情報保護學會論文誌
第20卷 第3號, 2010. 6

고속의 이동 IPv6를 위한 보안 프로토콜 연구

유 일 선,[1†‡] 요시아키 호리,[2] 코우이치 사쿠라이[2]
[1]한국성서대학교, [2]규슈대학교

State of Art on Security Protocols for Fast Mobile IPv6

Ilsun You,[1†‡] Yoshiaki Hori,[2] Kouichi Sakurai[2]
[1]Korean Bible University, [2]Kyushu University

요 약

고속의 이동 IPv6 (FMIPv6: Fast Handover for Mobile IPv6) 프로토콜은 2 계층에서 지원 가능한 트리거의 도움으로 핸드오버시 발생하는 과도한 지연시간과 시그날링 메시지를 효과적으로 감소시켰다. 뛰어난 효율성에도 불구하고 FMIPv6는 다양한 공격과 위협에 노출되어 있기 때문에 이를 보호하기 위한 여러 보안 프로토콜이 제안되었다. 본 논문에서는 FMIPv6의 취약점 및 보안요구사항을 정의한 후, 이를 바탕으로 주요 보안 프로토콜의 보안특성을 비교 분석하였다. 분석결과는 본 저자들에 의해 제안되었던 프로토콜이 다른 기법에 비해 과도한 연산을 유발하지 않으며 강력한 보안성을 지니고 있다는 것을 보여 주었다.

ABSTRACT

With the help of various Layer 2 triggers, Fast Handover for Mobile IPv6 (FMIPv6) considerably reduces the latency and the signaling messages incurred by the handover. Obviously, if not secured, the protocol is exposed to various security threats and attacks. In order to protect FMIPv6, several security protocols have been proposed. To our best knowledge, there is lack of analysis and comparison study on them though the security in FMIPv6 is recognized to be important. Motivated by this, we provide an overview of the security protocols for FMIPv6, followed by the comparison analysis on them. Also, the security threats and requirements are outlined before the protocols are explored. The comparison analysis result shows that the protocol presented by You, Sakurai and Hori is more secure than others while not resulting in high computation overhead. Finally, we introduce Proxy MIPv6 and its fast handover enhancements, then emphasizing the need for a proper security mechanism for them as a future work.

Keywords: FMIPv6 Security, SEND, AAA, CGA

I. Introduction

Mobile Internet Protocol version 6 (MIPv6), specified by IETF, is a protocol where nodes can stay reachable regardless of their movements and locations in the IPv6 Internet [1]. In this protocol, each *Mobile Node (MN)* needs to perform movement detection, IP address configuration, and binding update for its handover. However, these operations result in the excessive latency and signaling messages. In order to address the problems, several enhancements such as Fast handover for MIPv6 (FMIPv6) [2], Hierarchical MIPv6 (HMIPv6) [3], and Enhanced Route Opti-

접수일(2010년 4월 1일), 수정일(2010년 4월 17일), 게재확정일(2010년 6월 4일)
†‡ 주저자, 교신저자, isyou@bible.ac.kr

mization for Mobile IPv6 (ERO) [4] have been developed and standardized in the Internet Engineering Task Force (IEFT).

Especially, FMIPv6 achieves to optimize the handover overhead caused by both the movement detection and IP address configuration operations by making the best use of various Layer 2 (L2) triggers. Despite such an optimization, without any security mechanism, it is vulnerable to various attacks such as *Session Hijacking* (*SSH*), *Malicious Mobile Node Flooding* (*MMF*), *Man-In-The-Middle* (*MiTM*) and *Denial of Service* (*DoS*) attacks [5][6]. In order to secure FMIPv6, several security protocols have been presented [7]-[10]. Especially, they try to leverage the existing security approaches such as SEcure Neighbor Discovery (SEND) [11], Cryptographically Generated Addresses (CGA) [12] and the Authentication, Authorization, Accounting (AAA) infrastructure [13]. However, to our best knowledge, there is lack of analysis and comparison on the protocols though the security in FMIPv6 is recognized to be important.

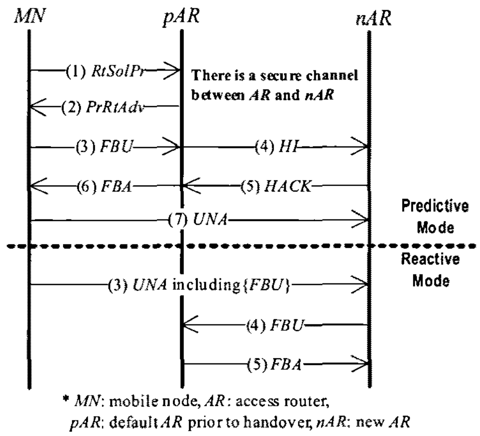
In this paper, we provide a survey on the security protocols for FMIPv6. For this, the security threats and requirements are presented. Based on them, we discuss the advantages and weaknesses of the protocols, which are then compared with each other.

The remainder of the paper is organized as follows. Section 2 introduces FMIPv6, analyzes its security threats and defines the security requirements. In section 3, the security protocols are explained and discussed. In section 4, we compare them, and then describe the related research challenges. Finally, we conclude this paper in section 5.

