國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

基植於 MACsec 的網路存取控制架構

NAC: A Secure Network Access Control Framework for MACsec

研 究 生：劉雨芊　　Student：Yu-Chien Liu

指導教授：謝續平　教授　　Advisor：Dr. Shiuhpyng Shieh

中華民國九十八年六月

基植於 MACsec 的網路存取控制架構

研究生: 劉雨芊　　指導教授: 謝續平 教授

國立交通大學

資訊科學與工程研究所

摘要

在傳統的網路，資料在區域網路 MAC 層中傳遞時沒有任何安全協定保護，容易遭受竊聽、修改、僞造等攻擊。爲了解決問題 MAC 上安全的漏洞，IEEE 組織近期提出仍在制定當中的 802.1 MACsec 標準，達到區域網路中設備的身分認證、網路資源的存取控制、以及資料的機密性。然而，MACsec 在點與點之間訊息傳送時造成中間裝置上過多的加密計算量以及資料訊框傳送的延遲時間，和群組金鑰分送協定中沒有達到 forward/backward secrecy。在此篇論文中，我們將提出一個新的架構，包含一個點對點的金鑰交換協定及群組金鑰分送協定。其中，點對點金鑰交換協定可減輕訊息經過中間裝置運算加解密的次數，而群組金鑰分送協定則可減少金鑰伺服器傳送金鑰訊息並減少運算複雜度，用這兩個協定來改進 MACsec 標準中目前不足的地方。在論文的最後，我們會分析提出的方法與先前其他研究在訊息數量、加密運算花費、資料訊框延遲的比較，並且分析提出協定的安全性。

NAC: A Secure Network Access Control Framework for MACsec

Student: Yu-Chien Liu Advisor: Dr. Shiuhpyng Shieh

Department of Computer Science

National Chiao Tung University

Abstract

In conventional networks, such as Ethernet, network access at data link layer is not authenticated and controlled. Any network device connecting a network can send and receive network frames. Consequently, data frames can be eavesdropped, modified and forged by an adversary who plugs in the network port. To cope with the problem, IEEE 802.1 MACsec has been proposed recently to authenticate a network device and its access to the local area network. However, MACsec requires high computation overhead, and does not provide forward and backward secrecy for group key distribution. Further enhancement is desirable. In this paper, we will propose a secure network access control framework (NAC) for MACsec, including the network access control architecture along with two key distribution protocols. The station-to-station key handshake protocol is for pairwise communication, while the group key distribution protocol allows a group of hosts in a local area network to communicate with each other in a secure and efficient way. A Group handshake protocol is also proposed to handle group joining and leaving. The design and implementation of NAC will be illustrated; the overhead of the proposed group key distribution protocol will be evaluated and compared with related work. The result shows that our protocols incur the lowest computation cost as well as communication overhead.

誌謝

首先，對於指導教授謝續平教授，誠摯的感謝這兩年來的教導，不論是研究上的協助，亦或是生活上的開導，我深深覺得研究所讓我受益良多，相信在未來的就業上也受益匪淺。更讓我這位從私立畢業的學生也能體會到交大是如此溫暖的學校。此外，感謝實驗室的助理群：明華、順瑩、Michael，每每幫助我生活上的大小事務，並在碰到問題時適時的給我勇氣，如果沒有你們，我的生活早已雜亂無章。而學長姐們 ：佳偵、繼偉、政仲、經偉、鼎鈞、佳純、子逸，總是會在我徬徨無助時伸出援手，一起討論研究，並不吝嗇的和我分享研究所的經驗，也給了我最棒又最溫馨的實驗室氣氛。同學們：家銘、家維、秉翰、蘇偉，我們一起成長，一起歡笑，一起共苦度過了這兩年，總是帶著革命精神，共同分擔壓力、計劃，沒有你們也不會讓我如此快速成長。學弟們也要謝謝你們時常帶給我們歡笑，並給我不少的幫助。當然，我更要謝謝我的家人，在我最艱苦的時候時常給我鼓勵，並慢慢讓我了解即將步入社會的做人道理。

謝謝ＤＳＮＳ實驗室的每一位夥伴，謝謝你們容忍我的壞脾氣，在未來的路上，我會以身為此實驗室一份子為榮，抬頭挺胸繼續邁進。

最後，我要感謝交大資工系羽的夥伴們，有你們的陪伴讓我研究所不孤單，在洩氣、失望、疲累時，能一起歡笑、一起運動、一起旅行。研究所也是要活得健健康康，快快樂樂。

有了大家的陪伴，很開心的度過這兩年短暫又璀璨的日子！有快樂、有悲傷、有驕傲，這些經歷都值得讓我一再回味。這些情誼與收穫，我會繼續保存著！

謝謝各位，我要畢業了！！

Table of Content

List of Figures

List of Tables

Chapter 1

Introduction

Network traffic exchanged in local area network (LAN) is usually transported by data link layer protocol. Most data link layer protocols don't support security mechanism and hence are vulnerable from several insider attacks which may not be detected by upper layer. There are several security protocols in network, transport and application layer of a network, such as Internet Protocol Security (IPsec), Secure Sockets Layer (SSL), and Transport Layer Security (TLS). However, these are end-to-end security protocol that cannot protect the messages in LAN. Malicious user resides in local network performs successful attacks with higher probability than those adversaries from outer network. The inside data frames would not be filtered with firewall or intrusion detection system (IDS).

The common security issues on data link layer are MAC flooding, ARP poisoning, and broadcasting. The fixed-size Content-Addressable Memory (CAM) tables store MAC addresses, port numbers, VLAN ID at switch. The attacker floods the switch with forged ARP packet until the CAM table is full. MAC flooding causes the switch becomes hub-like mode that switch broadcasts incoming data frames to all the port. ARP is used to map an IP address to a MAC address. When a host desires to find a MAC address for a particular IP address, it broadcasts an ARP request to the LAN. ARP is not authenticated that the attacker is able to send ARP packets with spoofed content to victims. In ARP poisoning attack, the victims may suffer from man-in-the-middle (MITM) attacks and Denial-of-service (DoS) attacks. MAC flooding and ARP poisoning are caused by the packets are transmitted without message authenticity.

Layer 2 also suffers from broadcasting attack that every device on the network has to consume CPU resources to deal with broadcast packets. In order to lessen the overhead of

broadcasting, virtual LAN (VLAN) is proposed to provide the segmentation of broadcast domain that the devices in the different VLANs cannot receive others broadcast frames. However, data broadcasted on local network are transmitted in plaintext and can be eavesdropped easily by every device connected over the same network. It is easy to leak the information on network. Layer 2 attacks take place because the frames are lack of authenticity and protection.

In addition, it is difficult to manage the devices attached to the network using higher layer secure protocols. Devices can access the network directly when moving to every physical location in the LAN without changing IP address. Hence, the devices cannot be restricted connecting to the fixed ports and locations, which the users can listen the network traffic and connect to the access points directly without permission.

IEEE 802.1X [2] standard is a framework that provides port-based access control using authentication. It only allows the authenticated and authorized devices are capable of accessing the network. However, data frames are not protected where any attackers is able to tamper and eavesdrop the messages.

MACsec [1][2], complementary to higher layer protocols, is a MAC layer security protocol for point-to-point or group communication data protection in local network. IEEE 802.1ae [1] defines the frame format, encryption algorithm, data authentication, frame processing. IEEE 802.1af [2], combined into IEEE 802.1x-rev, performs authentication and cryptographic key distribution. MACsec provides data confidentiality, data integrity, data origin authenticity, and replay protection. Devices have to be defined the access rights base on a variety of different criteria, and access local networks only after authenticated and authorized by the authentication server. Hence, MACsec can drop frames which are without authorization or are modified by any devices actively.

Linksec [3], based on MACsec design, is proposed by Intel which provides secure data communication between two immediately connected devices. However, Linksec has powerful

switches support which every incoming data frames shall be decrypted and re-encrypted.

There are plenty of applications which are based on broadcasting behavior in LAN, such as file sharing, multi-player gaming, broadcast conference, and ARP packets. For example, the NetBEUI protocol on which Microsoft network neighborhood based on broadcast messages. Most LAN games use UDP broadcasting to announce game establishment. However, these messages are often destined to only some specific users instead of all the connected devices under the same LAN. Therefore, a secure group communication scheme must be proposed to address the problem that only the members of the group can obtain the contents by encrypting and decrypting the messages. Thus, the protection of broadcast frames is significant in data link layer security.

However, MACsec doesn't define a well secure group communication protocol. MACsec is short of forward secrecy and backward secrecy during the group membership changes. Secure group communication systems [4, 5, 6, 7, 8, 9, 10, 11], are proposed for group key distribution and management. During the group membership changes, a new group key must be updated and distributed to all of the group members. The conventional method for secure group communication needs O(N) rekeying messages, where N is the number of group members. These secure group communication systems alleviate the number of rekeying messages to at least O($\log N$).

1.1 Contribution

In this paper, a station-to-station key handshake protocol is proposed. It alleviates the communication time and computation delay on the internal pass-through switches, while authenticity of data frame is only verified by the switches. Second, the scheme compatible with MACsec is designed. Compare to other secure group communication schemes, the

number of rekey messages reduces to $\log_2 N$. We provide the procedure of group key rekeying which achieves forward and backward secrecy. When a station joins or leaves the group, it is not able to access previous and future messages. Besides, the tree balance scheme is provided. Finally, the communication cost and computation cost are analyzed. Our secure group communication scheme uses smallest rekey messages and computation time.

1.2 Synopsis

The rest of the paper is organized as follows. Chapter 2 gives the detailed description of MACsec. Chapter 3 reviews other relevant research in secure group communication. In Chapter 4, we propose a new architecture, group key distribution protocol and STAKey handshake protocol. Security and performance are evaluated in Chapter 5 and Chapter 6. Finally, the conclusion is given in Chapter 7.

Chapter 2

MACsec

MACsec (Media Access Control Security) [1][2] is a secure protocol for MAC layer. The important features are connectionless user data confidentiality, data integrity, data origin authenticity, and replay protection. It restricts the use of local or metropolitan area networks to secure communication between authenticated and authorized devices. IEEE 802.1AE defines encryption algorithm and frame format. IEEE 802.1X-revision specifies about port-based network access control, authentication and key distribution.

Device connecting to the network should mutually authenticate with authentication server, exchange session key, and obtain the permission to access the network. MACsec defines two methods for authentication, pre-shared key (PSK) or EAP (Extensible Authentication Protocol) [12]. Authentication frames are only be passed through the uncontrolled port of the access point. Using EAP, the authentication server and the device would negotiate a pairwise secret key shared between each other. And then the authentication server sends the secret key to the access point. After authentication, Authentication server transfers the access policy to the access point that allows the device to communicate with others. During key distribution, the device exchange the group session key with the key server.

MACsec provides secure group communication concept among authenticated and authorized group members. The devices in the same domain can receive the broadcasting messages, but only the devices own the group key can decrypt the messages correctly. Thus, the use of broadcasting mechanism achieves multicast the messages to the particular devices. The following are the three secure relationships between the devices in MACsec.

i. Secure Connectivity Association (CA)

A secure relationship comprises a fully connected subset of the service access points in devices attached to a single LAN that are to be supported by MACsec. Each CA is supported by unidirectional Secure Channels. CAK (secure Connectivity Association key) is a secret key possessed by the members of a given CA. CAK can be derived from EAP MSK

ii. Secure Channel (SC)

A security relationship used to provide security guarantees for frames transmitted from on member of a CA to the others through the use of symmetric key cryptography. Each SC is supported by an overlapped sequence of Security Associations thus allowing the periodic use of fresh keys without terminating the relationship.

iii. Security Association (SA)

A security relationship that provides security guarantees for frames transmitted from one member of a CA to the others. Each SA is supported by a single secret key, or a single set of keys where cryptographic operations used to protect one frame require more than one key. The key protecting the SA is called SAK (Secure Association Key).

There are four kinds of keys, pairwise CAK, group CAK, pairwise SAK, and group SAK. CAK is a secret key possessed by members of a given CA that only the members of CA own the keys. Pair-wise CAK and group CAK are responsible for pairwise SAK and group SAK distribution protection respectively, and message authentication. Pairwise SAK and group SAK are responsible for data protection and message authentication.

Each of the keys used of key distribution is derived from the CAK using the AES Cipher in CMAC mode. The ICK (Integrity Check value Key) is not directly distributed by any protocol, but only derived from the CAK, verification of the ICV both ensures that the contents of the PDU have not been modified but also that it was composed by a system that possessed the CAK. The KEK (Key Encrypting Key) is used to distribute SAKs and group

CAKs to system protected with AES key wrapping that possesses the CAK.

Chapter 3

Related work

In this chapter, Linksec, an architecture based on MACsec, proposed by Intel is given. Besides, some general secure communication schemes, which are compatible with MACsec, are described.

3.1 Linksec

Linksec is a hop-by-hop architecture designed by Intel [3], which provides several advantages for encrypting enterprise network traffic compared to end-to-end encryption based solution. Every device in the same network has a secure channel with its neighboring devices. The access point stores O(m) keys, where m is the number of connections. When the data frames sent from station A to station B as shown in Figure 3.1, they shall be decrypted and re-encrypted by switch A, router, and switch B. This bottleneck of this solution is the switches and routers that they have a lot of overhead for encryption operation.

