Defending Against Redirect Attacks in Mobile IP

Robert H. Deng
Labs for Information Technology
21 Heng Mui Keng Terrace
Singapore 119613
+65 6874-7862
deng@lit.a-star.edu.sg

Jianying Zhou
Labs for Information Technology
21 Heng Mui Keng Terrace
Singapore 119613
+65 6874-8543
jyzhou@lit.a-star.edu.sg

Feng Bao
Labs for Information Technology
21 Heng Mui Keng Terrace
Singapore 119613
+65 6874-8456
baofeng@lit.a-star.edu.sg

ABSTRACT

The route optimization operation in Mobile IP Version 6 (MIPv6) allows direct routing from any correspondent node to any mobile node and thus eliminates the problem of "triangle routing" present in the base Mobile IP Version 4 (MIPv4) protocol. Route optimization, however, requires that a mobile node constantly inform its correspondent nodes about its new care-of addresses by sending them binding update messages. Unauthenticated or malicious binding updates open the door for intruders to perform redirect attacks, i.e., malicious acts which redirect traffic from correspondent nodes to locations chosen by intruders. How to protect binding update messages to defend against redirect attacks is a challenging problem given the open environment in which MIPv6 operates. In this paper, we first look at two solutions proposed by the IETF Mobile IP Working Group and point out their weaknesses. We then present a new protocol for securing binding update messages. We also show that our protocol achieves strong security and at the same time is highly scalable to wide spread deployment.

Categories and Subject Descriptors

C.3.3 [**Computer-Communication Networks**]: General – *security and protection.*

General Terms

Design, Security.

Keywords

Authenticated key-exchange, mobile IP, mobile IP security, redirect attack, secure binding update.

1. INTRODUCTION

Mobile networking technologies, along with the proliferation of numerous portable and wireless devices, promise to change people's perceptions of the Internet. In mobile networking, communications activities are not disrupted when a user changes his/her device's point of attachment to the Internet - all the network reconnections occur automatically and transparently to the user.

In today's Internet, the Internet Protocol (IP) routes packets from source to destination according to the subnet prefix derived from the destination IP address by masking off some of the low-order bits. Thus, an IP address typically carries with it information that specifies the IP node's point of attachment to the Internet. As a mobile node roams in the Internet, it needs to change its IP address every time it moves to a new location. On the other hand, however, to maintain existing transport-layer connections as a mobile node moves from one place to another, it must keep its IP address the same, changing the IP address will cause the existing transport layer connections to be disrupted and lost.

The above dilemma is solved in Mobile IP (MIP) by allowing a mobile node to be addressed by two IP addresses, a home address and a care-of address. The former is an IP address assigned to the mobile node within its subnet prefix on its home link and the latter is a temporary address acquired by the mobile node while visiting a foreign link. The dual address mechanism in MIP allows packets to be routed to the mobile node regardless of its current point of attachment and the movement of the mobile node away from its home link is transparent to transport and higher-layer protocols. MIP version 4 (MIPv4) was specified in [1] and the most recent specification for MIP version 6 (MIPv6) was published by the IETF Mobile IP Working Group in [2]. Mobility support in IPv6 is considered particularly important, since mobile devices are predicted to account for a significant fraction of the population of the Internet during the lifetime of IPv6.

MIPv6 shares many features with MIPv4 but there are several major differences. Among them is the support for "Route Optimization" as a built-in fundamental part of the MIPv6 protocol, rather than an after-thought being added as an optional extension that may not be supported by all the nodes as in MIPv4. The integration of route optimization functionality allows direct routing from any correspondent node to any mobile node, without needing to pass through the mobile node's home link and be forwarded by its home agent, and thus eliminates the problem of "triangle routing" present in MIPv4.

Route optimization in MIPv6 requires that the mobile node, its home agent and the correspondent node maintain a Binding Cache. A binding is the association of a mobile node's home address with a care-of address for that mobile node, along with the remaining lifetime of that association. A mobile node uses binding update messages to notify its correspondent node or its home agent of its current binding. Unfortunately, unauthenticated binding update messages provide intruders an easy means to launch "Redirect Attacks", i.e., malicious acts which redirect traffic from correspondent nodes to destinations chosen by intruders. Therefore, security of the binding update messages is of paramount importance for MIPv6 to meet its basic security requirements. An earlier IETF draft on MIPv6 was returned by the Internet Engineering Steering Group (IESG) to the Mobile IP Working Group due to concerns about the security and scalability of binding update messages [3].