II. FMIPv6 and Its Security Threats

2.1 FMIPv6 operation

FMIPv6 operates as shown in Fig. 1. When an *MN* detects its movement by using L2 triggers, it sends a *Router Solicitation for Proxy Advertisement* (*RtSolPr*) message to the current *Access Router* (i.e., *pAR*). In response, the *pAR* sends the *MN* a *Proxy Router Advertisement* (*PrRtAdv*) message including the new *AR* (i.e., *nAR*)'s information such as L2 and IP addresses, and prefix. By using the information given in the *PrRtAdv* message, the *MN* configures a new *Care-of Address* (*nCoA*) while still present on the *pAR*'s link. If there is no address conflict, the *MN* can use the *nCoA* instantly after its attachment to the *nAR*'s link. In this way, this protocol minimizes the latency caused by the *nCoA* configuration. Once the *nCoA* is formulated, the *MN* sends a *Fast Binding Update* (*FBU*) message containing this address to the *pAR*. Based on the *FBU* message, the *pAR* believes the association between the *MN*'s current *CoA* (*pCoA*) and *nCoA*. Such a belief triggers the *pAR* to exchange the *Handover Initiate* (*HI*) and *Handover Acknowledge* (*HACK*) messages with the *nAR* to establish a tunnel between the *pCoA* and the *nCoA*. Note that FMIPv6 uses such a tunnel to diminish the binding update latency. In order to protect this tunnel, FMIPv6 assumes that there exists a pre-established security association between the *pAR* and the *nAR*. As a result, the *pAR* starts to act as a temporary *Home Agent* (*HA*) for the *MN* while forwarding the traffic sent to *pCoA* on its link to *nCoA* on the *nAR*'s link. At the same time, it sends a *Fast Binding Acknowledgment* (*FBA*) message to the *MN*. Upon receiving this message, *MN* assumes that data pack-



(Fig. 1) FMIPv6 operation

ets are being forwarded to its new location. As soon as the *MN* handovers to the *nAR*'s link, it announces its attachment by sending an *Unsolicited Neighbor Advertisement* (*UNA*) message to the *nAR*. Subsequently, the *nAR* forwards arriving and buffered packets to the *MN*.

Depending on whether the *FBA* message is received on the *pAR*'s link, FMIPv6 has two operation modes: Predictive mode and Reactive mode. In the predictive mode, the *MN* exchanges the *FBU* and *FBA* messages with the *pAR* prior to its handover. It is clear that this mode allows the *MN* to truly achieve the essential advantage of FMIPv6. The reactive mode is executed when the predictive one is not feasible. That is, this mode, instead of the predictive one, is applied if the *MN* cannot send the *FBU* message or receive the *FBA* message on the *pAR*'s link. In this case, the *MN* encapsulates the *FBU* message in the *UNA* message, which triggers the *nAR* to exchange the *FBU* and *FBA* messages with the *pAR* on behalf of the *MN*.

2.2 Security threats

Despite its good efficiency, FMIPv6 without being protected opens the door to various attacks such as *SSH*, *MMF*, *MiTM* and *DoS* attacks. The *SSH* and *MMF* attacks can be launched by redirecting a victim or malicious *MN*'s traffic. In the *SSH* attack, an intruder deceives an *AR* into redirecting a victim *MN*'s traffic to itself. In the *MMF* attack, a legitimate but malicious *MN* tricks its *pAR* into redirecting its traffic to a victim node or network. Note that the *FBU* message is exploited for these two attacks. On the other hand, the *MiTM* and *DoS* attacks can be activated by modifying or fabricating the *PrRtAdv* and *UNA* messages.

In order to analyze security threats, we divide FMIPv6 into three phases: movement detection, fast binding update and new network attachment phases.

FMIPv6 supposes that a security association is pre-established between neighboring *AR*s. Thus, this paper analyzes FMIPv6 security on the assumption that communications between an *MN*'s *pAR* and *nAR* are protected.

Each phase's security threat is described as follows:

- **Movement detection phase:** During this phase, an *MN* can detect its movement by exchanging the *RtSolPr* and *PrRtAdv* messages with its *pAR*. If the *PrRtAdv* message is not protected, this phase is vulnerable to the *DoS* attack. That is, an intruder can send the *MN* a false *PrRtAdv* message indicating that the node is moving to a target *AR*'s network. If the *MN* trusts this message, it sends an *FBU* message to its *pAR*, which then establishes a tunnel with the target *AR*. As a result, the *MN*'s traffic is redirected to the target *AR*'s network. Note that this attack can be easily launched even in the case that the *FBU* message is secured. If successful, this attack allows the target *AR* to steal the *MN*'s packets. Differently, it can make both the *pAR* and *nAR* worthlessly waste their recourses while establishing a tunnel.

- Fast binding update phase: During this phase, the *FBU* message can be misused to perform the *SSH* or *MMF* attacks. In order to carry out the *SSH* attack, an intruder sends a victim *MN*'s *pAR* a fake *FBU* message, which indicates that the *MN* is about to move to its address. As a result, the *pAR* redirects the victim node's traffic to the intruder's address. In this way, the intruder hijacks the victim node's session. On the other hand, with the *FBU* message, a malicious but legitimate *MN* can inform its *pAR* that it is about to go to a victim node's address without going (i.e., the *MMF* attack). This attack causes the victim node and its network to suffer from unwanted traffic. Because the *MN* is a legitimate node, this attack is possible even if the *FBU* message is secured.
- New network attachment phase: During this phase, an *MN* announces its attachment via the *UNA* message. If the *UNA* message is not secured, this phase is vulnerable to the *MiTM* and *DoS* attacks. Assume that there is an intruder between a target *MN* and its *nAR*. Upon seeing the *UNA* message from the *MN*, the intruder gets the node's *nCoA* and *Link Layer Address* (*LLA*) from the message. At the same time, while masquerading the *nAR*, it sends the *MN* a false *Neighbor Advertisement Acknowledgment* (*NAACK*) message indicating that the *MN*'s *nCoA* is invalid. As a result, the *MN* performs address configuration or uses another *CoA* included in the message, and then is interrupted. At this point, the intruder can intercept the *MN*'s packets forwarded by the *nAR* through the *MN*'s original address information.