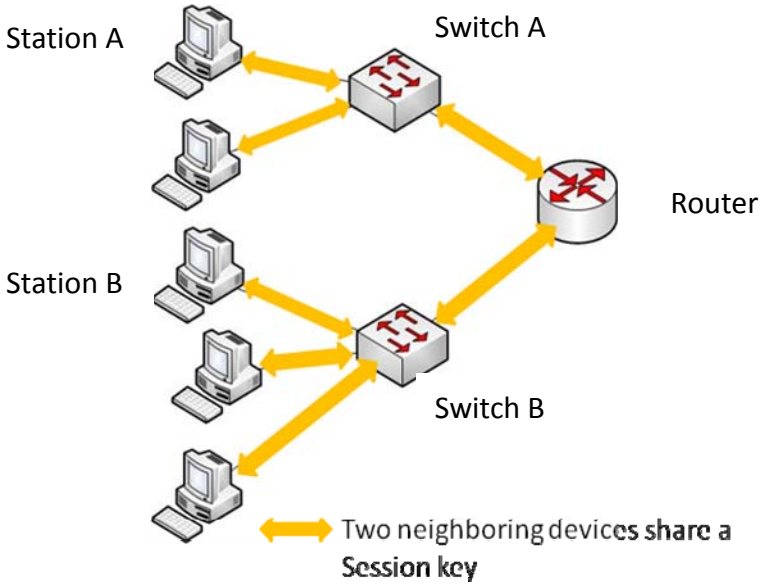


Figure 3. 1 Linksec architecture

3.2 Secure Communication Schemes

There are many researches on secure group communication such as GKMP, BMFT, LKH, and OFT. We give a brief introduction about their features, advantages, and disadvantages.

3.2.1 Group Key Management Protocol

Group Key Management Protocol (GKMP)[4] is a simple way for group key management. In this approach, Key Distribution Center (KDC), a trust third party, is responsible for distributing symmetric group key to the members of the group. When a new member joins the group, KDC would create and send a Group Key Packet (GKP) which contains the current Group Traffic Encrypting Key (GTEK) and future Group Key Encrypting Key (GKEK). When rekeying, KDC generates a Group Rekey Package (GRP), which contains the GKP encrypted with GKEK. Then, KDC encrypts GRP with current GKEK and distributes to the entire group. However, all of the group members, including departing member, know the GKEK; there is no solution for providing forward secrecy when a member leaves the group.

3.2.2 Logical Key Hierarchy (LKH)

The key graph approach [5] has two important types, key tree and key start. Key star is a special class of a secure group that the tree degree of the root is equal to the size of the group, as shown in Figure 3.2. Key star is an inefficient scheme which the key server needs to generate and distribute N message for rekeying, where N is the size of group.

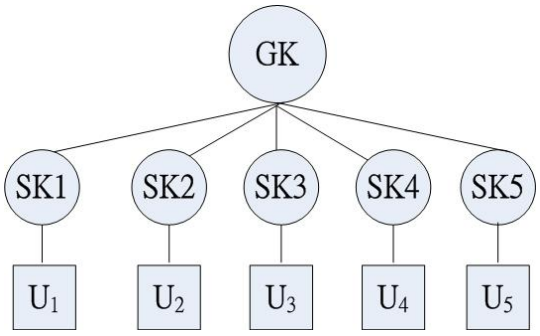


Figure 3. 2 Key star

Key tree is a single root tree that the leaf nodes are individual keys, and the other tree nodes are the auxiliary keys. In order to reduce the overhead of rekeying from linear to logarithm, the solution is are the logical key hierarchy (LKH) schemes [5][6]. The group leader (or key server) maintains the tree topology. The root represents the group key and the nodes of the tree are the auxiliary key encryption keys. The leaves of the tree are the group members and maintain the corresponding keys in the path from the leaf to the root. In balance tree, a member obtains at most $(\log_d N)+1$ keys, where d is the degree of the logical tree.

The number of the rekey messages for group membership changes in balance tree reduces from N to $2\times\log_d N$. All key nodes in the path from the changed node to the root have to be updated. In joining operation, the updated keys at each level have to be sent twice. One is unicasted to the new member encrypted with individual key, and the other is multicasted to the old members encrypted with the previous corresponding keys before rekeying. In leaving operation, the updated keys at each level have to be sent d times which are for each branch subgroups. Therefore, the number of rekey messages for joining and leaving are $2\times\log_d N$ and $d\times\log_d N$ respectively.

Figure 3.3 shows a hierarchical tree with eight members, where tree degree is 2. U-node is a group member and GK is the group session key. U_3 maintains $(\log_2 N)+1$ keys including GK, K_2, K_5, and K_{10}. Suppose that U_3 wants to leave the group and the remaining

seven members form a new group. The key server generates new auxiliary keys {GK', K_2', K_5'}. The key server encrypts K_5' with K11 before sending to U_4, and K_2' with K_4 and K_5' before sending to K_4 subgroup and K_5 subgroup respectively. And then, the key server generates new group session key GK' and encrypts with K_2' and K_3 before sending to K_2 subgroup and K_3 subgroup respectively.

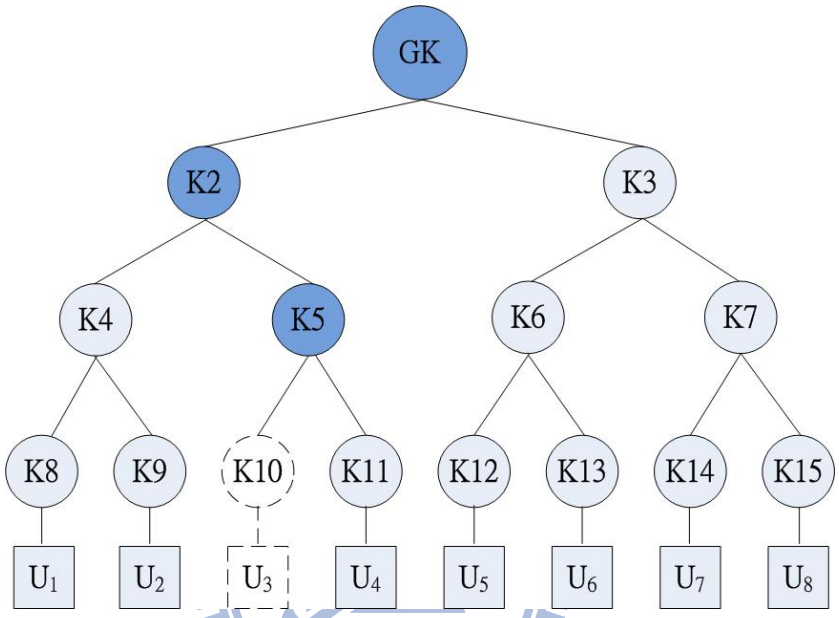


Figure 3. 3 Logical Key Hierarchy Tree

The major drawback of LKH is that every member must update their state whenever any members join or leave the group which limits the scalability for bulk membership changes.

3.2.3 One-way Function Tree (OFT)

One-way function tree (OFT)[7] is an improved approach for LKH. The user node stores $\log_2 n+1$ keys including group key and the blinded key of the sibling on the path from user to the root. The blinded key of i is $K_i^B = f(K_i)$, where f is a one-way function. The user is able to compute the ancestor keys from its position to the root using $K_i = m(f(K_{2i}), f(K_{2i+1}))$.

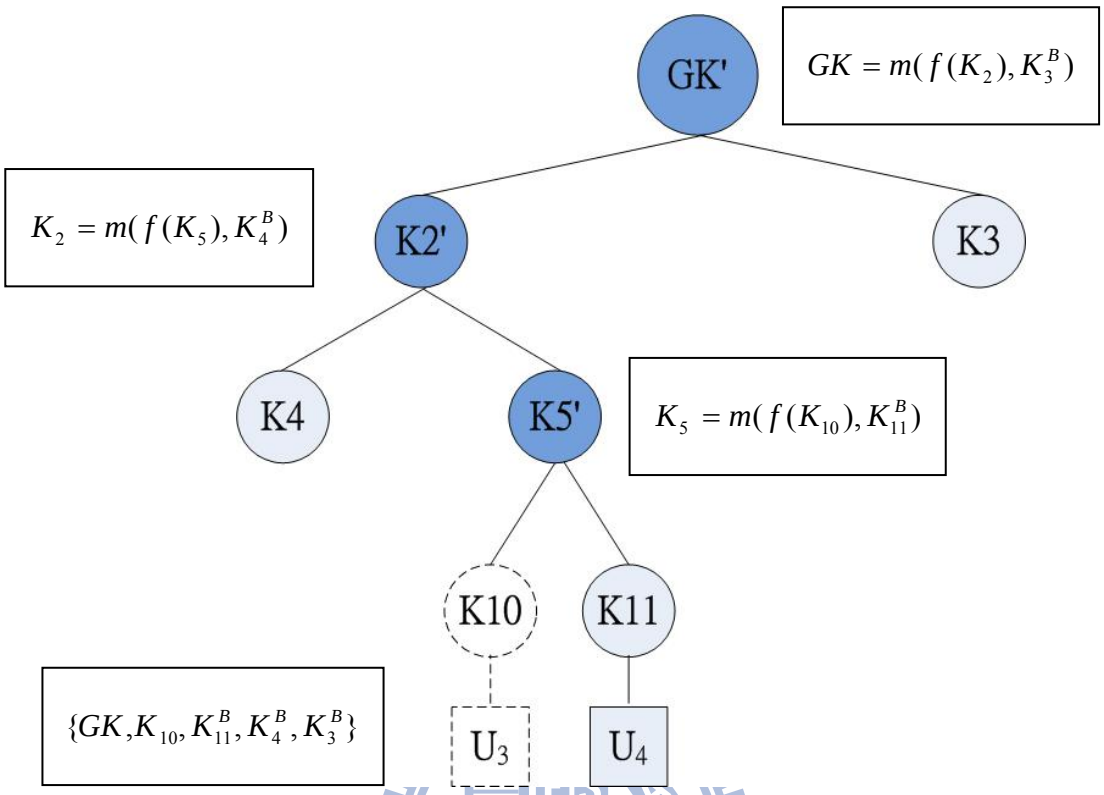


Figure 3. 4 One-way function tree

For example as shown in Figure 3.3, U_3 has $\{GK, K_{10}, K_{11}^B, K_4^B, K_3^B\}$, and it can use this keys to compute group key GK. The comparison between LHK, OFT reduces the number of the rekey message for each leaving from $2 \times \log_2 n$ to $\log_2 n$.Nevertheless, for each join operation, the number of the rekey message is still $2 \times \log_2 n$.

However, OFT has the same overhead that is similar to LKH. The drawback of OFT is the key server has to maintain $2^{\lg n+1} - 1$ keys.

Chapter 4

Proposed Scheme

In this chapter, we will describe the details of our proposed scheme, including the network access architecture, station-to-station key handshake protocol, and group key distribution protocol. The design and implementation will be introduced afterwards.

4.1 Architecture

In our new architecture, there are four parties involved. They are access point, stations, authentication server, and key management center, as shown in Figure 4.1. Devices shared with a group secret key are in the same CA. Thus devices in the same broadcast domain have the group key to protect group messages. The access control policy for each device is configured by the administrators.

Authentication server is a server that provides authentication service to users, devices, or other systems via networking. In MACsec, authentication server is responsible for authentication and authorization. It manages the privacy information such as users' certificates, pre-shared keys or passwords for authentication. Administrators define and manually configure the access right base on a variety of different criteria.

Key management center is a central server that records and generates the keys. There is more than one network in an enterprise local area network. The networks are formed the different groups which have their own key servers. The key servers maintain the keys used in their groups. However, this situation would cause that the different groups use the duplicate keys. In addition, the storages of the key servers (routers or switches) are restrained. We thus proposed the key management center in our MACsec architecture. The key management center has a secure tunnel with each key server, where the upper layer security protocols are

used, such as IPsec. Key server generates a new key and forwards it to the key management center. The key management center check whether the key is duplicate. If not, it records the key. Otherwise, it generates the new key and notifies the key server to update the key.

Access point is responsible for access control and message authenticity verification and acts as a key server for pairwise and group key distribution. Every station in the same network shares the pairwise secret key and pairwise session key with the access point. When the station desires to communicate with a station outside the network, the station encrypts the data frames by pairwise session key shared with the access point. When a station desires to communicate with another station in the same network, they first negotiate a station-to-station key (STAKey) over key server (section 4.4). And then they protect the data frames with this STAKey.

