operation. To decrease the complexity of the protocol, the used of an unused random number can be avoided.

3. DERIVED REQUIREMENTS FOR THE PROPOSED PROTOCOL

It can be seen in the last Section that all of the reviewed existing solutions have their own problems.

The properties that need to be satisfied by the proposed protocol are:

1. Use of a marker to distinguish the messages in different sessions.
2. Having a process to ensure that both users generate the same session key.
3. Using an encryption mechanism to protect the message, therefore, it is unreadable by the attacker.
4. Utilizing a mechanism to conceal the identity or the address of the sender.

Meanwhile, the properties that need to be avoided by the proposed protocol are:

1. Even though timestamp can be used as a marker, it has a disadvantage in term of clock synchronization between two communicating users. Therefore, timestamp can be avoided to reduce the risk of having synchronization issue.
2. Redundant computation that reduces the efficiency of the protocol.
3. The use of a useless random number that increases the complexity of the protocol.
4. To use several encryption mechanisms that reduces the efficiency of the protocol.

4. THE PROPOSED PROTOCOL

The proposed solution has to satisfy the desired properties and avoid the unwanted ones. Thus, the proposed solution is a complete protocol that provides a mechanism to mitigate replaying attack, provides an encryption mechanism, enables anonymous connection, and provides mutual authentication process. The protocol has the following properties:

1. Has markers in each session in the form of session keys (each host has one session key with length up to 280 bits).
2. The session keys are generated by XOR computation of four random numbers (70 hex per random number). The session keys are used by both users to differentiate the messages in different sessions.
3. Has a mechanism to ensure that the random numbers that are received at the receiver side are correct. This mechanism is needed for both hosts to create the same session key. This is achieved by checking the MAC in each host. The MAC value that is sent by User B has the random numbers that is generated by User A and has been received by User B. If User A finds the difference in the MAC value (e.g., someone is altering the random numbers, or there is an error in the network so that User B cannot obtain the random numbers from User A), then User A will terminate the session.

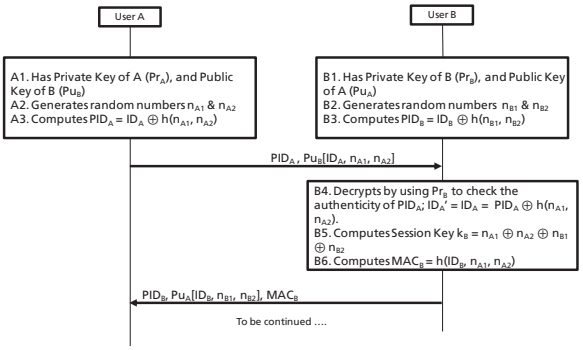

Figure 3. Sequence Diagram of The Proposed Protocol-Top Part

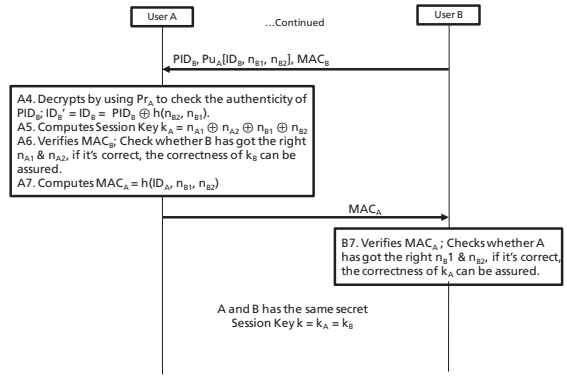

Figure 4. Sequence Diagram of The Proposed Protocol-Bottom Part

4. It also has a mechanism to ensure that both users generate a correct session key. This mechanism is needed to detect the replaying attack. This is also achieved by checking the MAC in each host and if each user has verified the MAC, then both users has generated a same session key.

5. Has two times data encryption, therefore, an attacker cannot read the message.

6. It does not have redundant computation and useless random number, thus reduces the complexity and increases the efficiency of the protocol.

7. It generates a parameter that is called Pseudo Identity (PID) to hide the host's identity.

The sequence diagrams of how the protocol works can be seen in Figures 3 and 4.