2.3 Security requirements

This subsection provides the security requirements to address the security threats described above.

- **Secure handover key exchange:** Basically, it is necessary that an *MN* and its *AR* establishes a handover key to protect FMIPv6. With the handover key, the *RtSolPr*, *PrRtAdv*, *FBU*, *FBA* and *UNA* messages should be secured to defend against the security threats. For this goal, the existing protocols mainly use the SEND protocol or the AAA infrastructure. The SEND protocol adopts the CGA method to protect signaling messages based on public key cryptography. In the CGA method, an IPv6 address (i.e., CGA) includes the hash value of its owner's public key in the last 64 bits. Thus, the address allows its owner's public key to be verified without any global Public Key Infrastructure (PKI) or Certificate Authority (CA). However, the SEND protocol causes involved entities to suffer from excessive public key operations and *DoS* attacks. On the other hand, the AAA infrastructure can be employed to establish handover keys. Such an approach is reasonable because the AAA infrastructure is widely deployed today for network access authentication. However, it needs involved nodes to establish a handover key through their authentication server, thus resulting in considerable handover latency.
- **Tight bind between an *MN*'s handover**

key and *CoA*: Each handover key should be tightly bound to its owner's *CoA*. Without such a bind, a malicious *MN* can masquerade another nodes by using its valid handover key.

- **Handover key independence:** Even though a handover key is compromised due to some reasons, its previous or successive keys should not be compromised. Note that this requirement can be satisfied just by using public key cryptography or depending on a trusted third party (i.e., authentication server). As mentioned above, this can result in substantial performance degradation. In addition to the above security requirements, the followings need to be considered.
- **FMIPv6-seamless structure:** It is necessary that there are no additional messages caused by employing a security protocol. In other words, the security protocol should exploit the original FMIPv6 messages such as the *RtSolPr*, *PrRtAdv*, *FBU*, *FBA* and *UNA* ones.
- **Efficiency:** It is required that a security protocol does not result in significant handover delay or excessive amount of messages or computations.

III. Existing Security Protocols for FMIPv6

This section describes the existing security protocols for FMIPv6. For this, we use the notations shown in Fig. 2.

$Msg(A, B)$	the message Msg sent from A to B, where A and B are an IPv6 address
$E(K, M)$	a function that encrypts the message M with the given key K, where K can be a secret key or a public key
$S(K, M)$	a function that digitally signs the message M with the private key K
$AR(i)$	the ith access router which the MN visits, and its IPv6 address, where $i > 0$
$CoA(i)$	the ith care-of address of the MN, where $i > 0$
PU_X	the X's public key from which the CGA is derived
PR_X	the X's private key which corresponds to PU_X
$HK(i)$	the ith handover key, where $i > 0$
$CGAP_X$	the parameters used to verify that the X's CGA is derived from PU_X
$H()$	An one way hash function
$HMAC(K, M)$	an HMAC value computed using the secret key K over the message M
$H^n(X)$	the value computed by performing the hash function on X n times
$K(i)$	the ith message protection secret, where $i > 0$
$\|$	concatenation operation
$Left(n, x)$	the left most n bits of x
$Right(n, x)$	the right most n bits of x

〔Fig. 2〕 Notations

3.1 Kempf-Koodli's protocol (KKP)

Since introduced by Kempf and Koodli, this protocol (after KKP) has been adopted as a standard for FMIPv6 (IETF RFC 5269) [7]. KKP leverages the SEND protocol to address the security threats of FMIPv6. Thus, each involved entity (i.e., *MN* or *AR*) in KKP uses a CGA as its source address, and signs signaling messages with the private key corresponding to this CGA. In this way, KKP verifies each entity's address ownership and public key, while using public key cryptography for both the handover key exchange and the message protection without any security infrastructure.

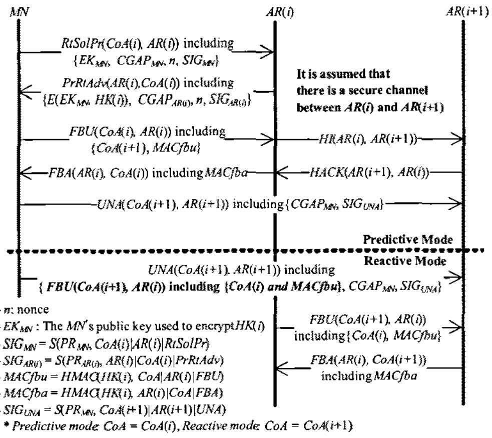
(1) Preliminary

- In order to employ the SEND protocol, each entity Has a public/private key pair (PU_{MN}/PR_{MN} *or* $PUAR_{(i)}/PRAR_{(i)}$) and uses a CGA derived from its public key as its address.
- Each *MN* possesses a handover encryption public/private key pair, which is generated using the same public key algorithm as used for the SEND protocol. Note that the key pàir should be used for only handover negotiation.
- Each *AR* uses *MN*s' CGA to identify their handover key.

(2) Operation

KKP is illustrated in detail in Fig. 3.

Note that the *RtSolPr*, *PrRtAdv*, and *UNA* messages are protected through the SEND protocol. That is, these messages are signed with their sender's private key corresponding to their source address (*i.e.*,



(Fig. 3) Kempf-Koodli's protocol

CGA) while accompanied by the CGA parameters.