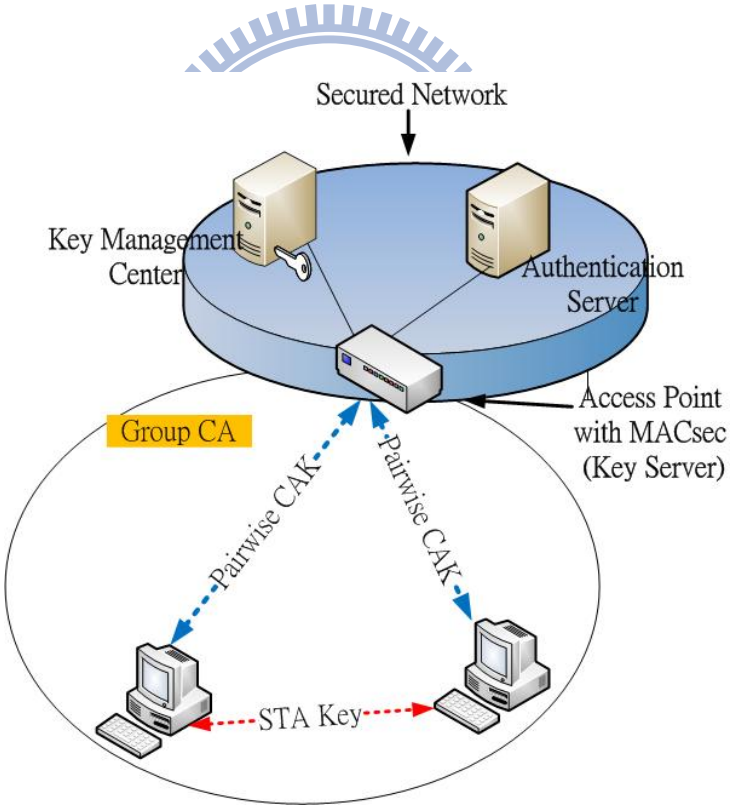


Figure 4. 1 Our MACsec architecture

4.2 Initialization

The authentication server mutually authenticates to the station when connecting to the network. MACsec claims that this standard should implement EAP-TLS (RFC 4346) [13]. The EAP result is a master secret key called EAP-MSK, which is shared between authentication server and station. Authentication server sends EAP-MSK and authentication success to access point that the station is enabled to access the network. After authentication, key server and station compute pairwise CAK and exchange pairwise SAK. Thus, the station can use pairwise CAK to negotiate group keys and station-to-station keys.

4.3 Notation

This section we introduce the notations used in group key distribution protocol and station-to-station key handshake. The notations and their descriptions are listed in Table 4.1.

Symbols	**Descriptions**
ID_i	Identifier (index) of i
MN_i	Message Number of user i or the members' MN of group i Against replay attack
MAC_i	MAC address of i
G_KEK(i) G_ICK(i)	Group Key Encrypting Key, Group Integrity Check Value Key of group i Derive from group CAK using AES-CMAC
P_KEK(i) P_ICK(i)	Pairwise Key Encrypting Key, Pairwise Integrity Check Value Key of user i Derive from pairwise CAK using AES-CMAC
G_CAK'	Updated Group CAK Secret key for group key distribution

P_SAK G_SAK	Pairwise SAK, Group SAK
(m)K	Encrypt message m with K

Table 4. 1 Notation

4.4 STAKey Handshake Protocol

In this section, we propose a station-to-station key (STAKey) handshake protocol. The procedure of the protocol is shown as Figure 4.8. It contains three parties, station A, station B and key server (KS). Station A and station B share with key server the pairwise CAKs respectively. Station A exchanges the messages with key server using P_ICK_A and P_KEK_A for data protection, and station B using P_ICK_B and P_KEK_B. Integrity check value is appended to each message. Key server, station A and station B perform the following steps, where station A desires to exchange a session key with station B:

1. Station requests the key server to perform the station-to-station key distribution process by sending out the message $\{MAC_B, MN_A, STAKey-request\}, ICV_{P\_ICK_A}$ to the key server in plaintext. MAC_B is used to inform the key server that station A desire to communicate with station B.
2. The key server generates a random number as STAKey shared between station A and station B. The key server encrypts the $STAK_{A,B}$ with P_KEK_B, and sends out the message $\{MAC_A, MN_{KS}, MN_A, (STAKey_{A,B})P\_KEK_B\}, ICV_{P\_ICK_B}$ to station B. MAC_A is used to inform station B that the STAKey is shared with station A. Station A is able to verify whether station B obtain the same STAKey by MN_A in the message.
3. The key server encrypts the $STAK_{A,B}$ with P_KEK_A, and sends out the message

$\{MN_A, MN_{KS}, (STAKey_{A,B})P\_KEK_A\}, ICV_{P\_ICK_A}$ to station A.

4. Station B send out the message $\{MN_B, MN_A, (MN_B, MN_A, STAKey_{A,B})STAKey_{A,B}\}, ICV_{STAKey_{A,B}}$ which contains the verification information $(MN_B, MN_A, STAKey_{A,B})STAKey_{A,B}$. MN_B is sent to station A that station B can check whether station A has the same STAKey with station B.
5. Station A send out the message $\{MN_B, (MN_B, STAKey_{A,B})STAKey_{A,B}\}, ICV_{STAKey_{A,B}}$ which contains the verification information $(MN_B, STAKey_{A,B})STAKey_{A,B}$.

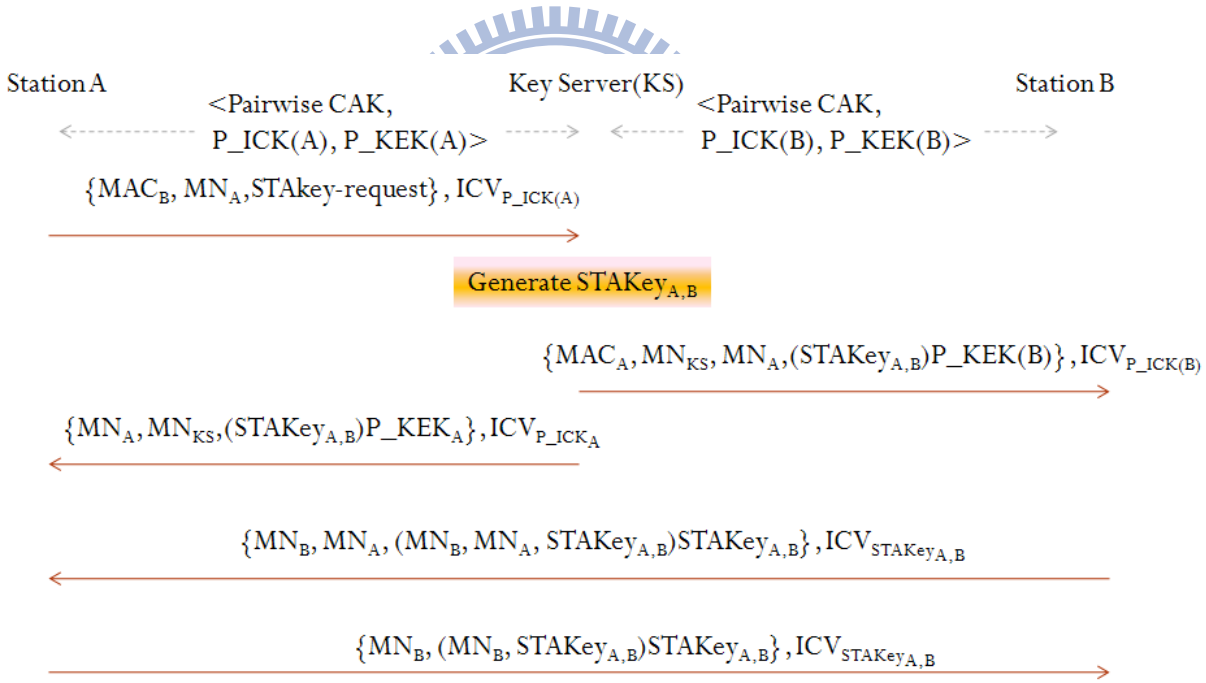


Figure 4. 2 The procedure of STAKey handshake

4.5 Group Key Distribution

In this section we propose a new One-way Function tree (OFT) scheme based on LKH for MACsc secure group communication.

The root of the tree is represented as group key and the leaves are user nodes. The other nodes are the auxiliary keys which assist key server in updating and protecting new group key. Key server is responsible for maintaining the tree structure. Every auxiliary node is associated with a unique index. Assume the root has index 1. The node has an index i, its children are fond at indices 2i and 2i+1, while its parent is found at index $\left\lfloor \frac{i}{2} \right\rfloor$. Key server stores 2N-1 keys which are the group key GK and all the auxiliary keys K_i. User U_i stores $(1+\log_2 N)$ keys which are the keys on the path from U_i to the root. The auxiliary keys are stored in secure locations such as smart cards, and etc.

Since the group membership changes, the group key and update key nodes shall be updated. Update key nodes are the auxiliary keys on the path from U_i to the root, where U_i is the user node joining or leaving the group. The example is shown as Figure 4.2. U_{10} is the changing node and {GK, K_2, K_5, K_{10}} are the update key nodes.

Every member in the group storing the update keys must update the auxiliary keys. Key server generates a random number as the new group key and sends out to the entire group. On the receipt of new group key GK', the members and key server use a one-way function (OF) to compute new update keys. The one-way function (OF) is AES-CMAC-128 [14], SHA-256, HMAC, and etc. The formula used for computing the new auxiliary key is shown as follows:

$$K_i' = OF(K_i \oplus GK', \text{"Auxiliary key index i"})$$

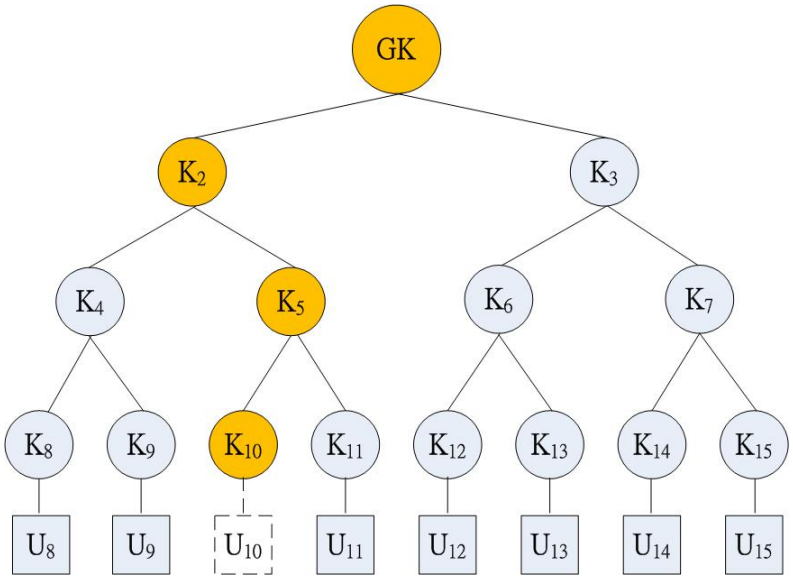


Figure 4. 3 Our One-way Function

In following sections, we describe the detail of join and leave operation using our OFT scheme compatible with MACsec structure. In join and leave sections, the process is used for updating new group CAK. The group SAK is able to be protected by new group CAK and broadcasted to all group members.

4.5.1 Join

When a station joins the group, the key server has to find the index in the tree for the new joining member and update the new group and auxiliary keys to address backward secrecy that prevent the station from gaining the previous messages.

Key server performs following steps during a new station U_q joins:

1. Find a node K_p with no child and smallest index. K_p is a tree leaf with user node U_p.
2. The index of U_p becomes 2p, and the new member U_q is (2p+1). N_p becomes the parent of U_p and U_q.
3. Generate a random number as new group key. Use the group key GK to encrypt new

group key GK' and broadcast to all the members. The message contain the index of U_p.

4. Use the pairwise key shared with U_p to encrypt the new auxiliary key K_{2p} and send to U_p.
5. Use the pairwise key shared with new member U_q to encrypt the auxiliary key on the path from U_q to the root and send to U_q.
6. After group key updated, it uses $K_i' = OF(K_i \oplus GK', \text{"Auxiliary key index i"})$ with new group key GK' to compute the new auxiliary keys stored in its local storage.

Old members perform following steps during a new station U_q joins:

1. Decrypt the new group key GK' with the group key GK.
2. Compute the new auxiliary keys store in their storages using $K_i' = OF(K_i \oplus GK', \text{"Auxiliary key index i"})$ with new group key GK'.

New member U_q decrypts the keys with pairwise key and stores them in the storage.

The topology of key tree for a new member join is shown in Figure 4.3. Key server finds the node K_5 with smallest index. The index of U_5 becomes U_{10} and key server generates a new auxiliary key K_{10} and sends to U_{10}. The index of new member is U_{11} . Key server computes the new auxiliary keys {GK', K_3', K_4', K_{10}'} and sends to U_{11}.