The proposed protocol works in the following ways:

1. The first assumption before running the protocol is as follows: User A and user B have exchanged their public key to be used for the encryption-decryption mechanisms.

2. User A generates two random numbers n_{A1} and n_{A2}, hashes these two random numbers, then computes pseudo identity (PID_A) to hide his identity. He encrypts his identity (HID_A) and his random numbers with the user B's public key, then sends it along with the PID_A to user B.

3. User B generates two random numbers n_{B1} and n_{B2}, hashes these two random numbers, then computes pseudo identity (PID_B) to conceal his identity. He encrypts his identity (HID_B) and his random numbers with user A's public key. After receiving the message from A, he decrypts the message using his private key, then he authenticates the identity of A, if it is not correct then he will terminate the connection. But if it is correct, he will compute session key k_B and MAC_B. Then he sends his PID_B along with the encrypted message (HID_B, n_{B1}, n_{B2}) and the MAC_B.

4. After user A receiving the message from user B, he decrypts the message by using his private key. Then he authenticates the identity of B, if it is not correct then he will terminate the connection. But if it is correct, he will compute session key k_A and MAC_A. After that, user A will authenticate the MAC_B to make sure that user B got the correct random numbers from him, this means B also has generated a correct session key. After completing all of the checking processes, user A sends his MAC_A to B.

5. User B will authenticate MAC_A to make sure that user A has got the correct random numbers from him, and also to make sure that user A has generated a correct session key.

6. After completing all of the checking processes, user A and user B have the same secret session key ($k_A = k_B$) to be used during their communication.

7. The random numbers and the session keys that are generated by user A and user B are different in every session.

5. IMPLEMENTATION

The protocol is implemented in XIA Prototype in order to prove that the protocol is able to make XIA robust against replaying attack and is able to generate the desired result (secured session keys). In order to simulate how the proposed protocol prevents the replaying attack, a topology is created by using VirtualBox version 4.2.12. The topology can be seen in Figure 5.

It can be seen in Figure 5 that the Attacker is connected to the Router via Ethernet 1 and to the NAT via Ethernet 2. In XIA, The Attacker cannot connect to Host0 and Host1 directly. It is necessary for the Attacker to connect with the Router. Since the HID of the Attacker is given by the Router. The Router is the one that connects the Attacker with Host0 and Host1. Also can be seen in in Figure 5 that each host

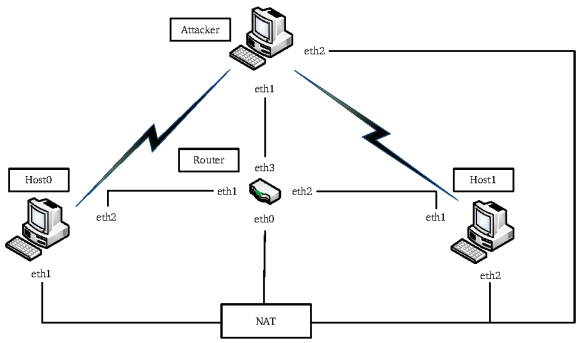

Figure 5. Topology for Evaluation

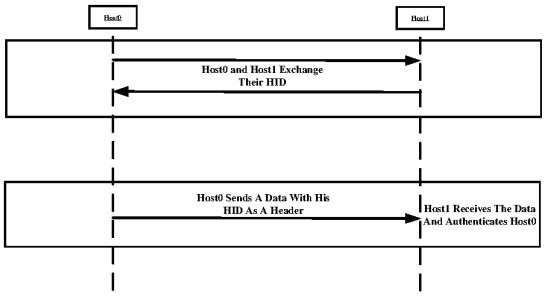

Figure 6. Sequence Diagram For Normal Scenario

connected via two interfaces, one of them is connected to the Router while the other is connected to NAT. Every hosts need to be connected to NAT in order to give them an internet connection that is used to obtain the XIA Prototype from the github [4].

There are three cases to be used to test the proposed protocol: First, a case when Host0 and Host1 are sending and receiving data without being interrupted by the Attacker. In this case, Host0 and Host1 do not run the proposed protocol applications. Second, when the Attacker is successfully performing the replaying attack. In this case, Host1 authenticates the Attacker as Host0. Third, a case when Host0 and Host1 run the applications for the proposed protocol before they start exchanging data. This case is used as a proof that the protocol is able to mitigate the replaying attack.