This protocol starts when the MN detects its movement through L2 triggers. As the first step, the *MN* sends the *RtSolPr* message including the handover key encryption public key EK_{MN} to the *AR(i)*. According to the SEND protocol, this message is signed with PR_{MN} corresponding to the *MN*'s care of CGA (i.e., *CoA(i)*) and is accompanied by the *MN*'s CGA parameter $CGAP_{MN}$. Upon receiving the message, the AR(i) uses the $CGAP_{MN}$ to verify the PU_{MN}, then validating the SIG_{MN} with the key. If the signature is valid, it generates the handover key *HK(i)*, which is then encrypted with the given EK_{MN}. Afterwards, the *AR(i)* returns the *PrRtAdv* message including the encrypted handover key. The *MN* verifies the message according to the SEND protocol, and decrypts the encrypted EK_{MN} with its handover private key.

Once successfully establishing the handover key *HK(i)*, the *MN* informs the *AR(i)* of its new *CoA* (i.e., *CoA(i+1)*) by using the *FBU* message. This message triggers the *AR(i)* and the *AR(i+1)* to establish a tunnel between the *MN*'s *CoA(i)* and *CoA(i+1)* through the *HI* and *HACK* messages. Then, the *AR(i)* returns the *FBA* message to the *MN*. Note that the *FBU* and *FBA* messages are protected with the *MACfbu* and *MACfba* values computed with the *HK(i)*. If the *FBA* message is valid, the *MN* believes that its data packets are being tunneled to its new *CoA*. As soon as the *MN* moves to the *AR(i+1)*'s link, it informs the *AR(i+1)* of its attachment by using the *UNA* message, which is secured by the SEND protocol. In the reactive mode, the *AR(i+1)* performs the fast binding update phase with the *AR(i)* only if the *UNA* message is valid.

(3) Discussion

With the help of the SEND protocol and the CGA method, KKP achieves the strong handover key exchange based on public key cryptography while protecting the *RtSolPr*, *PrRtAdv*, *FBU*, *FBA* and *UNA* messages. Because each handover key is newly generated and encrypted with the *MN*'s public key EK_{MN}, its compromise doesn't result in a compromise of its previous or successive keys. Furthermore, as the public key EK_{MN} is protected through the CGA method, the *AR(i)* can be sure that the *MN* truly owns the *CoA(i)* if the *MN* shows its knowledge of the handover key. Thus, KKP guarantees both the handover key independence and the tight bind between the *CoA(i)* and the *HK(i)*. Also, KKP has the FMIPv6-seamless architecture by exploiting the existing protocol messages. Thus, it results in no additional messages and *Round Trip Times* (*RTT*).

However, the SEND protocol causes this protocol to suffer from high computation cost. That is, the *MN* and the *AR(i)* should perform at least four asymmetric cryptographic operations for every handover. Such a computation overhead results in a significant burden on mobile devices, which generally tend to have limited computational capabilities and low battery power. Moreover, KKP is vulnerable to the *DoS* attacks because the *AR*s don't perform any check prior to asymmetric cryptographic operations.

3.2 Haddad-Krishnan's protocol (HKP)

In 2006, Haddad and Krishnan proposed a lightweight protocol (after HKP) while largely motivated by the problems of KKP (i.e., heavy cryptographic operations) [8]. This protocol, based on the *One Way Hash Chain* (*OWHC*) method, enables an AR to efficiently exchange a handover key with its *MN* without any public key operation while verifying the association between the *MN*'s handover key and CoA.

(1) Preliminary

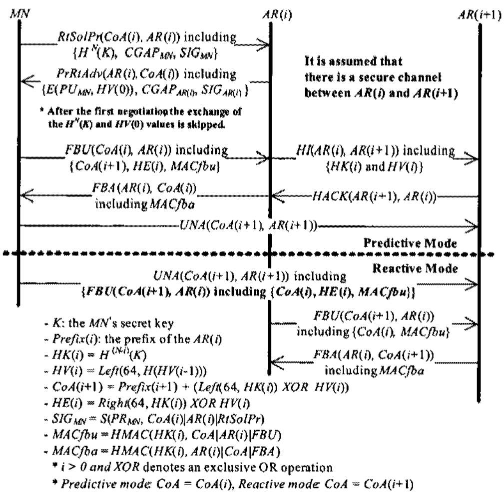
In HKP, the SEND protocol is needed for an *MN* and its *AR* to safely exchange new security parameters (normally at the beginning). Therefore, basically each entity has a public/private key pair (PU_{MN}/PR_{MN} or $PU_{AR(i)}/PR_{AR(i)}$) and, if necessary, uses a CGA derived from its public key as its address.

(2) Operations

Fig. 4 describes HKP in detail.

In order to apply the *OWHC* method, the *MN* should securely share its first hash value $HN(K)$ and first Handover Vector ($HV(0)$) with the $AR(i)$, where the *MN*'s hash chain consists of N hash values and each value's size is 128 bits. For this goal, the SEND secured *RtSolPr* and *PrRtAdv* messages are used when the *MN* makes the first handover or newly updates the security values. Thus, while the $HN(K)$ is protected with the *MN*'s digital signature SIG_{MN}, the $HV(0)$ is encrypted with the *MN*'s CGA public key PU_{MN}. Once the first security values are securely exchanged, the SEND protocol (*i.e.*, public key operations) will not be activated until a time when the security values will need to be newly updated.

