모바일 IPv6 환경에서 제한된 계산 능력을 갖는 모바일 노드를 지원하는 바인딩 갱신 인증 프로토콜에 관한 연구☆

A Study on Secure Binding Update Protocol Supporting Mobile Nodes with Constraint Computational Power in Mobile IPv6 Environment

최 승 교* 유 일 선**
Choi, Sung Kyo You, Il Sun

요 약

최근 PDA나 핸드폰과 같이 제한된 계산능력을 갖는 이동 장치가 증가함에 따라 공개키 암호화 연산을 적용하는 모바일 IPv6 바인딩 갱신 인증 프로토콜에서 모바일 노드의 공개키 연산을 최소화하는 것이 강력히 요구되고 있다. 이를 위해 CAM-DH와 SUCV 같은 기존의 공개키 기반 프로토콜에서는 모바일 노드의 공개키 연산을 홈 에이전트에 위임하는 연산 최적화 옵션을 제공하였다. 그러나 이러한 프로토콜들은 연산 최적화 옵션을 제공하는데 있어서 여러 가지 문제점을 노출하였다. 특히, CAM-DH의 경우 홈 에이전트가 서비스 거부 공격에 취약하며 모바일 노드의 공개키 연산을 완전히 위임받지 못하는 문제점을 갖는다. 본 논문에서는 이러한 CAM-DH의 문제점을 개선하며 또한 Aura의 이중 해쉬 기법을 통해 CAM-DH에서 적용하는 CGA의 보안성을 강화시킨다. CAM-DH와의 비교를 통해 개선된 프로토콜이 모바일 노드의 계산 비용을 최소화하고 강화된 보안성과 향상된 관리능력을 제공함을 알 수 있다.

Abstract

In MIPv6 environment, an important design consideration for public key based binding update protocols is to minimize asymmetric cryptographic operations in mobile nodes with constraint computational power, such as PDAs and cellular phones. For that, public key based protocols such as CAM-DH, SUCV and Deng-Zhou-Bao's approach provides an optimization to offload asymmetric cryptographic operations of a mobile node to its home agent. However, such protocols have some problems in providing the optimization. Especially, CAM-DH with this optimization does not unload all asymmetric cryptographic operations from the mobile node, while resulting in the home agent's vulnerability to denial of service attacks. In this paper, we improve the drawbacks of CAM-DH. Furthermore, we adopt Aura's two hash-based CGA scheme to increase the cost of brute-force attacks searching for hash collisions in the CGA method. The comparison of our protocol with other public key based protocols shows that our protocol can minimize the MN's computation overhead, in addition to providing better manageability and stronger security than other protocols.

☞ Keyword : Mobile IPv6, Binding Update, Cryptographically Generated Address (CGA), Return Routability (RR), CAM-DH

1. Introduction

In Mobile IP version 6 (MIPv6) environment, each mobile node (MN) belongs to a home link and is always addressable by its home address (HoA) assigned by the home link, regardless of its current point of attachment to the Internet [1,2,7]. While attached to some foreign link away from its home, each MN is also addressable by one or more care-of addresses (CoA), which provide information about its current location. The basic idea in MIPv6 is to allow a home agent (HA), a router on a

* 정 회 원 : 삼척대학교 컴퓨터공학과 교수
skchoi@samcheok.ac.kr
** 정 회 원 : 한국성서대학교 정보과학부 전임강사
isyou@bible.ac.kr
[2004/12/16 투고 - 2005/03/14 심사 - 2005/05/26 심사완료]
☆ 본 연구는 2004학년도 삼척대학교 학술연구비 지원으로 이루어졌음.

MN's home link, to work as a stationary proxy for the MN. Whenever the MN is away from home, the HA intercepts packets destined to the MN's HoA, encapsulates them, and tunnels them to the MN's registered CoA. When the MN wants to send packets to a CN, it sends them to the HA over the reverse tunnel. The HA un-encapsulates the packets and forwards them to the CN. Thus, MIPv6 enables MNs to have both mobility and reachability. But, since such a basic procedure requires tunneling through the HA, it results in longer paths and degraded performance. To mitigate the performance problem, MIPv6 includes route optimization that allows the MN and its CN to exchange packets directly, excluding the HA after the initial setup phase. A binding is the association between a MN's HoA and CoA. The MN initializes the route optimization by sending binding update (BU) messages including its current binding to the CN. Upon receiving the BU message, the CN learns and caches the MN's current binding. After that, the CN can directly send packets to the MN using the MN's CoA. However, without a security solution, such a binding update mechanism exposes the involved MNs and CNs to various security threats. The essential requirement to address the security threats is for the CN to authenticate the MN sending the BU message. Only after successfully authenticating the MN, the CN has to update its binding cache entries. Unfortunately, it is so difficult to achieve strong authentication between two previously unknown nodes (MN and CN) where no global security infrastructure is available.