The rest of the paper is organized as follows. In Section 2, we give a short overview of the operations in MIPv6, with emphasis on route optimization and binding update operations. We also detail the types of redirect attacks and state the security assumptions in MIPv6. In Section 3, we review two solutions proposed by the IETF Mobile IP Working Group for protecting binding update messages against redirect attacks and point out their security limitations. Section 4 is devoted to our new protocol and its analysis. Finally, Section 5 contains our concluding remarks.

2. OPERATION, REDIRECT ATTACKS AND SECURITY ASSUMPTIONS IN MOBILE IPV6

2.1 Mobile IPv6 Basic Operation

In MIPv6 [2], every mobile node has a home address, an IP address assigned to a mobile node within its home link. A mobile node is always addressable by its home address, whether it is currently attached to its home link or is away from home. While a mobile node is at home, packets addressed to its home address are routed using the normal IPv6 routing mechanisms in the same way as if the node were never mobile. Since the subnet prefix of a mobile node's home address is the subnet prefix of its home link, packets addressed to it will be routed to its home link.

While a mobile node is away from home and attached to some foreign link (see Figure 1), it is also addressable by one or more care-of addresses, in addition to its home address. A care-of address is an IP address associated with a mobile node while visiting a particular foreign link. The subnet prefix of the mobile node's care-of address is the subnet prefix on the foreign link being visited by the node. A mobile node typically acquires its care-of address through stateless [4] or stateful (e. g., DHCPv6, [5]) address autoconfiguration. While on the foreign link, the mobile node registers its care-of address with its home agent by sending a Binding Update message to the agent. The home agent thereafter uses proxy Neighbor Discovery to intercept any IPv6 packets addressed to the mobile node's home address on the home link, and tunnels each intercepted packet to the mobile node's care-of address. To tunnel intercepted packets, the home agent encapsulates the packets using IPv6 encapsulation, with the outer IPv6 header addressed to the mobile node's care-of address.

A mobile node may at any time initiate route optimization operations with a correspondent node, allowing the correspondent node communicate directly with the mobile node, avoiding delivering traffic via the mobile node's home agent. The binding update mechanism is also used by correspondent nodes to dynamically learn and cache the mobile node's current binding. When sending a packet to the mobile node, a correspondent node checks its cached bindings for an entry for the packet's destination address. If a cached binding for this destination address is found, the node uses an IPv6 Routing Header [6] to route the packet to the mobile node by way of the care-of address indicated in this binding. If, instead, the correspondent node has no cached binding for this destination address, the node sends the packet normally (i.e., to the mobile node's home address with no routing header), and the packet is subsequently intercepted and tunneled to the mobile node by its home agent as described above.

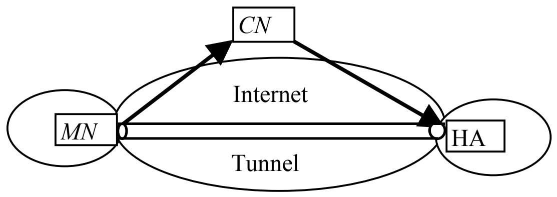

Figure 1. Basic operation in MIPv6.

2.2 Redirect Attacks

In the present paper we focus on redirect attacks and their countermeasures in MIPv6. We will not consider security issues such as data confidentiality, data integrity and user authentication since they are beyond the scope of MIPv6 and can be provided, for example, by IPsec or layers above IP. We classify redirect attacks in MIPv6 into two categories, *Session Hijacking* and *Malicious Mobile Node Flooding*, as depicted in Figure 2.

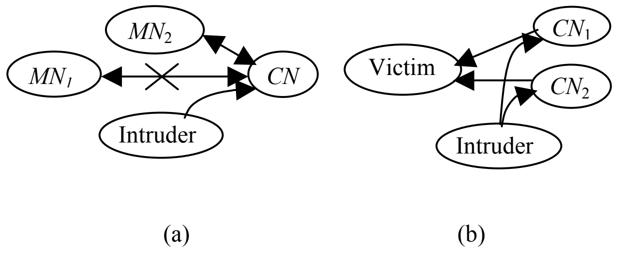

Figure 2. Redirect Attacks: (a) Session Hijacking; (b) Malicious Mobile Node Flooding.

1. <u>Session Hijacking</u>: In the session hijacking redirect attack shown in Figure 2(a), we assume that a mobile node MN_1 is communicating with a correspondent node *CN*. An intruder sends a forged binding update message (or replays an old binding update message) to *CN*, claiming that MN_1 has moved to a new care-of address belonging to a node MN_2. If *CN* accepts the fake binding update, it will start communicating with MN_2 instead of MN_1. This is an "outsider" attack since the intruder tries to redirect other nodes' traffic. Such an attack may result in information leakage, impersonation of the mobile node MN_1 or flooding of MN_2.