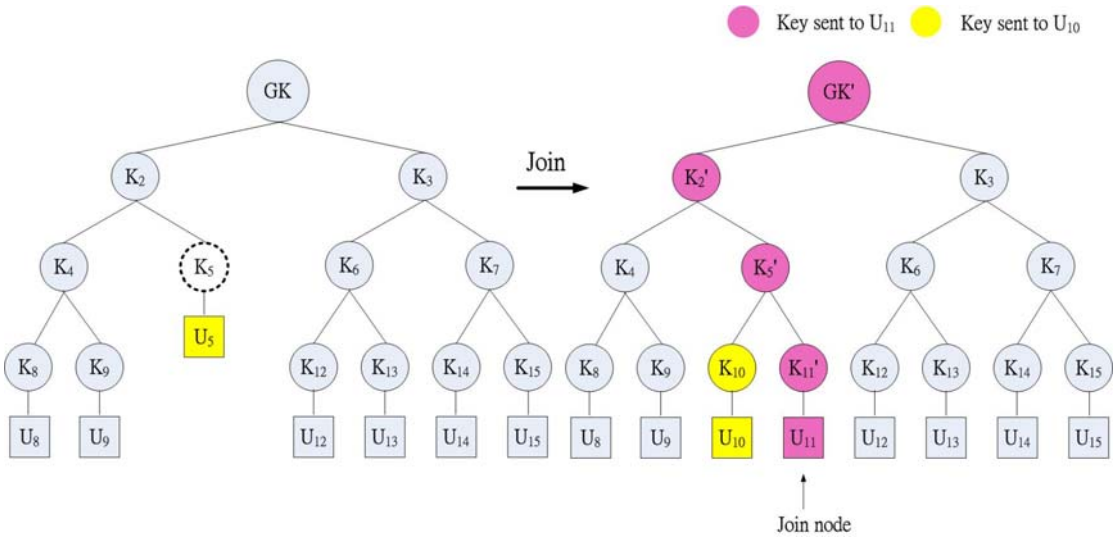


Figure 4. 4 The topology of key tree for a new group member joins

The procedure of key distribution protocol for join is shown as figure 4.4. It contains three parties, station A (join member), old group members (G) and key server. Key server pre-shares pairwise CAK (P_CAK), ICK (P_ICK) and KEK (P_KEK) with station A, and old group CAK (G_CAK), ICK (G_ICK) and KEK (G_KEK) with old group members. Due to station A does not have any group information, it can only unicast the message with key server. Integrity check value is appended to each message. The following is the steps of group key distribution for new member join:

1. Station A requests the key server to perform the group CAK distribution process by unicasting the message $\{MN_A, GroupCAK-request\}$, $ICV_{P\_ICK(A)}$ to the key server in plaintext.
2. The key server broadcast the message $\{MN_{KS}, MN_G, GroupCAK-updating\}$, $ICV_{G\_ICK(G)}$ to all the old group members to notify them to perform the group CAK updating process. MN_G is a set of message numbers of the members in group G.
3. All the group members send back the message

$\{MN_G, MN_{KS}, GroupCAK - response\}$, $ICV_{G\_ICK(G)}$ to the key server that they are ready to receive the new group key.

4. The key server generates a random number, using strong random number generator [17], as new group CAK G_CAK', which is encrypted by G_KEK(G). The message $\{ID_A, MN_{KS}, MN_G, (G\_CAK')G\_KEK(G)\}, ICV_{G\_ICK(G)}$ is sent out to all the members. The ID_A is used for notifying the members which key nodes should be updated.
5. Meanwhile, the key server computes the corresponding new auxiliary keys and encrypts G_CAK' and new keys with P_KEK(A) and unicasts the message $\{ID_A, MN_{KS}, MN_A, (G\_CAK')P\_KEK(A), (AKs)P\_KEK(A)\}$, $ICV_{P\_ICK(A)}$ to station A.
6. The response message from all the old members are $\{MN_G, MN_{KS}, (MN_p \| G\_CAK')G\_KEK'\}, ICV_{G\_ICK'}$, where p is the sender in group G. other group members and the key server can verify that all the members obtain the correct G_CAK'.
7. Meanwhile, the response message from station A are $\{MN_A, MN_{KS}, (MN_A \| G\_CAK')G\_KEK'\}, ICV_{G\_ICK'}$ that the key server can verify that station A obtain the G_CAK'.
8. The key server sends the message $\{MN_{KS}, MN_B, (AK)P\_KEK(B)\}, ICV_{P\_ICK(B)}$ to the station B, which station B is the sibling of station A (described above).
9. At least, station B responds a message $\{MN_B, MN_{KS}, (MN_B, AK)P\_KEK(B)\}, ICV_{P\_ICK(B)}$ to the key server. After verification all the member obtains the new keys, the process is accomplished.

In MACsec, an EAP frame is capable of including many keys that the auxiliary keys transmitted to the new group member can be sent by only one message.

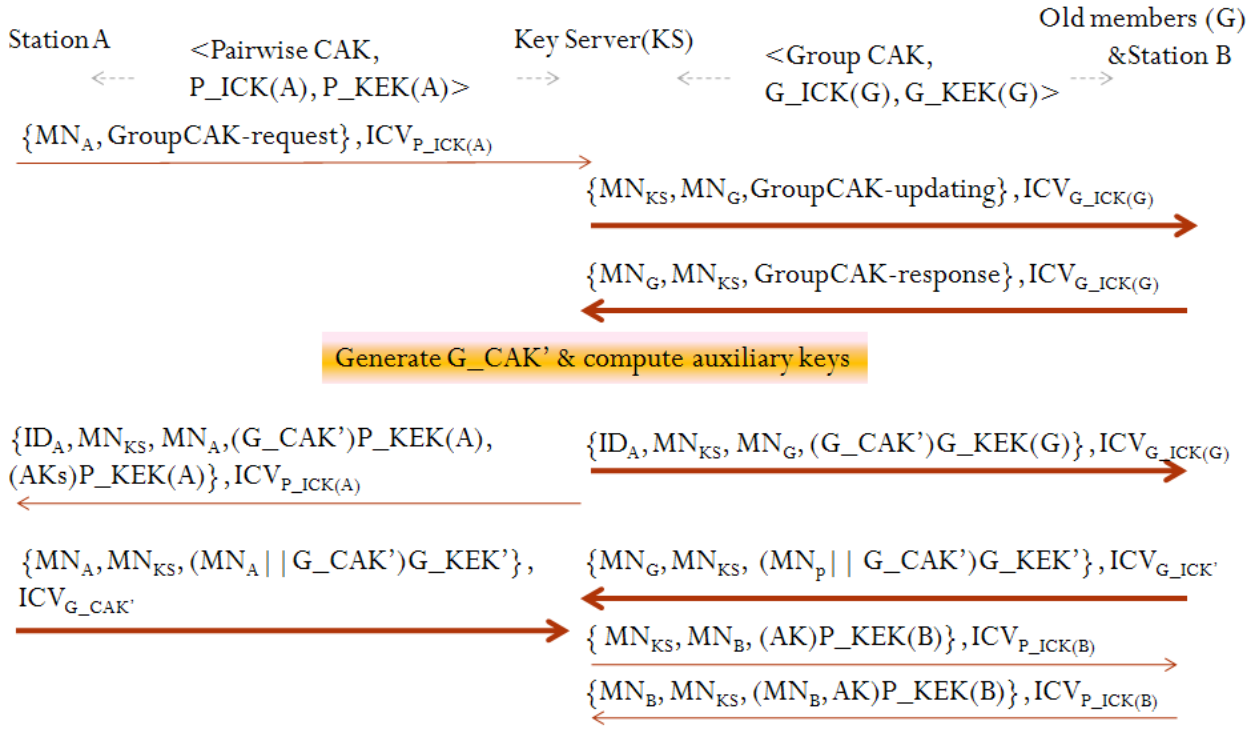


Figure 4. 5 Group CAK distribution for join

4.5.2 Leave

When a station leaves the group, the key server has to notify old members to update group key and the auxiliary keys to address forward secrecy that prevent the station from accessing the future messages.

Key server performs following steps during a station U_i leaves:

1. Key server generates a random number as the new group key encrypted by the siblings of update node keys respectively.
2. The parent of U_i is replaced by U_j. Key server removes key node i. The indices of

its sibling nodes and the nodes in U_j's subtree are updated to $\left\lfloor \frac{p}{2} \right\rfloor$, where p are their original indices respectively.

3. After group key updated, it uses new group key GK' to compute the new auxiliary keys stored in its local storage.

Old members perform following steps during a station U_i leaves:

1. On the receipt of rekey message from key server, all the members use the corresponding auxiliary keys to decrypt the new group key GK'.
2. After group key updated, it uses new group key GK' to compute other new auxiliary keys stored in its local storage.

For example, the topology of key tree for a group member U_{11} leave is shown in Figure 4.5. The update key nodes are {GK, K_2, K_5}. Key server encrypts the new group key GK' with {K_3, K_4, K_{10}} respectively and sends out to the members. The parent of U_{11} is replaced by U_{10} and the index of U_{10} is updated to U_5.

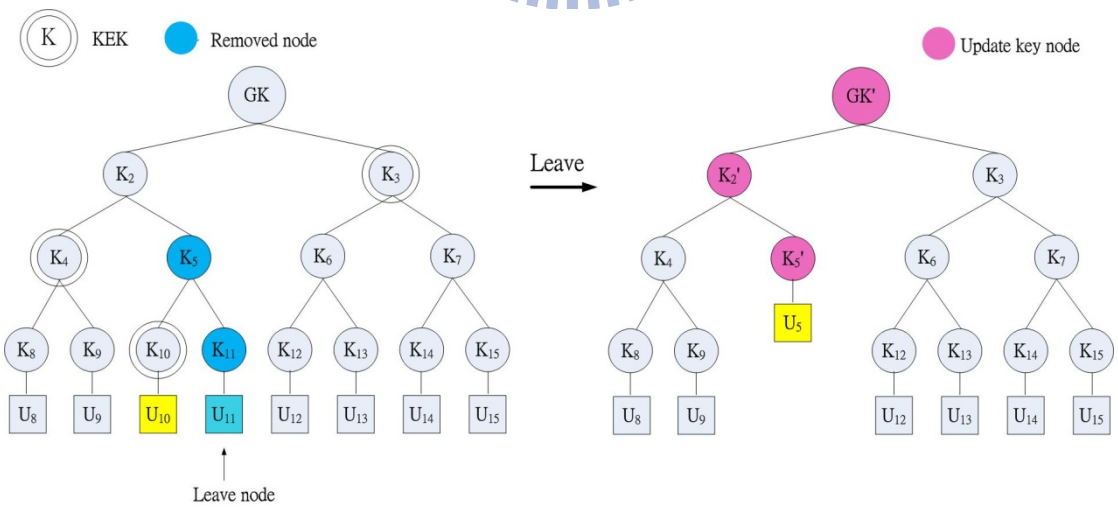


Figure 4. 6 The topology of key tree for a group member leaves

The procedure of key distribution protocol for U_i leave is shown as figure 4.7. In this

case, we only provide the procedure between key server and three subgroups. It contains three parties, subgroup (Ga), subgroup (Gb) and key server (KS). The subgroup is form of the sub-tree of the key tree with the auxiliary key as the sub-tree rooted key. The key server pre-share auxiliary key G_ICK(Ga) and G_KEK(Ga) with group Ga , and G_ICK(Gb) and G_KEK(Gb) with group Gb. Due to U_i does not know the subgroup keys, the rekey will not be access by departing station. Integrity check value is appended to each message. The following is the steps of group key distribution for member U_i leave:

1. The key server notifies remaining members to perform the group CAK distribution process by broadcasting the message $\{ID_i, MN_{KS}, MN_{Ga}, GroupCAK-update\}, ICV_{G\_ICK(Ga)}$ in plaintext to subgroup Ga. MN_{Ga} is a set of message numbers of the members in group Ga. ID_i is the index of leave station that others member is able to compute the corresponding update key nodes.
2. Same as step 1, the key server sends out the message $\{ID_i, MN_{KS}, MN_{Gb}, GroupCAK-update\}, ICV_{G\_ICK(Gb)}$ in plaintext to subgroup Gb.
3. All the members in subgroup Ga respond the message $\{MN_{Ga}, MN_{KS}, GroupCAK-response\}, ICV_{G\_ICK(Ga)}$ to the key server that they are ready to receive the new group key.
4. Same as step 3, all the members in subgroup Gb respond the message $\{MN_{Gb}, MN_{KS}, GroupCAK-response\}, ICV_{G\_ICK(Gb)}$ to the key server.
5. The key server generates a random number, using strong random number generator, as new group CAK (G_CAK') which is encrypted with G_KEK_i and broadcasts the message $\{MN_{KS}, MN_{Ga}, (G\_CAK')G\_KEK_i\}, ICV_{G\_ICK(Ga)}$ to subgroup Ga members.

6. Same as step 5, the key server broadcasts the message $\{MN_{KS}, MN_{Gb}, (G\_CAK')G\_KEK_i\}$, $ICV_{G\_ICK(Gb)}$ to subgroup Gb members.

7. The response messages from subgroup Ga members are $\{MN_{Ga}, MN_{KS}, (MN_p \| G\_CAK')G\_KEK'\}, ICV_{G\_ICK'}$ and from subgroup Gb members are $\{MN_{Ga}, MN_{KS}, (MN_q \| G\_CAK')G\_KEK'\}, ICV_{G\_ICK'}$, where p and q are the sender in each subgroups. The key server and other group members can verify that all the members obtain the correct G_CAK'.