Recently, the Return Routability (RR) protocol has been accepted as the basic technique for securing BUs. Nevertheless, the RR protocol has some potential drawbacks, both in terms of its security properties and also performance [2]. Unlike the RR protocol, the protocols such as Child-proof Authentication for MIPv6 (CAM), CAM-DH, Statistic Uniqueness and Cryptographic Verifiability (SUCV) and Address Based Keys (ABKs) have been proposed based on public key [2-6]. These protocols attempted to associate the MN's address with its public key to avoid additional security infrastructure such as Public Key Infrastructure (PKI), by using the novel methods such as Cryptographically Generated Address (CGA) and identity-based crypto systems. There are two important design considerations in the public key based protocols: performance and public key mechanism [2]. Since asymmetric cryptographic operations are computationally intensive, performance should be importantly considered. Especially, it is highly desirable to minimize the expensive cryptographic operations in mobile devices with constraint computational power, such as PDAs and cellular phones. For that, some protocols such as CAM-DH, SUCV and Deng-Zhou-Bao's protocol provide an optimization to off-load the expensive cryptographic operation of the MN to its HA [2,6-7]. But they have the following problems. In CAM-DH, all expensive cryptographic operations of the MN are not off-loaded to its HA [7]. In SUCV, the optimization results in the HA's additional cost of managing the MN's private key [6]. Deng-Zhou-Bao's protocol needs, an additional security infrastructure, PKI, since it uses public key certificates (PKC) [2].

CAM-DH is a CGA-based protocol for securing BUs in MIPv6, which combines the

BAKE/2 protocol with a digitally signed Diffie-Hellman key exchange [4]. In this protocol, the MN uses its private key to sign a Diffie-Hellman exponent that is then used to negotiate a session key. As mentioned above, this protocol provides the optimization for low-power MNs, but the optimization does not unload all asymmetric cryptographic operations from the MN, while resulting in HAs's vulnerability to denial of service attacks.

In this paper, we improve the optimization of CAM-DH in order that the HA can prevent the denial of service attacks and perform all asymmetric cryptographic operations on behalf its MNs. Furthermore, we adopt Aura's two hash-based CGA scheme to prevent brute-force attacks searching for hash collisions in the CGA method [8]. The rest of the paper is organized as follows. Section 2 reviews CAM-DH. In section 3, we describe the two hash-based CGA scheme and propose an enhanced CAM-DH. Section 4 analyzes the enhanced protocol. Finally, section 5 draws some conclusions.

2. Review of CAM-DH protocol

In this section, CAM-DH is reviewed and its weaknesses are analyzed.

Notation is as follows.

$h()$: a cryptographic secure one-way hash function

$prf(k, m)$: a keyed hash function. It accepts a secret key k and a message m, and generates a pseudo random output.

P_X/S_X : a public and private key pair of X.

$S_X(m)$: node X's digital signature on a message m.

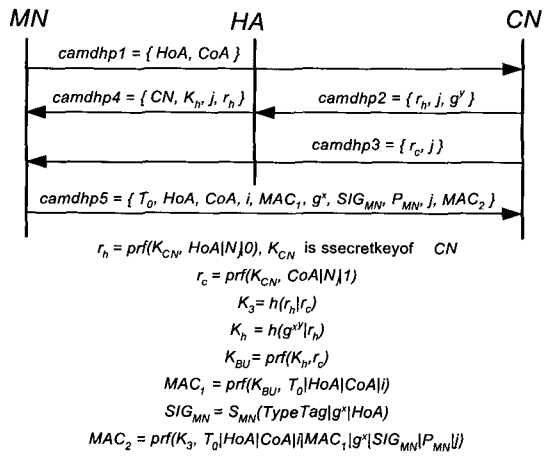
$m|n$: concatenation of two messages m and n.

CN : CN represents both the correspondent node and its IP address.

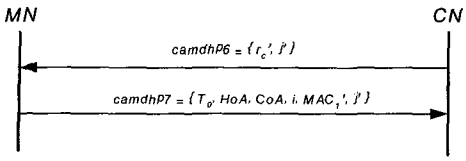
Let p and g be the public Diffie-Hellman parameters, where p is a large prime and g is a generator of the multiplicative group Zp*. To keep notations compact, $g^x \ mod \ p$ is written simply as g^x. It is assumed that the values of p and g are agreed upon before hand by all the parties concerned.

2.1 CAM-DH protocol

Fig. 1 outlines CAM-DH with an optimization for MNs with constraint computational power, such as PDAs and cellular phones. For the optimization, the HA intercepts the second message camdhp2 and performs certain processing on it before forwarding it to the MN. Because communication between the MN and the HA is protected with pre-establish security association in the MIPv6, such an optimization is available [2,7].



〈Fig. 1〉 CAM-DH Protocol



〈Fig. 2〉 Binding Request of CAM-DH Protocol

CGA is IPv6 address where the interface identifier is generated by hashing the address owner's public key [3,8-9]. The address owner can use the corresponding private key to assert address ownership and to sign messages sent from the address without any additional security infrastructure. In this protocol, each MN's *HoA* is generated from its public key P_{MN} and used as a CGA. The MN uses its private key S_{MN} to sign a Diffie-Hellman exponent that is then used to negotiate a session key. Especially, by testing RR of the MN's new addresses, it can protect against denial of service attacks on the CN. That is, the CN will not perform any asymmetric cryptographic operations until it knows it is talking to a MN that has been authenticated with RR. Moreover, it can prevent malicious mobile node flooding attacks where a malicious MN is a legitimate MN in its home link and its actions are legal BU operations [2]. When the CN's binding cache entry is about to expire, the CN sends the MN a binding request containing a fresh challenge as shown in Fig. 2.

2.2 Weaknesses of CAM-DH Protocol

In spite of high-level security, CAM-DH has the following drawbacks.

First, the optimization for low-power MNs results in the HA's vulnerability to denial of service attacks, since the HA uses Diffie-Hellman key agreement to calculate a session key K_h without authenticating the CN. Thus, the HA is easy to be flooded with a storm of camdp2 messages. Second, the protocol does not unload all asymmetric cryptographic operations from the MN, since the HA just performs expensive cryptographic operations for a session key K_h instead of the MN. Therefore, the MN should compute SIG_{MN} with its private key S_{MN}. Third, CAM-DH, a CGA-based protocol, is vulnerable to brute-force attacks searching for hash collisions, because of using only the 62 bits of the interface identifier as the hash value for the address owner's public key.