For the fast binding update phase, the *MN* firstly computes the handover key $HK(i) = H^{N-i}(K)$ and the handover vector $HV(i) = Left(64, H(HV(i-1)))$. Assume that the $HK(i-1)$ and the $HV(i-1)$ are forwarded by the $AR(i-1)$ during the previous handover or exchanged through the SEND secured *RtSolPr* and *PrRtAdv* messages. Then, it uses the two results to configure the $CoA(i+1)$ according to the equation shown in Fig. 4. At the same time, the hash extension value $HE(i) = Right(64, HK(i))$ XOR $HV(i)$ is calculated and then included in the *FBU* message. In order to be securely delivered to the $AR(i)$, the $HK(i)$ is split into two 64-bit parts, each of which is then exclusive-ORed with the $HV(i)$. On receiving the *FBU* message, the $AR(i)$ firstly computes the $HV(i)$ and uses it to recover the $HK(i)$ from both the $CoA(i+1)$ and the



[Fig. 4] Haddad-Krishnan's protocol

HE(*i*). After that, the *AR*(*i*) verifies the *HK*(*i*) by comparing it with *H*(*HK*(*i*−1)). If this verification is successful, the *AR*(*i*) uses the key to authenticate the *FBU* message. The valid *FBU* message makes the *AR*(*i*) believe the *MN's CoA*(*i*+1) and its association with the *HK*(*i*). Note that the *HK*(*i*) and the *HV*(*i*) are forwarded to the *AR*(*i*+1) through the *HI* message for the next handover. The rest of this protocol is the same as that of *KKP* except that the *UNA* message is not protected.

(3) Discussion

By using the *OWHC* method, HKP protects the *FBU* and *FBA* messages as well as achieves the efficient handover key exchange while minimizing public key operations. Also, this protocol provides the tight bind between the *MN*'s handover key and *CoA* by inserting the first 64-bit part of the handover key to the *CoA*. Furthermore, in HKP, only the *MN* can generate the next handover key, and each generated one is used just once. That is, a compromise of the past keys has no impact on this protocol and, even if a handover key is leaked, its successors cannot be derived. Thus, the handover key independence is guaranteed. In addition, like KKP, HKP results in no additional messages and *RTT*s due to the FMIPv6-seamless architecture.

Despite such a novel approach, HKP suffers from the following problems:

- In HKP, for secure delivery, each handover key is exclusive-ORed with its corresponding handover vector. As every *AR* can use its handover vector to compute the successive ones, it can easily recover the next handover keys in spite of not being able to generate them. That is, even though only one of the previous *AR*s is compromised, its next handover keys can be easily revealed just by eavesdropping.
- Because of focusing on securing the *FBU* and *FBA* messages, this protocol does not protect the *RtSolPr*, *PrRtAdv*, and *UNA* messages. Note that the SEND protocol even cannot be used because the *MN*'s *CoA* is not a CGA.
- HKP needs the SEND protocol when the *MN* executes its first handover or updates its hash chain. Thus, HKP is still vulnerable to *DoS* attacks due to the SEND secured movement detection phase.

3.3 Narayanan et al.'s protocol (NEP)

Narayanan et al. proposed a key management protocol (after NEP), which allows an *MN* and its *AR* to establish their handover key by relying upon the AAA infrastructure [9]. Such an approach is worth noting because the AAA infrastructure is widely deployed for the network access authentication.

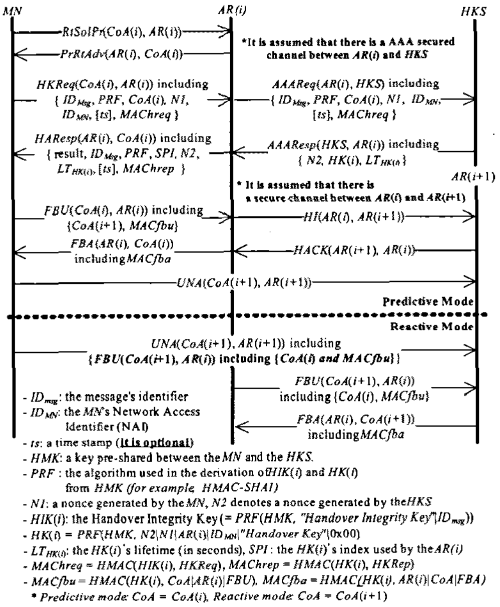
(1) Preliminary

There is the *Handover Key Server* (*HKS*), with which the *MN* shares the *Handover Master Key* (*HMK*) in advance. The *MN* derives the handover key and the *Handover Integrity Key* (*HIK*) from the *HMK*. This paper assumes that the *HKS* collocates with an AAA server in the infrastructure. That is, delivery of handover keys depends on the AAA infrastructure.

(2) Operation

As shown in Fig. 5, the *MN* starts the handover key exchange by sending the *AR*(*i*) the *HKReq* message protected with the *MAChreq*. Note that to compute the *MAChreq*, the *MN* should generate the *HIK*(i) according to the equation given in Fig. 5.