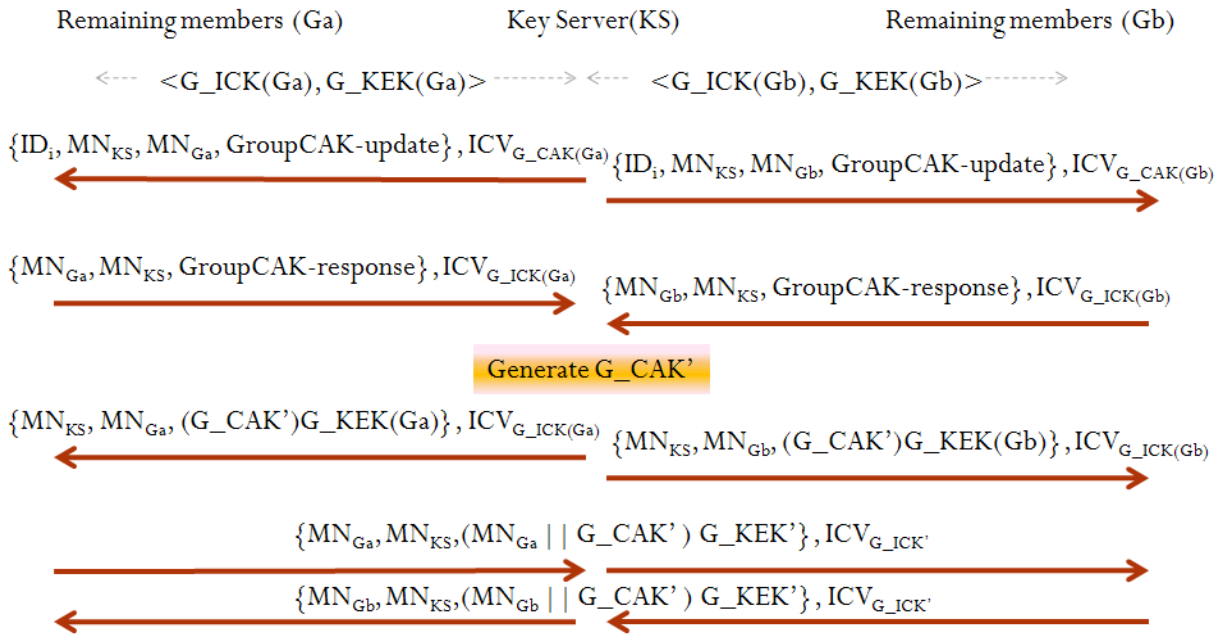


Figure 4. 7 Group CAK distribution for leave

4.5.3 Tree Balance

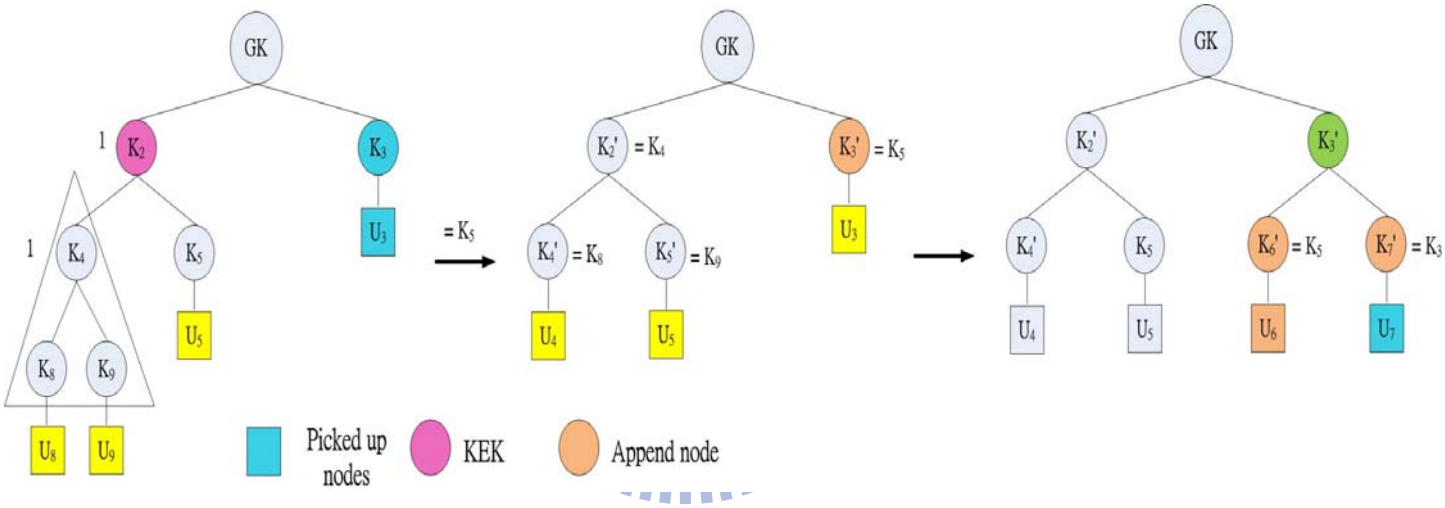
The key tree has to be balance since the height of the tree is more than $\lceil \log_2 N \rceil$. Generally, tree balance takes place as the number of the group members is 2 to the power of n. Consequently, the key tree will be a full tree after balanced. The following is the procedure of balance operation:

1. Key server finds the full subtrees which the leaves are in the $\lceil \log_2 N \rceil + 1$ level, where the set of subtrees belongs to the left subtree of the root node is S_L and the other is S_R. The nodes from the root of subtrees to the children of the key tree root are marked as record nodes.
2. Key server records the number of users in the sibling subtrees of the record nodes. The value is set to infinity if the subtree has no record nodes at the level. And then key server sum up the number of users at each level and find the smallest value in S_L and S_R respectively. If more than one value has the smallest value, then key server chooses the nodes with the smallest indices in each S_L and S_R respectively.
3. The user nodes at the sibling subtrees with the smallest value are marked as picked-up nodes. Key server replaces the siblings of the subtrees' root key to their parent nodes.
4. The sibling key nodes of the nodes in the ancestor node set are called KEK nodes, except the nodes with picked-up nodes in their subtrees.
5. Key server generates a random number R encrypted with KEK nodes and sends out to the members except pick-up nodes.
6. On the receipt of random number R, the members compute the new auxiliary keys, which are the keys in the path from picked-up nodes to the tree root, as Ki'=OF($K_i \oplus$R, "Auxiliary key index i").
7. Key server finds the nodes with no child and the indices are smaller than N, and the pick-up nodes appended to the found nodes and update their index. And then key server sends the auxiliary keys to the pick-up nodes.

The balance procedure is without updating group key. Therefore, the communication service would not be terminated during tree balance operation.

The example of key tree for balance is shown in Figure 4.7. The height of the tree is 4

which exceed the maximum value $\lceil \log_2 4 \rceil$. The full subtree A is found. Key server records the number of users in sibling subtrees of the nodes in the path from subtree A's ancestor. Subtree K3 and subtree K5 are both with one user. Key server thus choices the user nodes of K3's descents as picked-up node, shown as U_3 in Figure 4.8(a). And then key server use K_2 as the encryption key to protect the random number generated by key server. The K2 is removed and the indices of K4, K5, K8, K9 are update to the floor of half of their original indices as shown in Figure 4.8(b). At last, pick-up node U_7 is appended to K_3. Tree balance is accomplished.



(a) (b) (c)

Figure 4. 8 The topology of key tree balance

4.6 The Applications of Communication

In this section, we divide into three parts: station-to-station communication in LAN, station-to-station communication in VLAN and group communication.

First, station communication in LAN is shown as Figure 4.9. Since station A desires to communicate with station B in the same network, the STAkey handshake is required if there is no session key shared between station A and B. After that, station A, station B and internal switch(es) obtain the $STAKey_{A,B}$ which is used for message protection. The message is

encrypted by station A and concatenated with integrity check value (ICV).On the receipt of message, the internal switch search $STAKey_{A,B}$ in the local storage based on source MAC address. The ICV is verified by the switch using $STAKey_{A,B}$. If the verification is valid, the message is sent out without modified. Otherwise, the switch drops the message. After receiving the message, station B decrypt and verify the message using $STAKey_{A,B}$.

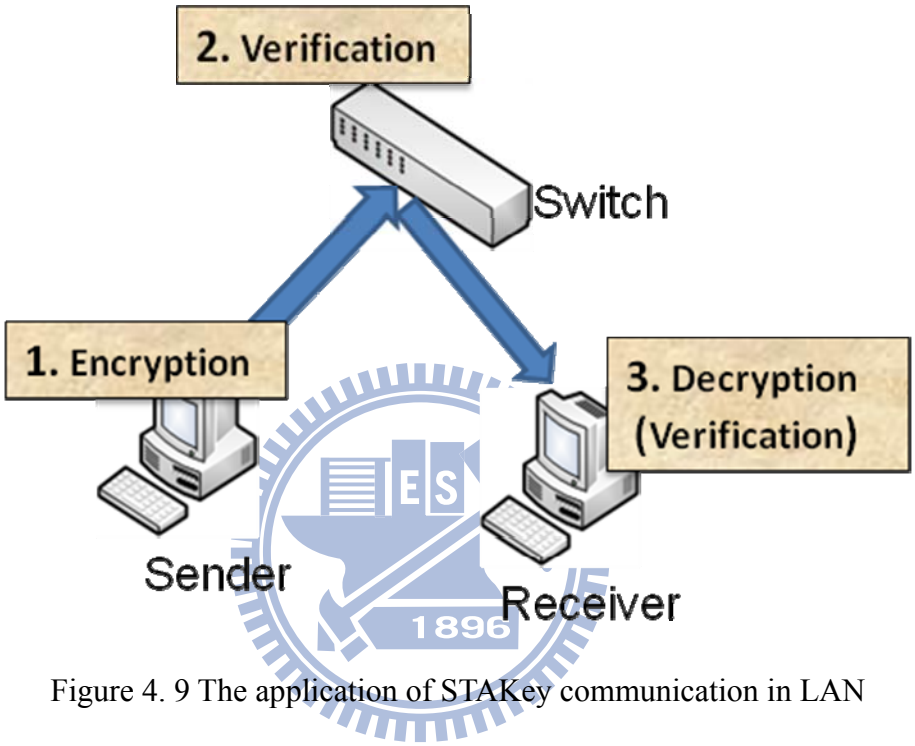


Figure 4. 9 The application of STAKey communication in LAN

Second, station communication in virtual-LAN is shown in figure 4.10. The sender and the receiver are acted as the same in station communication in LAN. The 802.1Q tunnel is between two connected switches. While the data frame is transmitted from sender and receiver, which are in the same VLAN, the switch A has to insert 802.1Q tag and encrypt it with the receipt message. And then switch B decrypts the message, verifies the message is sent from switch A. Switch B takes off 802.1Q tag and sends out the encrypted message protected by STAKey to receiver. After receiving the message, receiver utilizes the STAKey to decrypt and verify the message.

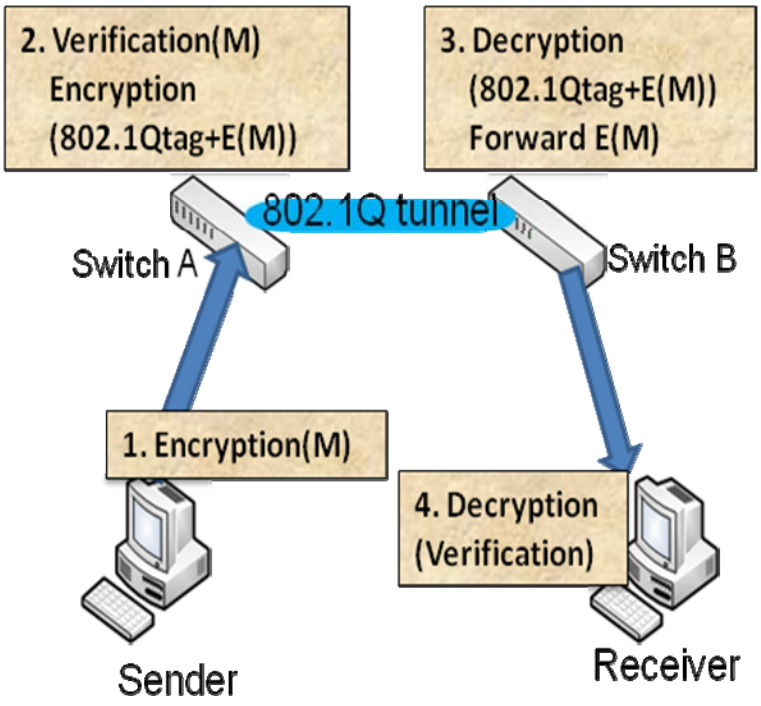


Figure 4. 10 The application of STAKey communication in VLAN

Third, the broadcast frames in local area network should be protected by the group session key that every station in the same network can access the data. After the member joins the group, they obtain the latest group SAK. The sender encrypts the frames by group SAK, and then the internal switch verifies the integrity and the authenticity of the message. If the verification is valid, the switch forwards the frames without decrypt the message. Until the frames arrived, they are decrypted and verified by the receivers.

Chapter 5

Security Analysis

In this section, we analyze the security requirements of the proposed protocol. Besides, SVO logic [16] is one of the most complete formal semantic analysis systems now widely used by researchers and designers. To confirm the correctness of the proposed protocols, we use SVO logic to prove STAKey handshake protocol and group CAK distribution protocol. For STAKey handshake protocol, we derive that the session key is shared only between two negotiated stations. For group CAK distribution protocol, we derive that the new join station obtains the new group CAK from key server and the remaining stations shared the group CAK mutually in join and leave operations.

5.1 Analysis of security requirements

The security of the proposed protocols is analyzed with some well known attacks and requirements.

Key secrecy – The delivered keys are generated independently by strong random number generator and distributed with a secure channel. The attacker has to utilize burst force attack to guess $\Omega(2^m)$ times with overwhelming probability, where m is the bit-length of the key. Besides, all the auxiliary keys are computed by one-way function with new group key. The one-way function is based on the property of hash function that is irreversible. The probability is negligible under the random oracle model [18], since the attacker tries to guess the correct keys.

Forward secrecy – Forward secrecy assures that a departing or expelled member knows a series of previous group keys cannot identify subsequent group keys. Suppose that now time period is T_i and a departing member left at T_j, where $j < i$. The departing member owns GK_j

but has not corresponding auxiliary key to decrypt GK_{j+1}. Besides, it cannot compute next auxiliary key without GK_{j+1}. Thus, the departing member is unable to obtain any keys in time period T_i. Forward secrecy is ensured.