3. Our Proposed Protocol

In this section, we propose a novel approach that can resolve the drawbacks of CAM-DH. Our approach advances the optimization for mobile devices with constraint computational power. Especially, it uses Aura's two hash-based CGA scheme to overcome the limitation of the CGA based protocol [8].

3.1 Applying the Two Hash Based CGA Scheme

CGA is IPv6 address where the interface identifier is generated by hashing the address owner's public key. However, as computers become faster, the 64 bits of the interface identifier will not be sufficient to prevent attackers from searching for hash collisions.

Recently, Aura proposed a new CGA scheme where two hash values are computed instead of one [8]. The first hash value (Hash1) is used to produce the interface identifier (i.e.

```
Sec   = Address & 7
Mask1 = 0x00000000000000000000000000000000 if Sec=0,
        0xfff00000000000000000000000000000 if Sec=1,
        0xffffff000000000000000000000000000 if Sec=2,
        0xfffffffff00000000000000000000000 if Sec=3
        0xffffffffffff0000000000000000000 if Sec=4,
        0xfffffffffffffff00000000000000000 if Sec=5,
        0xffffffffffffffffff000000000000000 if Sec=6, and
        0xffffffffffffffffffffff00000000000 if Sec=7
Mask2 = 0x00000000000000000300000000000000
Mask3 = 0x0000000000000000ffffffffffffffff8

  (Hash1 & Mask3) || Mask2  == Address & Mask3
              Hash2 & Mask1 == 0

where '&' means bit-and operation and '||' means bit-or operation
```

〈Fig. 3〉 The definition of a CGA using bit masks

rightmost 64 bits) of the address. The purpose of the second hash (Hash2) is to artificially increase that computational complexity of generating new addresses and, consequently, the cost of brute-force attacks.

In the proposed CGA scheme, a CGA format is defined as an IPv6 address where the 12*Sec leftmost bits of the second hash value Hash2 are zero, and the rightmost 64 bits of the first hash value Hash1 equal the interface identifier of the address. The three rightmost bits of the address, which encode the security parameter Sec to determine the level of security, and the universal and group bits are ignored in the comparison. The latter two bits must both be one. The above definition can be stated in terms of the following three bit masks (Mask1, Mask2, Mask3) as shown in Fig. 3.

3.2 System Setup

In our protocol, a home link is associated with a public/private key pair P_H and S_H in a digital signature scheme. A HA in the home link keeps the public/private key pair, and derives a CGA from the public key P_H.

Each CGA can be associated with the self-signed X.509 v3 certificate. Fig. 4 shows the self-signed X.509 v3 certificate structure, its extension and two 128-bit hash values (Hash1 and Hash2) [8-9]. As an alternative to the certificate, an optimized parameter format can be used. The optimized format is simply the concatenation of the DER-encoded subject Public Key Info and CGAParameters data value. In our protocol, the optimized format is used.

The process of obtaining a new CGA is as follows.

1) Generate a public/private key pair P_H and S_H for a home link.
2) Generate a new CGA via the algorithm presented in Fig. 5.
3) Create an optimized parameter format. The format is simply the concatenation of the DER-encoded subjectPublicKeyInfo and CGAParameters data value.

3.3 Protocol Operation

Our protocol is composed of two phases. In the first phase, the CN uses the CGA method

```
Certificate  ::=  SEQUENCE  {
     tbsCertificate       TBSCertificate,
     signatureAlgorithm   AlgorithmIdentifier,
     signature            BIT STRING  }    -- signature must be verified

TBSCertificate  ::=  SEQUENCE  {
     version         [0]  Version DEFAULT v1,
     serialNumber         CertificateSerialNumber,
     signature            AlgorithmIdentifier,
     issuer               Name,            -- value: home link subnet Prefix
     validity             Validity,        -- validity must be checked
     subject              Name,            -- value: home link subnet Prefix
     subjectPublicKeyInfo SubjectPublicKeyInfo,  -- value: address owner, Public key

     issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version shall be v2 or v3
     subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version shall be v2 or v3
     extensions      [3]  Extensions OPTIONAL
                          -- If present, version shall be v3 --  }

Extensions  ::=  SEQUENCE SIZE (1..MAX) OF Extension

Extension  ::=  SEQUENCE  {
     extnID      OBJECT IDENTIFIER,        -- value: cgaExtnID = { 1 3 6 1 4 1 311 TBD }
     critical    BOOLEAN DEFAULT FALSE,    -- value: false
     extnValue   OCTET STRING  }           -- value: encoded CGAParameters

CGAParameters ::= SEQUENCE {
                    modifier        OCTET STRING (SIZE 12),
                    routingPrefix   OCTET STRING (SIZE 8),
                    collisionCount  INTEGER (0..2) }

Hash1 = MD5(DER_encode(SubjectPublicKeyInfo)||CGAParameters data values)
Hash2 = MD5(DER_encode(SubjectPublicKeyInfo)||modifier data values)
```