Upon receiving the *HKReq* message, the *AR*(i) forwards it in the form of the *AAAReq* message to the *HKS* with the help of the AAA infrastructure. Given the message, the *HKS* firstly derives the *HIK*(i) from the *HMK*, and then uses it to verify the *MAChreq*. If the verification is successful, the *HKS* is sure that both the *MN* and its message are valid. As a result, the server delivers the *HK*(i) with its lifetime *LTHK*(i) and the nonce *N2* to the *AR*(i). That is, the *AAARep* message including them is sent to the *AR*(i). On receiving the message, the *AR*(i) prepares for the *HKRep* message, and then sends it to the *MN*. At this point, the *MAChrep* is computed with the *HK*(i) to protect the message. In order to verify the *HKRep* message, the *MN* firstly computes the *HK*(i), followed by checking with it whether the *MAChrep* is valid. In positive case, the *MN* believes that the *HK*(i) is successfully shared between the *AR*(i) and itself. Such a belief triggers the *MN* to perform the subsequent phases with the *AR*(i). The rest of this protocol is same as that of *KKP* except that the *UNA* message is not protected. For more details on *NEP*, see Fig. 5.



〔Fig. 5〕 Narayanan et al.'s protocol

(3) Discussion

Based on the AAA infrastructure, *NEP* allows the *MN* and the *AR* to securely exchange their handover keys even without any public key operation. Thus, this protocol can support resource-constrained mobile devices. Moreover, it provides the handover key independence because each handover key is newly generated with the help of the HKS server.

However, NEP has the following problems:

- In every handover, the HKS should help the MN and the AR to establish the handover key while relying on the AAA infrastructure. It is clear that the HKS's involvement results in the critical handover latency. Moreover, the HKS makes NEP non fault-tolerant as a bottleneck.
- NEP is not FMIPv6-seamless due to the *HKReq*, *AAAReq*, *AAARep* and *HKRep* messages. The additional messages have a significant impact on this protocol's performance while resulting in additional *RTT*s.
- In 〔9〕, it is emphasized that the *AR*(i) should check if the MN exists at the claimed *CoA*(i) prior to sending the *AAAReq* message. Without a proof of the *MN*'s *CoA*(i) ownership, legitimate but malicious *MN*s can redirect traffic belonging to themselves or any other node at will. Unfortunately, despite such an emphasis, NEP does not sup-

port the tight bind between the *MN*'s handover key and *CoA*.

- The *RtSolPr*, *PrRtAdv*, and *UNA* messages are not protected in this protocol.

3.4 You-Sakurai-Hori's protocol (YSHP)

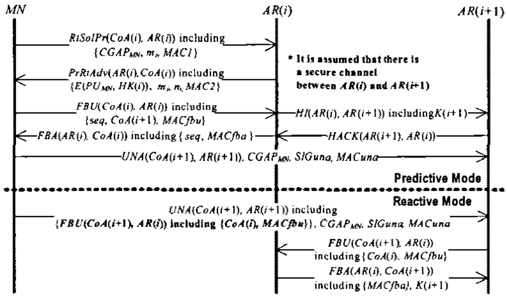
In 2009, You, Sakurai and Hori proposed a security protocol for FMIPv6, which addresses the high computation cost and the *DoS* attacks, from which KKP suffers (called YSHP) [10]. YSHP improves the flaws of KKP through the message protection secret shared between an *MN* and its *AR*. Especially, for the first message protection secret, YSHP depends on the AAA infrastructure.

(1) Preliminary

- Like KKP, each *MN* has a public/private key pair PU_{MN}/PR_{MN}, and its address is a CGA, which derived from PU_{MN}.
- During the bootstrapping step, each *MN* shares the message protection secret *K*(1) with the first access router *AR*(1) through the AAA infrastructure. Note that each *MN* is only once supported by the AAA infrastructure at the beginning.

(2) Operation

As depicted in Fig. 6, the *MN* starts this protocol by exchanging the *RtSolPr* and *PrRtAdv* messages with the current access router *AR*(*i*). At this point, a new handover key *HK*(*i*) is securely negotiated between the *MN* and the *AR*(*i*) in a way that it is encrypted with PU_{MN}. Prior to using PU_{MN}, the *AR*(*i*) verifies it with $CGAP_{MN}$. It is worth noting that unlike KKP, YSHP adopts the HMAC method to protect these



- m_i: The *i*th nonce of the *MN*, n_i: The *i*th nonce of the *AR*(*i*)
- $K(i+1) = SHA1(m_i|n_i|HK(i))$, where $i > 1$ and $K(1)$ is initially shared between the *MN* and the *AR*(1) during the bootstrapping with the help of the AAA infrastructure
- $MAC1 = HMAC(K(i), RtSolPr)$, $MAC2 = HMAC(K(i), PrRtAdv)$
- *seq*: It denotes the sequence number for the *FBU* and *FBA* messages
- $MACfbu = HMAC(HK(i), CoA|AR(i)|FBU)$, $MACfba = HMAC(HK(i), AR(i)|CoA|FBA)$
- $SIGuna = SIGN(PR_{MN}, UNA|CGAP_{MN})$,
  $MACuna = HMAC(K(i+1), UNA|CGAP_{MN}|SIGuna)$
  * Predictive mode: $CoA = CoA(i)$, Reactive mode: $CoA = CoA(i+1)$

(Fig. 6) You-Sakurai-Hori's protocol

messages. In this way, it can defend against the *DoS* attacks while reducing the heavy computation overhead caused by the public key based digital signature. For the HMAC method, this protocol uses a message protection secret *K*(*i*), which is derived from its related handover key and nonces. As assumed above, the *MN* shares the first message protection secret *K*(1) with the *AR*(1) relying on the AAA infrastructure during its bootstrapping step.