Backward secrecy –Backward secrecy ensures that an adversary that knows a subset of group keys cannot discover previous group keys. Suppose that a new member joins the group at time T_j. It has no group key and any auxiliary keys at time T_{j-1}. Thus the new member is unable to guess any previous keys.

Known key security – Known key security refers that if the session key is compromised, it should have no impact on other session keys. Each session key SAK_i is generated by strong random number generator. There is no relationship among session keys. The session key SAK_i is unable to derive previous or future session keys.

Replay attack resistance – Our protocols uses message number to prevent the replay attacks. Since the message numbers of each members are generated independently, attacks by replaying messages of previous sessions will be failed.

Active attack resistance –This attack requires the attacker to be able to transmit message to one or both of the parties, or block the data stream in one or both directions. The attacker has no way to modify the message and compute the correct integrity check value without CAK or SAK. Thus active attack resistance is ensured.

5.2 SVO logic

This section begins with the introduction to the basic inference rules and axioms. And then we use SVO logic to transform our protocol in to the idealized form and have prerequisite hypothesis and security objectives of the protocol. Finally, we analyze the goals based on formal semantic logic. $(X \supset Y)$ denotes that the current knowledge of X can be

demonstrated to Y. $SK_{A,B}$ denotes the secret key shared between A and B. $K_{A,B}$ denotes the session key shared between A and B.

5.2.1 Inference rules and axioms

Two inference rules:

- R1 – Modus Ponens: From φ and $(\varphi \supset \psi)$ infer ψ.
- R2 – Necessitation: From $\vdash \varphi$ infer $\vdash$P believes φ.

Axioms:

- Believing

  I1 – P believes $\varphi \wedge$ P believes $(\varphi \supset \psi) \supset$ P believes ψ

- Source Association

  I2 – $(P \xleftrightarrow{K} Q \wedge$ R received $\{X^{Q}\}_{K}) \supset$ (Q said X $\wedge$ Q has X)

- Receiving

  I3 – P received $(X_1, \ldots, X_n) \supset$ P received X_i

  I4 – (P received $\{X\}_K \wedge$ P has K) $\supset$ P received X

- Seeing

  I5 – P received $X \supset$ P sees X

  I6 – P sees $(X_1, \ldots, X_n) \supset$ P sees X_i

- Comprehending

  I7 – (P received $F(X) \wedge$ P believes P sees X) $\supset$ P believes P received $F(X)$

- Saying

  I8 – P says $(X_1, \ldots, X_n) \supset$ P says X_i

- Jurisdiction

  I9 – (P controls $\varphi \wedge$ P says $\varphi) \supset \varphi$

- Freshness

  I10 – fresh(X_i) $\supset$ fresh($X_1, \ldots, X_n$)

- Nonce-verification

  I11 – (fresh(X) $\wedge$ P said X) $\supset$ P says X

- Symmetric goodness of shared keys

  I12 – $P \xleftrightarrow{K} Q \equiv Q \xleftrightarrow{K} P$

- Having

  I13 – P has K $\equiv$ P sees K

5.2.2 Idealization, hypothesis and security objectives of station key handshake protocol

Idealization:

Take three parties, A, B, and KS into consideration, the protocol is transformed into an idealized form as follows:

- S1 – $A \rightarrow KS : \{MN_A\}$
- S2 – $KS \rightarrow B : \{MN_B, MN_{KS}, (A \xleftrightarrow{K_{A,B}} B)K_{KS,B}\}$
- S3 – $KS \rightarrow A : \{MN_A, MN_{KS}, (A \xleftrightarrow{K_{A,B}} B)K_{KS,A}\}$
- S4 – $B \rightarrow A : \{MN_B, MN_A, (MN_B, MN_A, A \xleftrightarrow{K_{A,B}} B)K_{A,B}\}$
- S5 – $A \rightarrow B : \{MN_B, (MN_B, A \xleftrightarrow{K_{A,B}} B)K_{A,B}\}$

Hypothesis:

We have prerequisite hypothesis of the STAKey handshake protocol as follows:

- A1 –
    1. A believes $A \xleftrightarrow{SK_{KS,A}} KS \wedge$ A has $SK_{KS,A}$
    2. B believes $B \xleftrightarrow{SK_{KS,B}} KS \wedge$ B has $SK_{KS,B}$
    3. KS believes $A \xleftrightarrow{SK_{KS,A}} KS$
    4. KS believes $B \xleftrightarrow{SK_{KS,B}} KS$
- A2 –
    1. A believes fresh (MN_A)
    2. B believes fresh (MN_B)
    3. KS believes fresh (MN_{KS})
- A3 –
    1. KS believes fresh ($A \xleftrightarrow{K_{A,B}} B$)
- A4 –
    1. A believes KS controls $A \xleftrightarrow{K_{A,B}} B$
    2. B believes KS controls $A \xleftrightarrow{K_{A,B}} B$
- A5 –
    1. A received $\{MN_A, MN_{KS}, (A \xleftrightarrow{K_{A,B}} B)SK_{KS,A}\}$
    2. A received $\{MN_B, MN_A, (MN_B, MN_A, \text{B has } K_{A,B})K_{A,B}\}$
- A6 –
    1. B received $\{MN_A, MN_{KS}, (A \xleftrightarrow{K_{A,B}} B)SK_{KS,B}\}$
    2. B received $\{MN_B, (MN_B, \text{A has } K_{A,B})K_{A,B}\}$

Goal:

The target protocol is idealized as follows:

- G1 – A believes ($A \xleftrightarrow{K_{A,B}} B \wedge$ A has $K_{A,B}$)
- G2 – B believes ($A \xleftrightarrow{K_{A,B}} B \wedge$ B has $K_{A,B}$)
- G3 – A believes (($A \xleftrightarrow{K_{A,B}} B \wedge$ A has $K_{A,B}$) $\wedge$ B says(B has $K_{A,B}$))
- G4 – B believes (($A \xleftrightarrow{K_{A,B}} B \wedge$ B has $K_{A,B}$) $\wedge$ A says(A has $K_{A,B}$))

5.2.3 Proof based on formal semantic logic of station key handshake protocol

Lemma 1. The proposed STAKey handshake protocol provides secure key establishment, i.e. Goals G1 A believes ($A \xleftrightarrow{K_{A,B}} B \wedge$ A has $K_{A,B}$) **and G2** B believes ($A \xleftrightarrow{K_{A,B}} B \wedge$ B has $K_{A,B}$) **are achieved.**

Proof:

By I3 and A5.1, we apply receiving rule to derive

A received $(A \xleftrightarrow{K_{A,B}} B)SK_{KS,B}$ (statement 1)

By I4, A1.1 and (statement 1), we apply comprehending rule to derive

A received $A \xleftrightarrow{K_{A,B}} B$ (statement 2)

By I5, (statement 2) , R1and I13, we apply seeing and having rules to derive

A sees $A \xleftrightarrow{K_{A,B}} B$

A believes (A has $K_{A,B}$) (statement 3)

By I2, A1.1 and (statement 1), we apply source association rule to derive

KS said $A \xleftrightarrow{K_{A,B}} B$ (statement 4)

By I11, A3.1, A2.3 and (statement 4), we apply nonce-verification rule to derive

KS says $A \xleftrightarrow{K_{A,B}} B$ (statement 5)

By R1 and (statement 5), we derive

A believes KS says $A \xleftrightarrow{K_{A,B}} B$ (statement 6)

By I9 and (statement 6), we apply jurisdiction rule to derive

A believes $A \xleftrightarrow{K_{A,B}} B$ (statement 7)

By (statement 3) and (statement 7), we derive A believes $A \xleftrightarrow{K_{A,B}} B$ and A believes (A has $K_{A,B}$), which Goal G1 is achieved.

Goal G2 is same as proof of G1. Thus, the goals G1 and G2 are achieved that we know $K_{A,B}$ is good key for A and B.

Lemma 2. The proposed STAKey handshake protocol provides key confirmation, i.e. Goal G3 and G4 are achieved.

Proof:

By A5.2 and I3, we apply receiving rule to derive

A received $(MN_B, MN_A, \text{B has } K_{A,B})K_{A,B}$ (statement 8)

By Lemma 1. A believes $A \xleftrightarrow{K_{A,B}} B$, I2 and (statement 8), we apply source association rule to derive

B said $(MN_B, MN_A, \text{B has } K_{A,B})$ (statement 9)

By I11, A2.2 and (statement 9), we apply nonce-verification rule to derive

B says $(MN_B, MN_A, \text{B has } K_{A,B})$ (statement 10)

By I8 and (statement 10), we apply saying rule to derive

B says (B has $K_{A,B}$) (statement 11)

By R1 and (statement 11), we derive

A believes B says (B has $K_{A,B}$) (statement 12)

By Lemma 1. and (statement 12), we derive A believes (($A \xleftrightarrow{K_{A,B}} B \wedge$ A has $K_{A,B}$) $\wedge$ B says(B has $K_{A,B}$)), which Goal G3 is achieved.

Goal G4 is same as proof of G2. Thus, the goals G3 and G4 are achieved that we know $K_{A,B}$ is the session key shared with only A and B.

5.2.4 Idealization, hypothesis and security objectives of group key distribution protocol

In this section, there are three parts in the proof: A, B and KS. The proof of each pair of the members obtains the good group key is the same as A and B pair achieves the goals.

Idealization:

Take three parties, A, B, and KS into consideration, the protocol is transformed into an idealized form as follows:

- S1 – $KS \rightarrow A : \{MN_A, MN_{KS}, MN_B, (A \xleftrightarrow{K_G'} B)K_{G1}\}$
- S2 – $KS \rightarrow B : \{MN_B, MN_{KS}, MN_A, (A \xleftrightarrow{K_G'} B)K_{G2}\}$
- S3 – $A \rightarrow B : \{MN_A, MN_B, (MN_A, A \xleftrightarrow{K_G'} B)K_G'\}$
- S4 – $B \rightarrow A : \{MN_B, MN_A, (MN_B, A \xleftrightarrow{K_G'} B)K_G'\}$

Hypothesis:

- A7 –
  1. A believes $A \xleftrightarrow{K_{G1}} KS \wedge$ A has K_{G1}
  2. B believes $B \xleftrightarrow{K_{G2}} KS \wedge$ B has K_{G2}
  3. KS believes $A \xleftrightarrow{K_{G1}} KS$

4. KS believes $A \xleftrightarrow{K_{G2}} KS$

- A8 –
  1. A believes fresh (MN_A)
  2. B believes fresh (MN_B)
  3. KS believes fresh (MN_{KS})
- A9 –
  1. KS believes fresh ($A \xleftrightarrow{K_G'} B$)
- A10 –
  1. A believes KS control $A \xleftrightarrow{K_G'} B$
  2. B believes KS control $A \xleftrightarrow{K_G} B$
- A11 –
  1. A received $\{MN_A, MN_{KS}, MN_B, (A \xleftrightarrow{K_G'} B)K_{G1}\}$
  2. A received $\{MN_B, MN_A, (MN_B, \text{B has } K_G')K_G'\}$
- A12 –
  1. B received $\{MN_B, MN_A, (A \xleftrightarrow{K_G'} B)K_{G2}\}$
  2. B received $\{MN_A, MN_B, (MN_A, \text{A has } K_G')K_G'\}$

Goal:

The target protocol is idealized as follows:

- G5 – A believes ($A \xleftrightarrow{K_G'} B \wedge \text{A has } K_G'$)
- G6 – B believes ($A \xleftrightarrow{K_G'} B \wedge \text{B has } K_G'$)
- G7 – A believes (($A \xleftrightarrow{K_G'} B \wedge \text{A has } K_G'$) $\wedge$ B says(B has K_G'))
- G8 – B believes (($A \xleftrightarrow{K_G'} B \wedge \text{B has } K_G'$) $\wedge$ A says(A has K_G'))

5.2.5 Proof based on formal semantic logic of group key distribution protocol

Lemma 3. The proposed group key distribution protocol provides secure key establishment, i.e. Goals G5 A believes ($A \xleftrightarrow{K_G'} B \wedge$ A has K_G') **and G6** B believes ($A \xleftrightarrow{K_G'} B \wedge$ B has K_G') **are achieved.**

Proof:

By I3 and A11.1, we apply receiving rule to derive

A received $(A \xleftrightarrow{K_G'} B)K_{G1}$ (statement 13)

By I4, A1.1 and (statement 13), we apply comprehending rule to derive

A received $A \xleftrightarrow{K_G'} B$ (statement 14)

By I5, (statement 14) , R1and I13, we apply seeing and having rules to derive

A sees $A \xleftrightarrow{K_G'} B$

A believes (A has K_G') (statement 15)

By I2, A1.1 and (statement 13), we apply source association rule to derive

KS said $A \xleftrightarrow{K_G'} B$ (statement 16)

By I11, A9.1 and (statement 16), we apply nonce-verification rule to derive

KS says $A \xleftrightarrow{K_G'} B$ (statement 17)

By R1 and (statement 17), we derive

A believes KS says $A \xleftrightarrow{K_G'} B$ (statement 18)

By I9 and (statement 18), we apply jurisdiction rule to derive

A believes $A \xleftrightarrow{K_G'} B$ (statement 19)

By (statement 15) and (statement 19), we derive A believes $A \xleftrightarrow{K_G'} B$ and A believes (A

has K_G'), which Goal G5 is achieved.