〈Fig. 4〉 A Self-Signed X.509 v3 certificate structure for the CGA

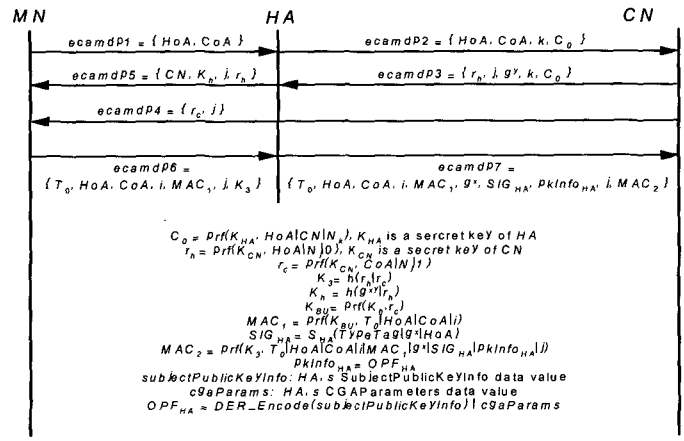
and the Diffie-Hellman key agreement to authenticate the first BU message from the MN, while sharing a session key K_h with the MN. In the second phase, the subsequent BU messages from the MN are authenticated through the session key shared between the MN and the CN.

```
/*-------------------------------------------------------------
// type
SubjectPublicKeyInfo  : an ASN.1 structure of type SubjectPublicKeyInfo
CGAParameters         : an ASN.1 structure of type CGAParameters

// input
HL: the home link subnet prefix (a 64-bit Routing Prefix)
PH: a HA's public key
Sec: security parameter Sec, which is an unsigned 3-bit integer
-------------------------------------------------------------*/

IPv6Addr* generateCGA(IPv6AddrPrefix HL, PublicKey PH, unsigned int Sec)
{

   SubjectPublicKeyInfo *pkInfo=NULL;
   CGAParameters *cgaParams=NULL;
   unsigned char *derPKInfo=NULL,*derCgaParams=NULL;
   unsigned char *Hash1=NULL, *Hash2=NULL;
   IPv6Addr *newCGA=NULL;

         // 1. DER-encode a HA's public key as an ASN.1 structure
                              of the type SubjectPublicKeyInfo

   pkInfo = new SubjectPublicKeyInfo(PH); CHK_ERR(pkInfo);
   derPKInfo = pkInfo->DER_encode(); CHK_ERR(derPKInfo);

         // 2.Create an ASN.1 structure of type CGAParameters
   cgaParams = new CGAParameters(); CHK_ERR(cgaParams);
   cgaParams->modifier = 0;
   cgaParams->routingPrefix = HL;
   cgaParams->collisionCount = 0;
   derCgaParams = cgaParams->DER_encode(); CHK_ERR(derCgaParams);

         // 3. Compute Hash2
   while(1){
         Hash2 = MD5(DER_Concatenate(derPKInfo, DerValue(cgaParams->modifier)));
         CHK_ERR(Hash2);
            // compare the 12*Sec leftmost bits of Hash2 with zero
         if(Is_Leftmost_bits_Zero(Hash2,12*Sec)) break;
         cgaParams->modifier++;
         if(cgaParams->modifier == Max_Modifier))
                  goto error_handler;
   }

               // 4. Generate a new CGA
   while(1){
         Hash1 = MD5(DER_Concatenate(derPKInfo,derCgaParams)); CHK_ERR(Hash1);

            // a new CGA = HL| rightmost 64 bits of Hash1
         newCGA = Make_IPv6_Address(HL,Rightmost_64_bits_of(Hash1));
         CHK_ERR(newCGA);
         Set_Group_Bit(newCGA1);
         Set_Universal_Bit(newCGA1);
         Set_Sec_Bit(newCGA,Sec);
         if(Is_There_Address_Collision(newCGA))  break;

         cgaParams->collisionCount++;
         if(cgaParams->collisionCount > 2) goto error_handler;
         Free_DER_Value(derCgaParams);
         derCgaParams = cgaParams->DER_encode();
   }

         ......      // 5. deinitialize values

   return newCGA;

error_handler:                  // 6. Handle errors
         .....

   return NULL;
}
```

<Fig. 5> CGA Generation Algorithm

Our protocol improves the drawbacks of CAM-DH as follows.

First, to off-load the asymmetric cryptographic operations of the MN to the HA, our protocol allows the HA to perform the expensive operations on behalf of the MN. For that, the HA keeps the public/private key pair P_{HA}/S_{HA} and uses CGA, derived from its public key P_{HA}, as its own MIPv6 address. The CN should validate the public key P_{HA} with the HA's CGA before verifying the signature SIG_{HA}. Such a mechanism enables our protocol to be more manageable and scalable than other public key based protocols where the MN binds its public key with its own address, in addition to unloading all asymmetric cryptographic operat-



〈Fig. 6〉 The First Phase of Our Secure Binding Update Protocol

ions from the MN.

Second, to prevent denial of service attacks on the HA, a cookie C_0 is created and added to the first message {*HoA*, *CoA*} sent by the MN. Only if the cookie is valid, the HA performs asymmetric cryptographic operations.

Third, instead of just using the standard BU message of the RR protocol, we provide a new additional step for the second phase. This step, unlike the standard BU message, enables our protocol to prevent malicious mobile node flooding attacks in the second phase as well as the first phase.

Fourth, to overcome the limited length of the hash used in the CGA, our protocol uses Aura's two hash based CGA, which enhances CAM-DH's security by increasing the cost of brute-force attacks by a factor 2^{12*Sec}.

1) First Phase

Fig. 6 outlines the first phase of our protocol where the HA functions as a security proxy for the MN, testifies the legitimacy of the MN's HoA, facilitates authentication of the MN to the CN and establishes a session key for them.