Once *HK*(*i*) has been established, the *MN* starts to perform the fast binding update by sending the *FBU* message to the *AR*(*i*), which then exchanges the *HI* and *HACK* messages with the *AR*(*i*+1). At this point, the (*i*+1)th message protection secret *K*(*i*+1) is included in the *HI* message. That makes it possible for the *AR*(*i*+1) to securely share *K*(*i*+1) with the *MN*. Then, the *AR*(*i*) returns the *FBA* message to the *MN*. As soon as the *MN* arrives at the new network, it sends the *UNA* message to the *AR*(*i*+1). In order to secure the *UNA* message, *YSHP* uses both the digital signature, *SIGuna*, and the HMAC value, *MACfna*. Note that while *SIGuna* is adopted to provide the handover key independence, *MACfna* is used to prevent the *DoS* attacks.

In the reactive mode, the *AR*(*i*+1) performs the fast binding update phase and obtains $K(i+1)$ prior to verifying the *UNA* message.

IV. Comparison Analysis and Research Challenges

4.1 Comparison

Table 1 summarizes and compares the protocols introduced in the previous section. Based on the public key cryptography, KKP and YSHP achieve the strong handover key exchange, the handover key independence, and the tight bind between *HK*(*i*) and *CoA*(*i*) while protecting all messages in addition to preventing the redirection attacks. However, KKP, unlike YSHP, is vulnerable to the *DoS* attack where an adversary sends a lot of *RtSolPr* messages to the *AR*(*i*), thus being occupied with considerable public key operations.

On the other hand, as shown in table 2, HEP and NEP are more efficient than others considering the computation overhead. In particular, NEP needs no public key operations, thus supporting resource-constraint devices. However, these protocols fail to satisfy the security requirements, while just protecting the *FBU* and *FBA* messages. That makes them vulnerable to the *DoS* and redirection attacks. Note that in spite of the light computation overhead, NEP is not indeed efficient due to the long latency caused by depending on the AAA infrastructure. Therefore, it is not desirable to apply these two protocols to FMIPv6 without any improvement. As a result, considering both security and efficiency, YSHP is superior to other protocols. But, this protocol still needs public key operations, while not being able to support resource-constraint devices. Clearly, that makes its application scope narrowed. Thus, it needs to provide an option for such devices.

[Table 1] Security comparison of the security protocols for FMIPv6 (S1: Tight bind between handover key and *CoA*, S2: Handover key independence, A1: Vulnerable to the DoS attacks, A2: Vulnerable to the redirection attacks)

Protocols	KKP	HKP	NEP	YSHP
Security Methods	CGA	CGA, OWHC	AAA	CGA, AAA
Protected Messages	all	*FBU, FBA*	*FBU, FBA*	all
S1	yes	yes	no	yes
S2	yes	no	yes	yes
A1	yes	yes	yes	yes
A2	no	yes	yes	yes
FMIPv6-Seamless Structure	yes	yes	no	yes

[Table 2] Computation overhead of the security protocols for FMIPv6
(S: sign, V: verify, E: public key encryption, D: public key decryption, H: hash, M: HMAC)

Protocols	KKP	HKP
KKP	n(2S+V+D +2M)	n(S+2V+E +2M)
HKP	S+V+D +2n(H+M)	S+V+E +2n(H+M)
NEP	6nM	3nM
YSHP	n(S +D +5M)	n(V+E+5M)

4.2 Research challenges

Though MIPv6 has become the major de facto standard for a mobility support in the Internet, it is not widely deployed and available yet [14]. The main reason is why MIPv6 requires *MN*s to be involved in their own mobility management, resulting in their modification for mobility related signaling messages. In order to solve this problem, Proxy Mobile IPv6 (PMIPv6) has been proposed while gaining much attention

[15]. As a network based mobility management approach, this protocol supports mobility for *MN*s without their involvement. That is, it does not require *MN*s to be modified for mobility function. With the expectation that PMIPv6 will overcome the obstacles of MIPv6, related research challenges are introduced. Among them, fast handover for PMIPv6 has been studied as one important research issue [14][16][17]. Yokota et al. have developed and standardized Fast Handovers for Proxy Mobile IPv6 (called FPMIPv6) [16]. Similar to FMIPv6, this scheme makes *MN*s' handover context directly transferred between their *AR*s. But, such an approach requires all *AR*s to have a security association with others, resulting in the inefficiency in flow management. In [17], Han and Park tried to solve this problem by indirectly transferring the handover context via the *Local Mobility Anchor* (*LMA*), which serves as a home agent for *MN*s in PMIPv6. On the other hand, Kiriyama et al. improve the Han-Park's scheme to make the best use of the PMIPv6 functions as well as efficiently perform the follow management [14].

Unfortunately, any proper security mechanism has not been proposed for PMIPv6 and its fast handover enhancements so far. It is clear that without being protected, they can be vulnerable to the various security threats [5][18]. Therefore, it is necessary to develop a security mechanism, which seamlessly works with them as well as address the security threats.

V. Conluding Remark

In this paper, the security protocols for FMIPv6 were analyzed, and then compared with each other in terms of the security requirements and the computation overhead. After discussing each protocol's advantages and weaknesses, we showed that YSHP achieves the strong security with less computation overhead than that of KKP.

We believe that this study provides a proper overview of security issues related to FMIPv6, thus contributing to strengthen the security of mobile network and its protocol.