Goal G6 is same as proof of G5. Thus, the goals G5 and G6 are achieved that we know K_G' is good key for A and B.

Lemma 4. The proposed group key distribution protocol provides key confirmation, i.e. Goal G5 and G6 are achieved.

Proof:

By A11.2 and I3, we apply receiving rule to derive

A received $\{MN_B, MN_A, (MN_B, \text{B has } K_G')K_G'\}$ (statement 20)

By Lemma 1. A believes $A \xleftrightarrow{K_G'} B$, I2 and (statement 20), we apply source association rule to derive

B said $(MN_B, \text{B has } K_G')$ (statement 21)

By I11 and (statement 21), we apply nonce-verification rule to derive

B says $(MN_B, \text{B has } K_G')$ (statement 22)

By I8 and (statement 22), we apply saying rule to derive

B says (B has K_G') (statement 23)

By R1 and (statement 23), we derive

A believes B says (B has K_G') (statement 24)

By Lemma 3. and (statement 24), we derive A believes (($A \xleftrightarrow{K_G'} B \wedge$ A has K_G') $\wedge$ B says(B has K_G')), which Goal G7 is achieved.

Goal G8 is same as proof of G7. Thus, the goals G7 and G8 are achieved that we know K_G' is the group session key shared with A and B.

Chapter 6

Performance Analysis

In this chapter, both the computation and communication cost of NAC will be evaluated. As the results showed, NAC incurs the lowest cost compared with related work.

6.1 Communication Cost of Group key distribution

In this section, we will compare our proposed scheme with other schemes that we mention previously. The comparison divides into two parts: communication cost, and computation cost.

	Communication Cost			
	Join		Leave	Tree Balance
	Unicast	Multicast	Multicast	
LKH	$\log_2 N+1$	$\log_2 N$	$2\times\log_d N$	No
OFT	$\log_2 N+1$	$\log_2 N$	$\log_2 N$	No
Proposed	$\log_2 N+1$	1	$\log_2 N$	Yes

Table 6. 1 Comparison of communication cost

The comparison of communication cost is shown in Table 6.1. As shown in this table, we list the number of messages which are sent by key server. As the table, our proposed scheme is the best result. For each join operation, key servers send at least $2\log_2 N$ messages in LKH and OFT schemes but we reduce to only need one message for multicast. For each leave operation, our proposed scheme is also the best result. Table 6.1 shows that LKH and OFT do

not provide the schemes for tree balance.

6.2 Computation Cost of Group key distribution

The comparison of computation cost is shown in Table 6.2 and Table 6.2. The computation operations contain AES-GCM encryption (T_{GCM}), verification message integrity (T_{ICV}), key verification (T_{KV}), and one-way function for auxiliary key computation (T_{OF}). The values are included key server and stations computing the integrity check for each round-trip message, encrypting the delivered keys, verifying other members obtain the correct keys, and computing the auxiliary keys in OFT and our scheme. The delivered key is able to be appended in a key distribution message. The delivered auxiliary keys to the same station thus needs only one message. The computation time is improved since the number of group member raises. Key server is the bottleneck of the central group key management architecture.

A famous speed benchmarks for some of the most commonly used cryptographic algorithms are specified in [15], which run on an Intel Core 2 1.83 GHz processor under Windows in 32-bit mode. A 128-bit AES-GCM operation takes about 0.15 microseconds, a 1000-byte AES-GCM operation takes about 12.35 microseconds, and a 128-bit SHA-256 takes about 0.577 microseconds on common stations.

Intel IXP 465 network processor [20] integrates hardware acceleration of popular cryptography algorithms for protected applications. The performance of this network process can support encryption and decryption rates of up to 70Mbps for AES algorithm. Consequently, a 128-bit AES-GCM operation takes about 1.83 microseconds, a 1000-byte AES-GCM operation takes about 171 microseconds, and a 128-bit SHA-256 takes about 3.65 microseconds.

		Computation Cost Join		
		Key server	New Member	Old member
LKH	T_{GCM}	3+$\log_2$N	2	$\log_2$N+1
	T_{ICV}	3N+$\log_2$N+3	N+1	2N+ $\log_2$N+2
	T_{KV}	3N	N-1	2N-2
OFT	T_{GCM}	$\log_2$N+3	2	2
	T_{ICV}	3N+$\log_2$N+3	N+1	3N/2+4
	T_{KV}	2N+1	N-1	3/2N
	T_{OF}	$\log_2$N	-	$\log_2$N
Proposed	T_{GCM}	4	2	2
	T_{ICV}	2N+6	N+1	N+5
	T_{KV}	N+2	N-1	N+1
	T_{OF}	$\log_2$N-1	-	$\log_2$N-1

Table 6. 2 Comparison of computation cost for join operation

		Computation Cost Leave	
		Key server	Old member
LKH	T_{GCM}	2 $\log_2$N	$\log_2$N
	T_{ICV}	3N+3 $\log_2$N-1	N+2$\log_2$N+1
	T_{KV}	2N-1	2N-1
OFT	T_{GCM}	$\log_2$N	1
	T_{ICV}	2 $\log_2$N+2N	N+3
	T_{KV}	N	N-1
	T_{OF}	$\log_2$N	$\log_2$N
Proposed	T_{GCM}	$\log_2$N	1
	T_{ICV}	2 $\log_2$N+2N	N+3
	T_{KV}	N	N-1
	T_{OF}	$\log_2$N-2	$\log_2$N-2

Table 6. 3 Comparison of computation cost for leave operation

The realistic computation time is plugged into the values in Table 6.2 and Table 6.3. For key server, T_{GCM} is 1.83us, T_{ICV} is 114us, T_{KV} is 1.83us, and T_{OF} is 3.65us. For group members (stations), T_{GCM} is 3.096us, T_{ICV} is 12.35us, T_{KV} is 3.096us, and T_{OF} is 0.577us. The results with the number of group members as the variables are shown in Figure 6.1 to Figure 6.4.

The join operation of computation cost for key server and old member are shown in Figure 6.1 and Figure 6.2. Our proposed scheme is better than LKH and OFT due to key server that needs only deliver two multicast messages to the old members and new members. In the worst cast, one of the old members should update $\log_2 N$ auxiliary keys and receive more than N-1 verification messages in LKH, and N/2 in OFT. The cost for old members, therefore, is higher than proposed scheme. In the comparison of key server join operation with LKH and OFT schemes, our scheme improves about 34.5% and 33.9%. Moreover, the computation-costs are improved about 66% and 32.7% compared to our scheme and LKH or OFT for old member join operation.

The leave operation of computation cost for key server and old member are shown in Figure 6.3 and Figure 6.4. LKH has to update much more auxiliary keys and verify that every member obtains the new keys. Thus, the overhead is on the computing the integrity of the messages. The message responded from group members cause the most computation time for key server and members. Our proposed scheme and OFT scheme only send $\log_2 N$ messages to update the key. Besides, our proposed scheme reduces the number of one-way function compared to OFT. In the comparison of key server leave operation with LKH and OFT schemes, our scheme improves about 33.5% and 1.5%. Moreover, the computation-costs are improved about 8.7% and 4.5% compared to our scheme and LKH or OFT for old member leave operation.

Compare to LKH and OFT, our proposed scheme has the least computation overhead in these schemes.