① ecamdp1= {*HoA*, *CoA*} : In ecamdp1, the MN tries to contact the CN, giving its *HoA* and *CoA*. Here we use CN to represent both the correspondent node and its IP address. ecamdp1 is sent to the MN's home link via the IPsec protected secure tunnel.

② ecamdp2 = {*HoA*, *CoA*, *k*, C_0} : Upon arriving at the home link, ecamdp1 is intercepted by the HA using IPv6 Neighbor Discovery [7,10]. Instead of forwarding the message to the CN, the HA creates a cookie C_0 and sends ecamdp2 to the CN. Before performing asymmetric cryptographic operations for ecamdp5 message, the HA compares the created cookie C_0 with the one sent by the CN to prevent denial of service attacks.

③ ecamdp3 = {r_h, *j*, g^y, *k*, C_0}, ecamdp4 = {r_c, *j*} : When receiving ecamdp2, the CN sends ecamdp3 to the MN's *HoA* and ecamdp4 to the MN's *CoA*. The

two messages enable the CN to test RR of the MN's new *CoA* and prevent denial of service attacks. To prevent denial of service attacks, the CN uses the same g^y as its Diffie-Hellman public value for each protocol run instead of generating a new value.

④ ecamdp5 = {CN, K_h, j} : The HA intercepts ecamdp3 using IPv6 Neighbor Discovery, and then validates the cookie C_0 to prevent denial of service attacks. If C_0 is valid, it freshly generates the Diffie-Hellman public value g^x and computes $K_h = h(g^{xy} | r_h)$, and then sends ecamdp5 to the MN.

⑤ ecamdp6 = {T_0, *HoA*, *CoA*, i, MAC_1, j, K_3} : After receiving ecamdp4 and ecamdp5, the MN computes $K_{BU} = prf(K_h, r_c)$, $MAC_1 = prf(K_{BU}, T_0|HoA|CoA|i)$ and $K_3 = h(r_h|r_c)$. Then, it sends ecamdp6 to the CN. The message is delivered to the MN's home link and is intercepted by the HA using IPv6 Neighbor Discovery.

⑥ ecamdp7 = {T_0, *HoA*, *CoA*, i, MAC_1, g^x, SIG_{HA}, $pkInfo_{HA}$, j, MAC_2} : After receiving ecamdp6, the HA computes SIG_{HA} with its private key S_{HA} and $MAC_2 = prf(K_3, T_0, HoA, CoA, i, MAC_1, g^x, SIG_{HA}, pkInfo_{HA}, j)$. Then, it sends ecamdp7 to the CN. On receipt of ecamdp7, the CN firstly checks MAC_2 with K_3. It should attempt to compute K_{BU} and verify MAC_1 with the computed K_{BU} only if MAC_2 is valid. Since the MN cannot compute K_3 without receiving ecamdp4 and ecamdp5, the CN is able to test RR of the MN's new *CoA* through MAC2. Thus, through this RR check, the CN can resist against denial of service attacks and malicious mobile node flooding attacks while being confident that *CoA* is the MN's new care of address. If MAC_1 is valid, the CN creates a cache entry for the MN's *HoA* and the key K_h, which will be used for authenticating subsequent BU messages from the MN. Before computing $K_{BU} = prf(K_h, r_c)$, the CN should verify the HA's *CGA* and SIG_{HA}. The algorithm for verifying the HA's *CGA* is defined in Fig. 7. $pkInfo_{HA}$ and *cgaParams* are retrieved from the optimized parameter format. If the HA's CGA is valid, the CN verifies SIG_{HA} using $pkInfo_{HA}$. When the verification is positive, the CN can be confident that the MN's *HoA* is valid and the Diffie-Hellman public value g^x is freshly generated by the HA.

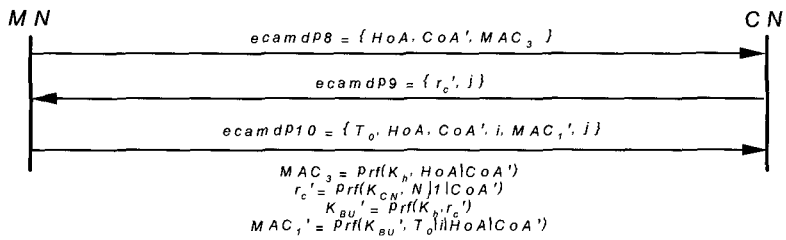
2) Second Phase

Fig. 8 outlines the second phase of our protocol where the CN authenticates the subsequent BU messages from the MN with the session key *Kh* established in the first phase. Especially, we provide an additional step, ecamdp8 and ecamdp9, to test RR of the MN's new care of address *CoA'*, instead of just using the standard BU message. Since the MN cannot compute valid MAC_1' without receiving ecamdp9 which the CN sent to its new care of address *CoA'*, malicious mobile node flooding attacks are not available in the second phase. When the CN's binding cache entry is about to expire, the CN sends the MN a binding request containing a fresh challenge as shown in Fig. 2.