References

[1] D. Johnson, C. Perkins, and J. Arkko, "Mobility Support in IPv6," IETF RFC 3775, June 2004

[2] R. Koodli, "Mobile IPv6 Fast Handovers," IETF RFC 5268, June 2008

[3] H. Soliman, C. Castelluccia, K. ElMalki, and L. Bellier, "Hierarchical Mobile IPv6 (HMIPv6) Mobility Management," IETF RFC 5380, Oct. 2008

[4] J. Arkko, C. Vogt, and W. Haddad, "Enhanced Route Optimization for Mobile IPv6," IETF RFC 4866, May 2007

[5] R.H. Deng, J. Zhou, and F. Bao, "Defending against redirect attacks in mobile IP," In Proc. of the 9th ACM conference on Computer and Communications Security, pp. 59-67, Nov. 2002.

[6] I. You, K. Sakurai and Y. Hori, "A Security Analysis on Kempf-Koodli's Security Scheme for Fast Mobile IPv6," IEICE Transaction on Communications, Vol. E92-B, no. 06, pp.2287-2290, June 2009

[7] J. Kempf and R. Koodli, "Distributing a Symmetric Fast Mobile IPv6 (FMIPv6) Handover Key Using SEcure Neighbor Discovery (SEND)," IETF RFC 5269, June 2008

[8] W. Haddad and S. Krishnan, "Authenticating FMIPv6 Handovers," IETF Internet Draft, draft-haddad-mipshop-fmipv6-auth-02, Sep. 2006

[9] V. Narayanan, N. Venkitaraman, H. Tschofenig, G. Giaretta, and J. Bournelle,

"Establishing Handover Keys using Shared Keys," IETF Internet Draft, draft-vidya-mipshop-handover-keys-aaa-04, March 2007

[10] I. You, K. Sakurai, and Y. Hori, "An Enhanced Security Protocol for Fast Mobile IPv6," IEICE Transaction on Information & Systems, Vol. E92-D, No.10, pp. 1979-1982, Oct. 2009

[11] J. Arkko, J. Kempf, B. Zill, and P. Nikander, "SEcure Neighbor Discovery (SEND)," IETF RFC 3971, Mar. 2005

[12] T. Aura, "Cryptographically Generated Addresses (CGA)," IETF RFC 3972, Mar. 2005

[13] F. Dupont, M. Laurent-Maknavicius and J. Bournelle, "AAA for mobile IPv6," IETF Internet Draft, draft-dupont-mipv6-aaa-01, Nov. 2001

[14] S. Kiriyama, R. Wakikawa, J. Xia and F. Teraoka, "Context Reflector for Proxy Mobile IPv6," In Proc. of 2009 International Conference on Complex, Intelligent and Software Intensive Systems, pp. 588-594, Mar. 2009

[15] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil, "Proxy Mobile IPv6," IETF RFC 5213, Aug. 2009

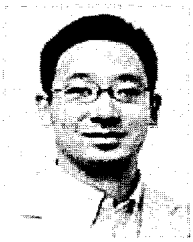
[16] H. Yokota, K. Chowdhury, R. Koodli, B. Patil, and F. Xia, "Fast Handovers for Proxy Mobile IPv6," IETF Internet Draft, draft-ietf-mipshop-pfmipv6-12, Dec. 2009.

[17] Y. Han and B. Park. "A Fast Handover Scheme in Proxy Mobile IPv6," IETF Internet Draft, draft-han-netlmm-fast-pmipv6-00, July 2008.

[18] C. Vogt and J. Kempf, "Security Threats to Network-Based Localized Mobility Management (NETLMM)," IETF RFC 4832, Apr. 2007

〈著者紹介〉

유 일 선 (Ilsun You) 정회원
Ilsun You received his M.S. and Ph.D. degrees in Computer Science from Dankook University, Seoul, Korea in 1997 and 2002, respectively. Since March 2005, he has worked as an Assistant Professor in the School of Information Science at the Korean Bible University, South Korea. He is in the editorial board for International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC), Computing and Informatics (CAI), and Journal of Korean Society for Internet Information (KSII). Also, he has served as a guest editor of several international journals such as Mobile Information System, Intelligent Automation & Soft Computing, Journal of Intelligent Manufacturing and Wireless Communications and Mobile Computing. His main research interests include internet security, authentication, access control, MIPv6 and ubiquitous computing. He is a member of the IEICE, KIISC, KSII and IEEK.



요시아키 호리 (Yoshiaki Hori)
Yoshiaki Hori received B.E., M.E., and D.E. degrees from Kyushu Institute of Technology, Iizuka, Japan in 1992, 1994, and 2002, respectively. From 1994 to 2003, he was a Research Associate in Common Technical Courses, Kyushu Institute of Design, Fukuoka. From 2003 to 2004, he was a Research Associate in the Department of Art and Information Design, Kyushu University, Fukuoka. Since March 2004, he has been an Associate Professor in the Department of Computer Science and Communication Engineering, Kyushu University. He has been an Associate Professor in the Department of Informatics, Kyushu University. His research interests include network security, network architecture, and performance evaluation of network protocols on various networks. He is a member of IEEE, ACM, and IPSJ.

코우이치 사쿠라이 (Kouichi Sakurai)
Yoshiaki Hori received B.E., M.E., and D.E. degrees from Kyushu Institute of Technology, Iizuka, Japan in 1992, 1994, and 2002, respectively. From 1994 to 2003, he was a Research Associate in Common Technical Courses, Kyushu Institute of Design, Fukuoka. From 2003 to 2004, he was a Research Associate in the Department of Art and Information Design, Kyushu University, Fukuoka. Since March 2004, he has been an Associate Professor in the Department of Computer Science and Communication Engineering, Kyushu University. He has been an Associate Professor in the Department of Informatics, Kyushu University. His research interests include network security, network architecture, and performance evaluation of network protocols on various networks. He is a member of IEEE, ACM, and IPSJ.