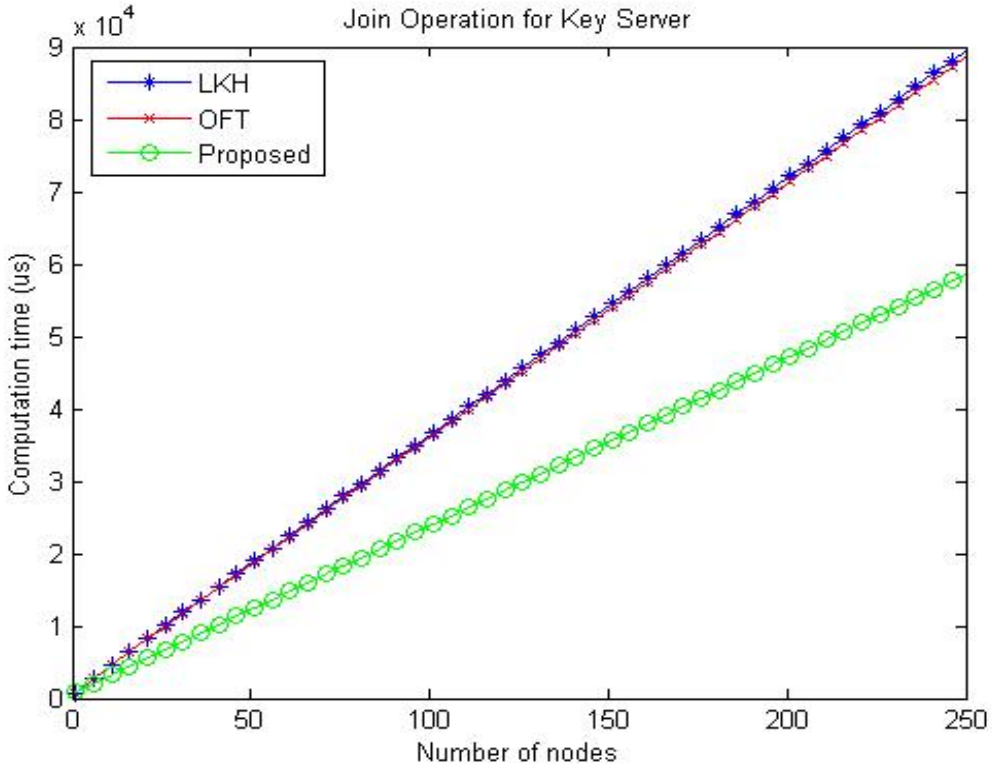


Figure 6. 1 Comparison of computation time with join operation for key server

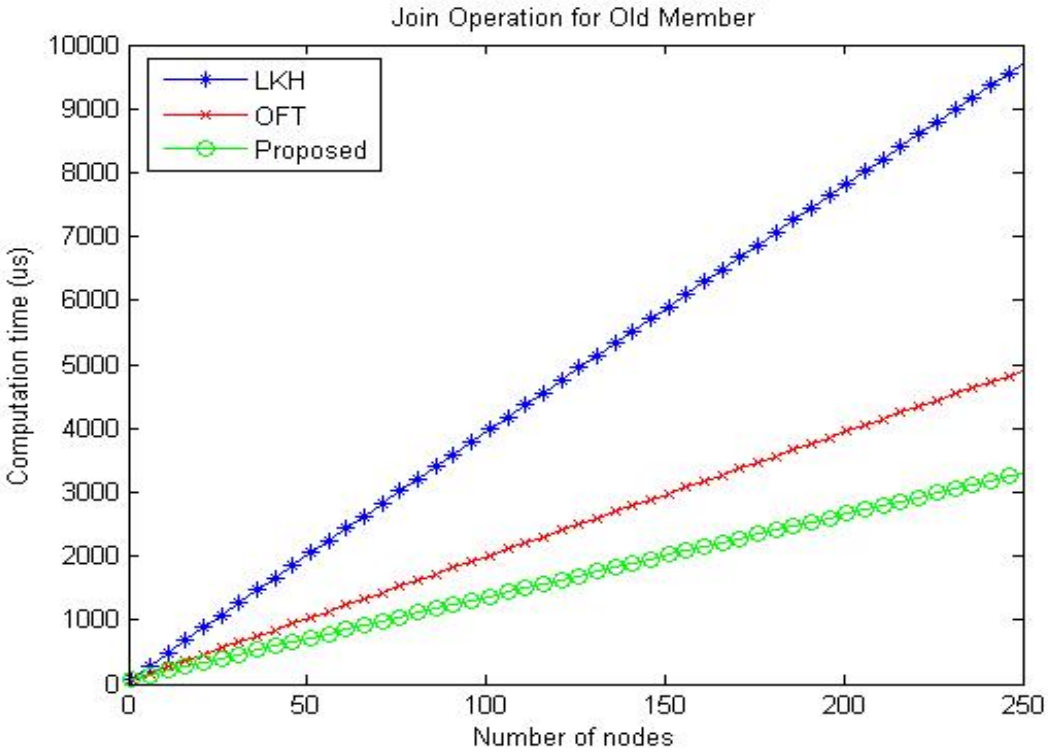


Figure 6. 2 Comparison of computation time with join operation for old member

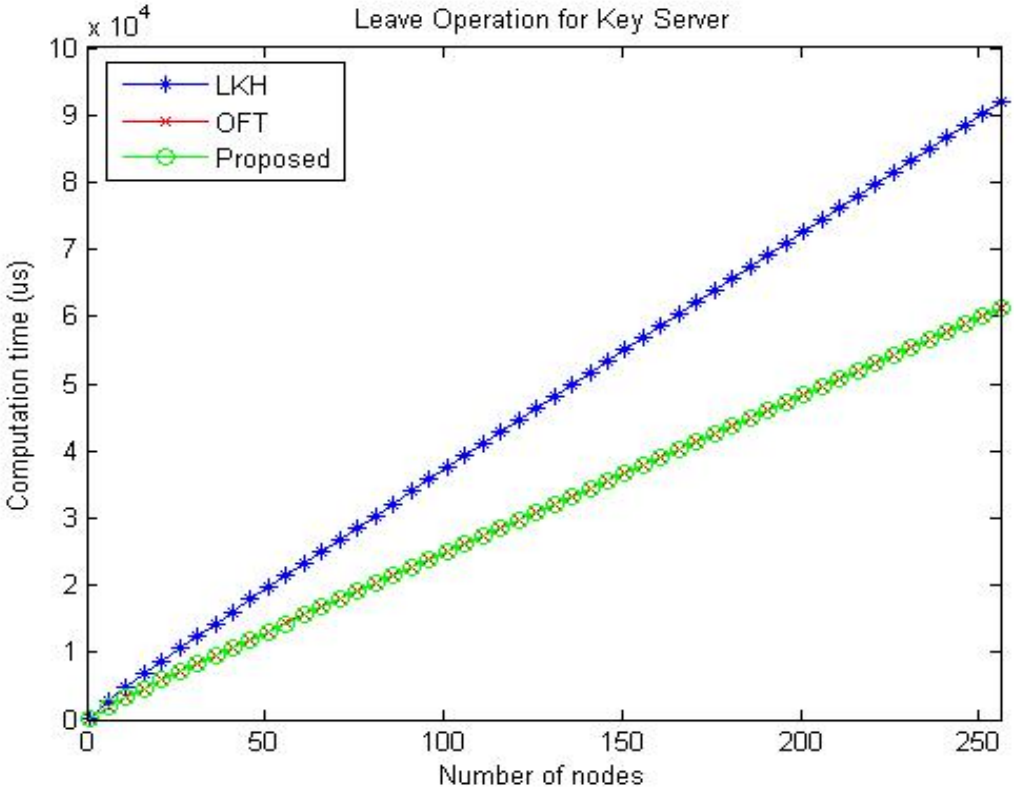


Figure 6. 3 Comparison of computation time with leave operation for key server

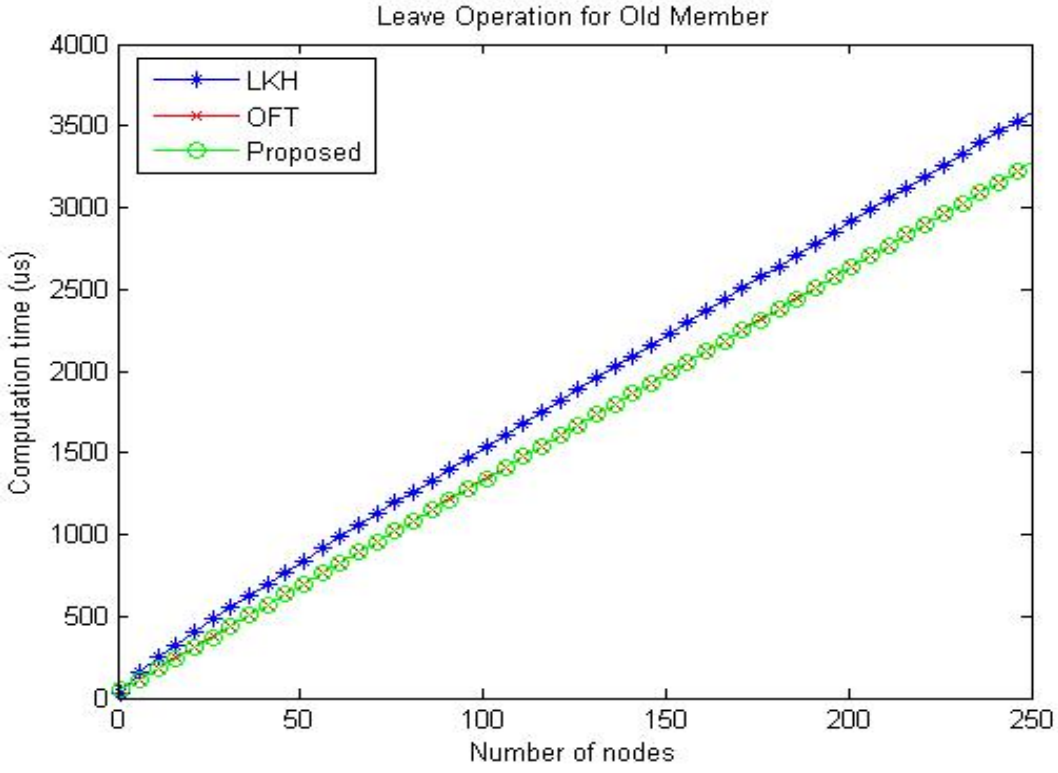


Figure 6. 4 Comparison of computation time with leave operation for old member

6.3 Transmission Time of Data Forwarding

In this section, we compare original MACsec and Linksec with our proposed architecture. The data frames must be decrypted and re-encrypted when they pass through all the internal switches in MACsec and Linksec.

The performance of the proposed station-to-station communication architecture is evaluated by comparing it with MACsec or Linksed. NS2 (Network Simulator) [20] is an excellent and widely used research tool. The evaluation is based on NS2 for background traffic generation to create a local area network environment. Accordingly, the hardware computation time for an AES operation on Cisco Catalyst 4948 supporting Gigabit Ethernet is about 5.5 microseconds [19]. The switch is with a 266MHz CPU. Thus, we insert the encryption processing time into the simulation using NS2.

In the simulation, the environment is the station nodes connected to a switch node with 1Gb bandwidth and 0us delay. The analysis is transmission time of a TCP data flow from one station passing through the switch node to the other station. We calculate the average transmission time is shown in Figure 6.5. The curved lines are no encryption, proposed and MACsec. The result shows that the AES operation is the overhead for data communication. Comparison of MACsec and no encryption, AES operation causes about 50% delay time during the traffic is high. In addition, our proposed scheme has about 12% improvement compare to MACsec. Consequently, the results show that reduces AES operation times is able to alleviate the latency of data flows.

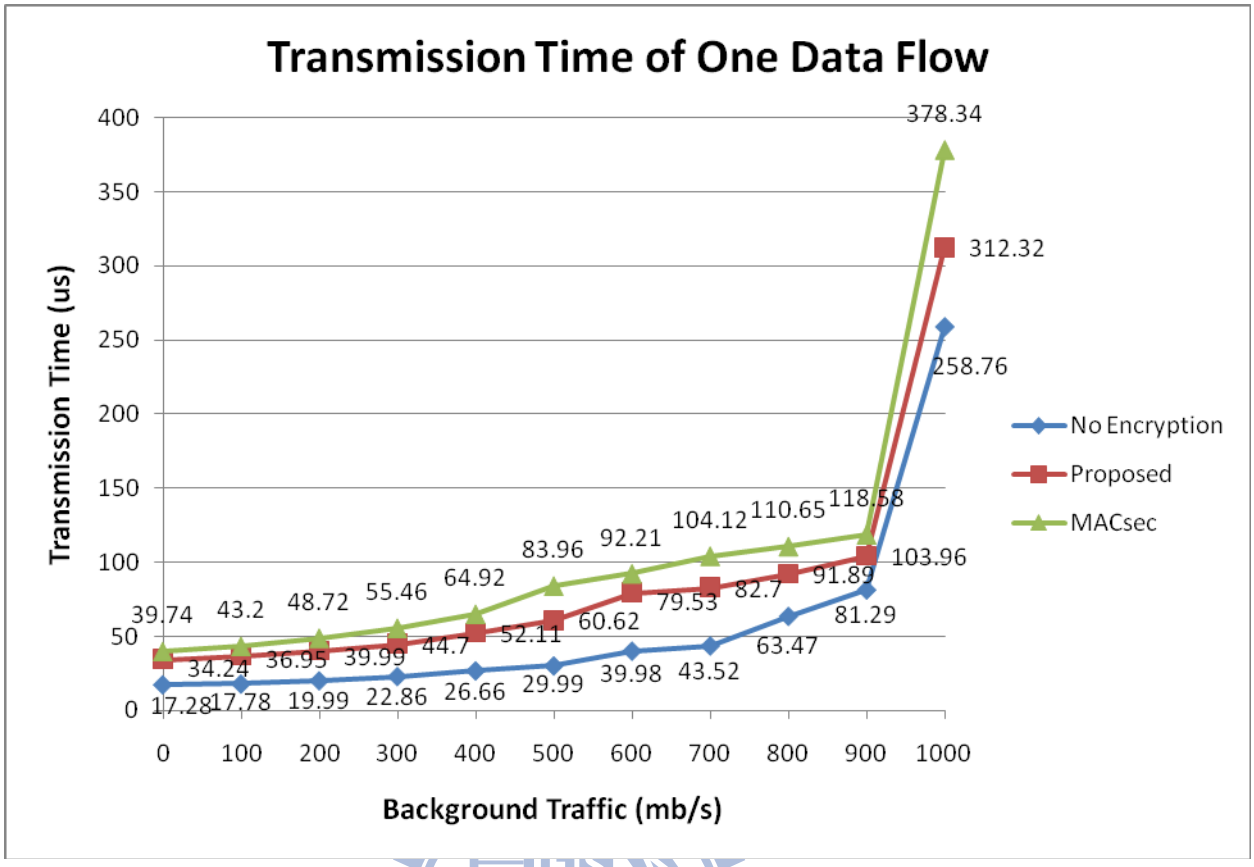


Figure 6. 5 Transmission Time of One Data Flow with AES Encryption

Chapter 7

Conclusion

In this paper, we first introduce a new standard for MAC layer security. In order to enhance and complement the insufficiency of MACsec, we propose a new secure group communication scheme, and provide the procedure of group key distribution protocol and station-to-station key protocol. The station key handshake protocol reduces the overhead of encryption computation on internal devices during the data delivery. There is no key tree scheme addresses tree balance. In our scheme, the tree balance procedure is thus proposed that key server and group members record and distribute the lowest number of keys and messages. In summary, the proposed protocols provide forward and backward secrecy and against replay attack, active attack, and passive attack. We further prove the correctness of the proposed protocols by SVO logic. At last, we compare our group communication scheme to other works. Our proposed scheme enhances computation time for key server and group members during joining and leaving operations. In addition, the computation cost is lessened much more since the number of group members is raised.

Reference

[1] IEEE P802.1AE/D5.1 Draft Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security. Available at http://www.ieee802.org/1/file/private/ae-drafts/d5/802-1ad-d5-1.pdf , 2006.

[2] IEEE 802.1X-REV - Revision of 802.1X-2004 Draft 2.9 - Port Based Network Access Control. Available at http://www.ieee802.org/1/files/private/x-REV-drafts/d2/8 02-1X-rev-d2-9.pdf, 2008.

[3] Prashant Dewan, Larry Swanson, Mem Long, “Are Your Company Secrets Really Safe?” Available at http://www.intel.com.ru/netcomms/casestudies/linksec.pdf, 2006.

[4] H. Harney, C. Muckenhirn, “Group Key Management Protocol (GKMP) Architecture,” Internet Engineering Task Force, Request for Comment 2094, 1997.

[5] Chung Kei Wong, Mohamed Gouda, and Simon S. Lam, “Secure Group Communications Using Key Graphs,” IEEE/ACM Transcations on Networking, Vol. 8, No. 1, February 2000, p. 16-30.

[6] E. Harder, D. M. Wallner, R. C. Agee, "Key Management for Secure Internet Multicast: Issues and Architectures," IETF, RFC 2627, 1999.

[7] D. McGrew, A. Sherman, “Key Establishment in Large Dynamic Groups Using One-way Function Trees,” IEEE Transactions on Software Engineering, May 2003.

[8] Isabella Chang, Robert Engel, Dilip Kandlur, Dimitrios Pendarakis, and Debanjan Saha, “Key Management for Secure Internet Multicast using Boolean Function Minimization Techniques,” IEEE INFOCOM 1999, p. 689-698.

[9] Xiaozhou Steve Li, Yang Richard Yang, Mohamed G. Gouda, and Simon S. Lam, “Batch Rekeying for Secure Group Communications,” ACM SIGCOMM 2001, p. 525-534.

[10] X. Brian Zhang, Simon S. Lam, Dong-Young Lee, and Y. Richard Yang, “Protocol Design for Scalable and Reliable Group Rekeying,” IEEE/ACM Transcations on

Networking, Vol. 11, No. 6, December 2003, p. 908-922.

[11] S. Banerjee, B. Bhattacharjee, “Scalable Secure Group Communication over IP Multicast,” IEEE Journal on Selected Areas in Communications, vol. 20, No. 8, October 2002.

[12] B. Aoba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz, “Extensible Authentication Protocol (EAP),” RFC 3748, 2004.

[13] T. Dierks, E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.1,” RFC 4346, 2006.

[14] J. Song, J. Lee, T. Iwata, “The AES-CMAC-PRF-128 Algorithm for IKE,“ RFC 4615, 2006.

[15] http://www.cryptopp.com/benchmarks.html, 2009.

[16] P. F. Synverson, P. C. van Oorschot, “On unifying some cryptographic protocol logics,” Proceedings of IEEE Computer Society Symposium ’94, pages 14-28, 1994.

[17] NIST, “Security requirements for Cryptographic Modules,” FIPS 140-2, 2001. Available at http://csrc.nist.gov/publications/fips/index.html

[18] R. Canetti, O. Goldreich, S. Halevi, “The random oracle methodology, revisited,” STOC 1998, pages 209-218.

[19] AES-GCM MACsec and FC-SP Cores GCM1/GCM2/GCM3. http://ipcores.com/macsec_802.1ae_gcm_aes_ip_core.htm, 2009.

[20] Intel IXP 425 Network Processor. http://download.intel.com/design/network/ProdBrf/27905105.pdf, 2006.