① ecamdp8 = {*HoA*, *CoA'*, MAC_3} : When the

```
/*-------------------------------------------------------------------
        // constant
cgaExtnID : { 1 3 6 1 4 1 311 TBD }

        // input
CGA : a HA's address
pkInfo : a HA's public key information
cgaParams : CGAParameters

-------------------------------------------------------------------*/

BOOL VerifyCGA(IPv6Addr* CGA,
        SubjectPublicKeyInfo* pkInfo,
        CGAParameters* cgaParams)
{
        unsigned char *derPKInfo=NULL, *derCgaParams=NULL;
        unsigned char *Hash1=NULL, *Hash2=NULL;

        unsigned int Sec;

                // 1. Check input values
        CHK_ERR(CGA); CHK_ERR(pkInfo); CHK_ERR(cgaParams);

                // 2. Compare the group and universal bits
                //                  in the address to one
        if(!Is_Set_Group_Bit(CGA) || !Is_Set_Universal_Bit(CGA))
                goto error_handler;

                // 3. Get Sec value and
                //    check that the collisionCount value is 0, 1 or 2
        Sec = Sec_Value_of(CGA);
        if(cgaParams->collisionCount > 2) goto error_handler;

                // 4. Verify the subnet prefixe of the CGA
        if(!Is_Equal_Two_Subnet_Prefixes(
                Subnet_Prefix_of(CGA),
                cgaParams->routingPrefix)) goto error_handler;

                // 5. Verity the interface identifier of the CGA
        derPKInfo = pkInfo->DER_encode(); CHK_ERR(derPKInfo);
        derCgaParams = cgaParams->DER_encode(); CHK_ERR(derCgaParams);
        Hash1 = MD5(DER_Concatenate(derPKInfo,derCgaParams)); CHK_ERR(Hash1);


        if(!Verify_Interface_Identifier(
                Interface_Identifier_of(CGA),
                Rightmost_64_bits_of(Hash1))) goto error_handler;

                // 6. Verity the Sec of the CGA
        Hash2 = MD5(DER_Concatenate(derPKInfo, DerValue(cgaParams->modifier)));
        CHK_ERR(Hash2);
                // compare the 12*Sec leftmost bits of Hash2 with zero
        if(!Is_Leftmost_bits_Zero(Hash2,12*Sec)) goto error_handler;

        .....   // 5. deinitialize values
        return TRUE;

error_handler:          // 6. Handle errors
                .....

        return FALSE;
}
```

〈Fig. 7〉 CGA Verification Algorithm



〈Fig. 8〉 The Second Phase of Our Secure Binding Update Protocol

MN moves to a new location, it computes MAC_3 with the shared session key K_h and then sends it with a new care of address *CoA'* and its *HoA* to the CN.

② ecamdp9 = {r_c', j} : Upon receiving ecamdp8, the CN verifies MAC_3. If it is valid, the CN calculates and sends a new challenge r_c' with j to the MN's *CoA'*.

③ ecamdp10 = {T_0, *HoA*, *CoA'*, i, MAC_1', j} : Upon receiving ecamdp9, the MN computes K_{BU}' and MAC_1', and then sends ecamdp10 to the MN. After the CN receives ecamdp10, it validates MAC_1'. Since the MN cannot compute MAC_1'

without receiving ecamdp9 which the CN sent to its *CoA'*, the valid MAC_1' can assure the CN that *CoA'* is the MN's new care of address. In that case, the BU is accepted.

4. Analysis of the proposed protocol

In this section, our protocol is analyzed in terms of security and performance. Then, it is compared with other protocols based on public key.

4.1 Security

1) Denial of Service Attacks

In this section, we analyze the behaviors of our protocol against denial of service attacks. Since the MN-HA path is protected with pre-establish security association, we focus on denial of service attacks in the MN-HA path.

① ecamdp2 : An intruder can try to attack the CN by sending a storm of ecamdp2 messages. Since our protocol uses the same g^y as the CN's Diffie-Hellman public value instead of generating a new one, it is not vulnerable to such an attack.

② ecamdp3 : An intruder can try to attack the HA by sending a storm of ecamdp3 messages. In our protocol, the HA prevents such an attack by using a cookie C_0. That is, it compares the created cookie C_0 with the one included in ecamdp3, before performing asymmetric cryptographic operations for ecamdp5 message.

③ ecamdp7 : An intruder can try to attack the CN by sending a storm of ecamdp7 messages. In our protocol, the CN tests RR of the MN's new *CoA* to protect itself against denial of service attacks. For that, the CN sends ecamdp4 to the MN's HoA and ecamdp5 to the MN's *CoA*. Since the MN cannot compute K_3 without receiving ecamdp4 and ecamdp5, the CN is able to test RR of the MN's new *CoA* through MAC_2. Thus, through this RR check, the CN can resist against denial of service attacks while being confident that *CoA* is the MN's new care of address.

2) Redirection Attacks

Redirection attacks can be classified into two categories, session hijacking and malicious mobile node flooding [2]. In our protocol, the CN can strongly authenticate the MN's BUs while detecting forged ones by using the RR check, the two hash-based CGA and the signature SIG_{HA}. Thus, this strong authentication enables our protocol to prevent the session hijacking attacks. Also, our protocol tests RR of the MN's new *CoA* to prevent malicious mobile node flooding attacks in both the first phase and the second phase. Especially, it, unlike CAM-DH, achieves such security in the second phase through an additional step, ecamdp8 and ecamdp9, to test RR of the MN's new *CoA*.

3) The Cost of Brute-Force Attacks

As computers become faster, the 64 bits of the interface identifier will not be sufficient to prevent attackers from searching for hash collisions. Our protocol uses the two hash-based CGA scheme to prevent such brute-force attacks. The scheme includes the routing prefix of the address in the input for the first hash

value Hash1 and uses the second hash value Hash2 to increase the cost of brute-force attacks.