5.1. Common Data Exchange

In this case, Host0 and Host1 are exchanging data without using the proposed protocol. Instead of using the session key as a header of a data, they use their HID. To use the HID as a header, they are exchanging their HID before they start exchanging data. The sequence diagram of this step can be seen in Figure 6. It can be seen that, Host0 and Host1 communicate in one session only. It is assumed that the Attacker is idle.

The result from this case is, each host uses its HID as a header of the message that it wants to send. The HID is used by the receiver to authenticate the sender.

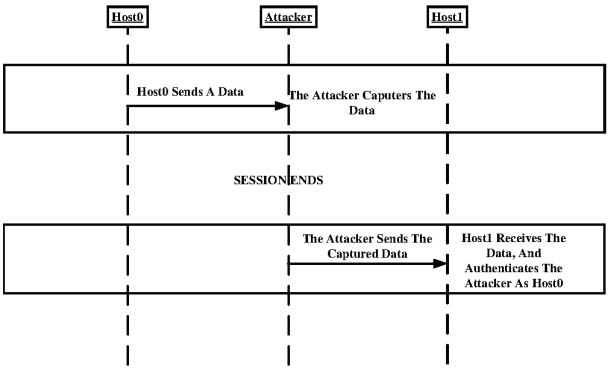

Figure 7. Sequence Diagram For Replaying Attack Scenario

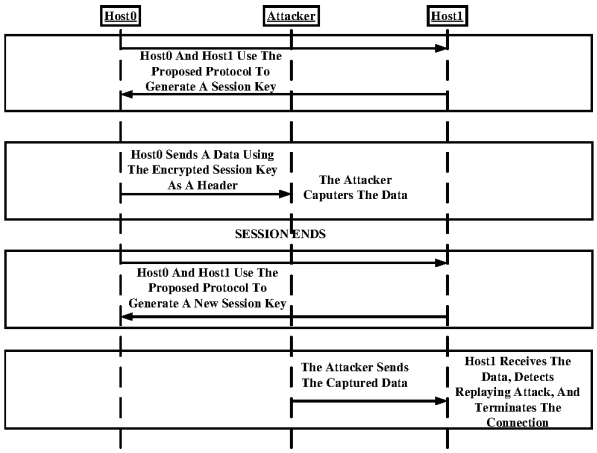

Figure 8. Sequence Diagram For Mitigating Replaying Attack Scenario

5.2. Replaying Attack Scenario

This case is to simulate the replaying attack. It is assumed that Host0 and Host1 have already exchanged HID. These HIDs are always same in each session. The scenario is, the Attacker captured and saved the data from Host0. To simulate the replaying attack, it is assumed that the previous session has ended and the Attacker replays the data from Host0 to Host1 in the next session. The sequence diagram of this step can be seen in Figure 7.

The result from this case is, the Attacker managed to perform the replaying attack. The Attacker replayed the data from Host0 to Host1 and Host1 authenticated the Attacker as Host0. The data is replayed in another session.

5.3. Mitigating Replaying Attack By Applying The Proposed Protocol

This case is to simulate how the proposed protocol mitigates the replaying attack. Host0 and Host1 create a session key by running the protocol. This protocol will be run in each session to create a session key that is unique in every session.

	Year Founded	Number of Messages	Amount of Random Numbers	Number of Data Encryption	Number of Data Decryption	Key Length
Diffie-Hellman	1976	4	2	0	0	
Lamport's Password	1981	5	0	0	0	
S/KEY One-Time Password	1994	2	0	0	0	64 bits
Keung-Siu Protocol	1995	4 (Client-Server), 7 (Peer-to-peer)	4	4 (Client-Server), 8 (Peer-to-peer)	4 (Client-Server), 8 (Peer-to-peer)	
Message Binding	1997	0	0	0	0	
Timestamp	1991	0	0	0	0	
Luo-Shieh-Shen Protocol	2006	5	4	8	8	
Yoon-Jeon Protocol	2010	4	2	2	2	
Tseng-Jou Protocol	2011	5	6	4	4	
Proposed Protocol	2013	3	4	2	2	280 bits

Figure 9. Comparison of the Proposed Protocol With Existing Solutions

The sequence diagram of this step can be seen in Figure 8.

The result from this case is, Host1 detects a replaying attack because the session key that is used by the Attacker is different than the session key that were generated by Host0 and Host1. This is because the session keys are different in every session. Once the attack is detected, Host1 terminates the session, and generates a new session key with Host0.

6. EVALUATION

This Section presents the evaluation of the proposed protocol by comparing it with the existing solutions as shown in Figure 9.

It can be seen in Figure 9 that:

1. The proposed protocol generates a session key with a length of 280 bits. The session key is never exchanged between hosts, therefore, the Attacker needs to guess the session key if he wants to carry out an attack (e.g., replaying or modification attacks). The possibility for the attacker to guess the session key is 2^{280}.

2. It has three messages to be exchanged while running the protocol. This amount of message is smaller than the other solutions that have four (Diffie-Hellman, Keung-Siu Protocol, and Yoon-Jeon Protocol) or five messages (Lamport's Password Authentication, Luo-Shieh-Shen Authentication Protocol, and Tseng-Jou Protocol).

3. It has four random numbers. The random numbers are used to generate the session key, a possibility for an attacker to guess four random number is $10^{70} \times 4$ (given 10 possibilities in one digit) and it is larger than a possibility to guess two random numbers (used in Diffie-Hellman and Yoon-Jeon Protocol), which is $10^{70} \times 2$. Furthermore, it is less complex than the solution that have six random numbers (Tseng-Jou Protocol). In addition, none of these random numbers are useless like in Yoon-Jeon and Tseng-Jou Protocols.

4. It has two times data encryption and decryption, therefore, it reduces the complexity than the other solutions that have four (Tseng-Jou Protocol and Keung-Seu Protocol between client and server) or even eight times data encryption-decryption (Keung-Seu Protocol between two clients and Luo-Shieh-Shen Authentication Protocol).

7. CONCLUSION AND FUTURE WORK

The eXpressive Internet Architecture (XIA) is a open-source Content-Centric Network (CCN) which has potential to be standardized in future as CCN is claimed by the Future Content Networks (FCN) Group to be the Future Internet (FI). However, XIA lacks mechanism to mitigate replaying attack. Therefore, a solution for replaying attack has been proposed and implemented in this paper.

Nine existing solutions such as Diffie-Hellmann, Lamport's Password Authentication, S/Key One Time Password, Keung-Siu Protocol, Message Binding, Timestamp, Luo-Shieh-Shien Authentication Protocol, Yoon-Jeon Protocol, and Tseng-Jou Protocol have been analyzed to derive the requirements for the proposed protocol. Based on the derived requirements, the solution has been developed.

The protocol has been implemented in XIA prototype and has been proven to be able to mitigate the replaying attack. The proposed protocol has the following properties: First, There is a unique session key for each host in every session. Second, There is a checking process to ensure that the session key that is generated at each host is the same. Third, it has mechanisms to encrypt the messages and to conceal the identity of the hosts.

The proposed protocol has been evaluated to have more advantages over the reviewed existing solutions. It is more secure by having session key with length of 280 bits. Moreover, it is less complex as none of the random numbers used in the protocol are worthless. By applying the proposed protocol, XIA is now able to mitigate all of the reviewed attacks.

According to the current standard [21], a session key with a length of up to 280 bits is secure. In the future, when 280 bits is not enough, the size of the session key can be extended.

8. REFERENCES

[1] Anja Feldmann, "Internet Clean-Slate Design: What and Why?," in *SIGCOMM Computer Communication Review*. 2007, pp. 59–64. Volume 37, Number 3, ACM.

[2] Beny Nugraha, "Security Analysis Of Future Network Architectures," M.S. thesis, Hochschule Darmstadt, 2013.

[3] Future Internet Assembly (FIA) Future Content Networks (FCN) Group, "Technical Report. Why do we

need a Content Centric Future Internet?," pp. 1–23, 2009.

[4] XIA Project Team, "XIA Prototype," https://github.com/XIA-Project/xia-core/wiki, 2013, [Online; Accessed on 01-August-2013].