During the address generation phase of our protocol, the input for the additional hash Hash2 is modified by varying the value of modifier until the leftmost 12*Sec bits of Hash2 are zero. This increases the cost of address generation approximately by a factor of 2^{12*Sec}. It also increases the cost of brute-force attacks by the same factor (from 2^{59} to $2^{59+12*Sec}$). Therefore, our protocol is more secure than other CGA based approaches such as CAM-DH and SUCV, which require the cost of brute-force attacks, $O(2^{62})$.

4.2 Manageability and Scalability

There are two important design considerations in the public key based protocols: performance and public key mechanism [2]. The public key mechanism is used to securely bind a subject's name with its public key, while having significant impact on the entire system architecture and operation. In most public key based BU protocols such as CAM, CAM-DH, SUCV and ABKs, a MN's public key is bound with its HoA as the subject's name. However, such a bind is not desirable for the following reasons. First, HoAs of MNs are subject to renumbering both when service providers change and when configurations change so they may not be as persistent as other subject names (e.g., domain names). Second, HoAs can be leased to a MN for a fixed length of time. When the HoA's lease time expires, the association of the HoA with the MN becomes invalid and the HoA may be reassigned to another MN elsewhere in the Internet. Therefore, it is very difficult in practice to keep track of correct associations between all MNs and their HoAs in a consistent and timely manner. Also, if a HoA's lease time is short, the HoA should be often changed

On the other hand, a MIPv6 network system where a HA's public key is bound with its HoA is much more manageable and scalable because of the following reasons. First, HoAs of HAs are normally much more persistent than ones of MNs. Second, they are managed by system administration staff who can do a much better job in keeping track of their changes than keeping track of MNs's changes. Third, the number of HAs is significantly smaller than the number of MNs.

In our protocol, the HA, instead of the MN, keeps the public/private key pair and derives a CGA from the public key P_{HA}. Thus, it is more manageable and scalable than other protocols where the MN binds its public key with its own HoA.

4.3 Performance

We evaluate the performance of our protocol in terms of the cryptographic operations that each MN should perform. The costs can be expressed as follows.

$C_{OurP\text{-}MN}$ = *the cost for computing* K_3 + *the cost for computing* K_{BU} + *the cost for computing* $MAC_1 = C_{hash} + 2*C_{hmac}$

$C_{CAMDH\text{-}MN}$ = *the cost for computing* K_3 + *the cost for computing* K_{BU} + *the cost for computing* MAC_1 + *the cost for computing* SIG_{MN} + *the cost for computing* MAC_2 =

$C_{sign} + C_{hash} + 3*C_{hmac}$

$C_{OurP\text{-}MN}$: *the cost of cryptographic operations that a MN should perform in our protocol*

$C_{CAMDH\text{-}MN}$: *the cost of cryptographic operations that a MN should perform in CAM-DH protocol*

C_{sign} : *the cost for one signing operation*

C_{hash} : *the cost for one hash operation*

C_{hmac} : *the cost for one HMAC operation*

In comparison to CAM-DH, our protocol needs an additional cost, $2*C_{hmac} + C_{hash}$, for a cookie C_0 and two hash-based CGA. But, as shown above the MN in our protocol just needs $C_{hash} + 2*C_{hmac}$ without any cost for asymmetric cryptographic operations, whereas the one in CAM-DH needs $C_{sign} + C_{hash} + 3*C_{hmac}$. Thus, it can minimize cryptographic operations that each MN should perform, while satisfying an important design consideration for public key based BU protocols [2].

4.4 Comparison

In this section, we compare our protocol with other public key protocols, such as CAM-DH, SUCV and Deng-Zhou-Bao's protocol, which provide an optimization to off-load the expensive cryptographic operation of the MN to its HA [2,6,11-12]. The comparison is summarized in Table 1.

In spite of providing the strongest security, Deng-Zhou-Bao's protocol should have a trusted CA, which is a critical limitation where no global security infrastructure is available. Unlike Deng-Zhou-Bao's protocol, our protocol needs no additional infrastructure since it uses

〈Table 1〉 The Comparison of the Proposed Protocol with Other Protocols

	Ours	CAM-DH	SUCV	DZB
1	2-hash-CGA	1-hash-CGA	1-hash-CGA	PKC
2	HA	MN	HA	HA
3	O(259+12*Sec)	O(262)	O(262)	O(2128) or O(2160)
4	0	1	0	0
5	High	Low	Low	High
6	◉	X	◉	◉
7	◉	△	△	△
8	No	No	No	Yes

◉: not vulnerable, △: attacks may happen, X: vulnerable
1. Mechanism binding the public key with its owner
2. Node who generates and manages the private key/public key pair
3. Cost of brute force attacks
4. Asymmetric cryptographic operations the MN should perform
5. Manageability and Scalability
6. Ability to Prevent Denial of Service Attacks
7. Ability to Prevent Redirection Attack
8. Is there Trusted CA?
* DZB: Deng, Zhou and Bao's approach
* PKC: Public Key Certificate

two hash based CGA, which has more expensive cost of brute force attacks than SUCV and CAM-DH. Furthermore, our protocol can prevent redirection attacks, whereas other protocols are vulnerable to malicious mobile node flooding attacks.

Except for CAM-DH, the above protocols allow MNs to perform no asymmetric operations. Our protocol and Deng-Zhou-Bao's protocol are more manageable and scalable since they allow HAs, instead of MNs to perform cryptographic operations with their own public/private key pairs. Like our protocol and Deng-Zhou-Bao's protocol, SUCV allows HAs, instead of MNs, to perform cryptographic operations. But, in the protocol HAs keep public/private key pairs of MNs and use them to perform cryptographic operations. Therefore, SUCV's cost for managing public/private key pairs is higher than the other protocols.