[5] Ashok Anand, Fahad Dogar, Dongsu Han, Boyan Li, Hyeontaek Lim, Michel Machado, Wenfei Wu, Aditya Akella, David Andersen, John Byers, Srinivasan Seshan, and Peter Steenkiste, "XIA: An Architecture for an Evolvable and Trustworthy Internet," in *Proceedings of the tenth ACM Workshop on Hot Topics in Networks (HotNets-X)*. 2011, pp. 1–32. Article No. 2, ACM.

[6] Hsu-Chun Hsiao, Tiffany Hyun-Jin Kim, Adrian Perrig, Akira Yamada, Samuel C. Nelson, Marco Gruteser, and Wei Meng, "LAP: Lightweight Anonymity and Privacy," in *Proceedings of the IEEE Symposium on Security and Privacy*. 2012, pp. 506–520, IEEE Computer Society.

[7] Hsu-Chun Hsiao, Tiffany Hyun-Jin Kim, Sangjae Yoo, Xin Zhang, Soo Bum Lee, Virgil Gligor, and Adrian Perrig, "STRIDE: Sanctuary Trail Refuge from Internet DDoS Entrapment," in *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS)*. 2013, pp. 415–426, ACM.

[8] Tiffany Hyun-Jin Kim, Lin-Shung Huang, Adrian Perrig, Collin Jackson, and Virgil Gligor, "Accountable Key Infrastructure (AKI): A Proposal for a Public-Key Validation Infrastructure," in *Proceedings of the 22nd international conference on World Wide Web*. 2013, pp. 679–690, International World Wide Web Conferences Steering Committee.

[9] Hannes Gredler and Walter Goralski, *The Complete IS-IS Routing Protocol*, Springer, 2004.

[10] Eun-Jun Yoon and Il-Soo Jeon, "An efficient and secure Diffie Hellman key agreement protocol based on Chebyshev chaotic map," *Communications in Nonlinear Science and Numerical Simulation*, pp. 23832389. Volume 16, Issue 6, 2010.

[11] Whitfield Diffie and Martin E. Hellman, "New Directions in Cryptography," *Journal IEEE Transactions on Information Theory*, pp. 644–654. Volume 22 Issue 6, 1976.

[12] Leslie Lamport, "Password Authentication With Insecure Communication," *Magazine Communications of the ACM*, pp. 770–772. Volume 24 Issue 11, 1981.

[13] Neil Haller, "The S/KEY One-Time Password System," in *In Proceedings of the Internet Society Symposium on Network and Distributed Systems*, 1994, pp. 151–157.

[14] Stephen Keung and Kai-Yeung Siu, "Efficient Protocols Secure Against Guessing and Replay Attacks," in *Proceedings, Fourth International Conference on Computer Communications and Networks*, 1995, pp. 105–112.

[15] Tuomas Aura, "Strategies against Replay Attacks," in *Proceedings of the 10th IEEE workshop on Computer Security Foundations CSFW'97*, 1997, pp. 59–68.

[16] Cdric Adjih, Daniele Raffo, and Paul Mhlethaler, "Attacks Against OLSR: Distributed Key Management for Security," *2nd OLSR Interop/Wksp.*, pp. 1–7, 2005.

[17] Jia-Ning Luo, Shiuhpyng Shieh, and Ji-Chiang Shen, "Secure Authentication Protocols Resistant to Guessing Attacks," *Journal of Information Science and Engineering*, pp. 1125–1143. Volume 22 No. 5, 2006.

[18] Huei-Ru Tseng and Emery Jou, "An Efficient Anonymous Key Agreement Protocol Based on Chaotic Maps," in *IEEE 13th International Conference on High Performance Computing and Communications (HPCC)*, 2011, pp. 752–757.

[19] Tsuji Takasuke, "A One-Time Password Authentication Method," M.S. thesis, Graduate School of Engineering, Kochi University of Technology, 2002.

[20] Sung-Ming Yen and Kuo-Hong Liao, "Shared authentication token secure against replay and weak key attacks," in *Information Processing Letters*. 1997, pp. 77–80. Volume 62 Issue 2, Elsevier North-Holland, Inc.

[21] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid, "Recommendation for Key Management Part 1: General (Revision 3)," pp. 1–147, 2012.