5. Conclusions

In MIPv6 environment, an important design consideration for public key based binding update protocols is to minimize asymmetric cryptographic operations in mobile nodes with constraint computational power, such as PDAs and cellular phones. For that, CAM-DH provides an optimization to offload asymmetric cryptographic operations of a MN to its HA. However, this optimization does not unload all asymmetric cryptographic operations from the MN, while resulting in HAs's vulnerability to denial of service attacks.

In this paper, we propose a novel approach that can resolve the drawbacks of CAM-DH.

Our protocol improves the drawbacks of CAM-DH as follows.

First, our protocol advances the optimization for mobile devices with constraint computational power in order that the HA can perform all asymmetric cryptographic operations on behalf of the MN. For that, the HA keeps the public/private key pair P_{HA}/S_{HA} and uses CGA, derived from its public key P_{HA}, as its own MIPv6 address. Such a mechanism enables our protocol to be more manageable and scalable than other public key based protocols where the MN binds its public key with its own address,

Second, to prevent denial of service attacks on the HA, a cookie is used. Only if the cookie is valid, the HA performs asymmetric cryptographic operations.

Third, instead of just using the standard BU message of the RR protocol, we provide a new additional step for the second phase. This step, unlike the standard BU message, enables our protocol to prevent malicious mobile node flooding attacks in the second phase as well as the first phase.

Fourth, to overcome the limited length of the hash used in the CGA, our protocol uses Aura's two hash based CGA, which enhances CAM-DH's security by increasing the cost of brute-force attacks by a factor 2^{12*Sec}.

Thus, our protocol can achieve stronger security than other CGA-based protocols.

The comparison of our protocol with other public key based protocols shows that our protocol can provide better manageability and stronger security than other ones, in addition to minimizing the MN's computation overhead.

References

[1] J. Arkko, "Security Framework for Mobile IPv6 Route Optimization," IETF, <draft-arkko-mipv6ro-secframework-00.txt>, Nov. 2001. Work in progress.

[2] R. Deng, J. Zhou, and F. Bao, "Defending Against Redirect attacks in Mobile IP," Proceedings of the 9th ACM Conference on Computer and Communications Security, Nov. 2002.

[3] G. O'Shea and M. Roe, "Child-proof authentication for MIPv6 (CAM)," ACM Computer Communications Review, Vol. 31, No. 2, April 2001.

[4] M. Roe, T. Aura, G. O'Shea, and J. Arkko, "Authentication of Mobile IPv6 Binding Updates and Acknowledgments," IETF, <draft-roe-mobileip-updateauth-02.txt>, Feb. 2002. Work in progress.

[5] S. Okazaki, A. Desai, C. Gentry and et. el., "Securing MIPv6 Binding Updates Using Address Based Keys (ABKs)," IETF, <draft-okazaki-mobileip-abk-01.txt>, Oct. 2002. Work in progress.

[6] G. Montenegro, C. Castelluccia, "Crypto-Based Identifiers(CBIDs): Concepts and Applications", ACM Transations on Information and System Security, Vol. 7, No. 1, February 2004, pp97-127

[7] D. Johnson, C. Perkins and J. Arkko, "Mobility Support in IPv6," IETF, <draft-ietf-mobileip-ipv6-24.txt>, Jun. 2003. Work in progress.

[8] T. Aura, "Cryptographically Generated Addresses (CGA)," IETF, <draft-aura-cga-00.txt>, Feb. 2003. Work in progress.

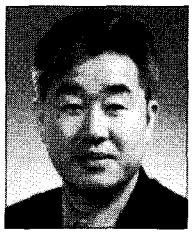
[9] R. Housley, W. Ford, T. Polk, and D. Solo, "Internet X.509 public key infrastructure certificate and CRL profile," RFC 2459, Jan. 1999.

[10] T. Narten, E. Nordmark, and W. Simpson, "Neighbor Discovery for IP Version 6 (Ipv6)," RFC 2461, Dec. 1998.

[11] Y. Won, K. Cho, "Comparison and Analysis of Protocols for the Secure Binding Updates in MIPv6," the KIPS transactions : part C, Vol.10-C, No.6, Oct., 2003.

[12] I. You, Y. Won, and K. Cho, "A An Improved Protocol for the Secure Mobile IPv6 Binding Updates," the KIPS transactions : part C, Vol.11-C, No.5, Oct., 2004.

◐ 저 자 소 개 ◑



최 승 교 (Sung Kyo, Choi)
1982년 단국대학교 전기공학과 졸업(학사)
1992년 단국대학교 대학원 전산통계학과 졸업(석사)
2001년 단국대학교 대학원 전산통계학과 졸업(박사)
1994년~현재 삼척대학교 컴퓨터공학과 교수
관심분야 : 컴퓨터구조론, 성능평가, 시뮬레이션, 컴퓨터 통신, 네트워크보안.
E-mail : skchoi@samcheok.ac.kr

유 일 선 (Il-Sun You)
1995년 단국대학교 전산통계학과 졸업(학사)
1997년 단국대학교 대학원 전산통계학과 졸업(석사)
2002년 단국대학교 대학원 전산통계학과 졸업(박사)
2005년~현재 한국성서대학교 정보과학부 전임강사
관심분야 : 네트워크 보안, 사용자 인증 및 접근통제, .NET 프레임워크
E-mail : isyou@bible.ac.kr