2. Malicious Mobile Node Flooding: In the malicious mobile node flooding attack depicted in Figure 2(b), an intruder, i.e., a malicious mobile node, sends valid binding update messages to its correspondent nodes CN_1 and CN_2, claiming that it has moved to the Victim's location. Here the Victim can be either a node or a network. For example, the intruder could initiate requests to video streaming servers, and flood the Victim's node or network by redirecting traffic from the video servers to the Victim. This attack is an "insider" attack since the malicious mobile node is a legitimate mobile node in its home link and its actions are "legal" binding update operations.

We note that, instead of targeting at correspondent nodes, the above attacks apply equally to home agents of mobile nodes. By sending forged or malicious binding update messages to a mobile node's home agent, an intruder can redirect traffic intended to the mobile node to a location of its choice.

2.3 Security Assumptions

We will make more or less the same security assumptions on MIPv6 as stated in [2]:

1. Pre-established security association between a mobile node and its home agent: Mobile nodes and home agents know each other, and can thus have a pre-established strong security association to reliably authenticate exchanged messages between them. IPsec's Encapsulating Security Payload (ESP) [7] is used to set up a secure tunnel between a mobile node and its home agent.
2. No pre-established security association between a mobile node and a random correspondent node: It is expected that MIPv6 will be used on a global basis between nodes belonging to different administrative domains, hence building a global authentication infrastructure to authenticate mobile nodes and random correspondent nodes would be a very demanding task, at least in the near to medium terms. However, this does not rule out the possibility of having fragmented authentication infrastructures within individual administration domains or even cross different domains. Furthermore, making a traditional authentication infrastructure to keep track of correct IP addresses for all hosts is either impossible or at least very hard due to the dynamic association between IP addresses and hosts.

Since a pre-established security association between a mobile node and its home agent is assumed, securing the binding update messages from a mobile node to its home agent is straightforward. Hence, during the rest of the paper, we will work on securing binding updates from mobile nodes to corresponding nodes.

2.4 Cryptographic Notation

We list below the cryptographic notation used throughout the paper for ease of reference:

$h()$	: a cryptographic secure one-way hash function, or one-way hash function in short, such as MD5 [8] and SHA [9].
$prf(k, m)$	: a keyed pseudo random function – often a keyed hash function [10]. It accepts a secret key k and a message m, and generates a pseudo random output. This function is used for both generation of message authentication codes and derivation of cryptographic keys.
P_X/S_X	: a public and private key pair of node X in a digital signature scheme such as RSA [11] or DSS [12].
$S_X(m)$	: node X's digital signature on a message m or on the hash of m.
$m\|n$	: concatenation of two messages m and n.

3. IETF'S SECURE BINDING UPDATE PROTOCOLS

In this section we describe and analyze two protocols for authenticating binding update messages. The Return Routability (RR) protocol appeared in [2] and the Cryptographic Generated Address (CGA) protocol was under discussion by the working group according to [13].

3.1 The RR Protocol and Its Analysis

Protocol Operation: In this protocol, each correspondent node *CN* keeps a secret key k_{CN} and generates a nonce N_j at regular intervals, say every few minutes. *CN* uses the same key k_{CN} and nonce N_j with all the mobile nodes it is in communication with, so that it does not need to generate and store a new nonce when a new mobile node contacts it. *CN* keeps both the current value of N_j and a small set of previous nonce values, $N_{j-1}, N_{j-2}, \cdots$. Older values are discarded, and messages using them will be rejected as replays. Message exchanges in the RR protocol are shown in Figure 3, where the *HoTI* (Home Test Init) and *CoTI* (Care-of Test Init) messages are sent by the mobile node *MN* simultaneously. The *HoT* (Home Test) and *CoT* (Care-of Test) are replies from *CN*. All the RR protocol messages are sent as IPv6 "Mobility Header" in IPv6 packets. In the representation of a protocol message, we will use the first two fields to denote source IP address and destination IP address, respectively. We will misuse the notation *CN* to let it denote the correspondent node as well as its IP address.

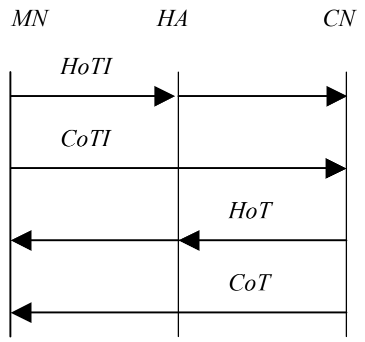

Figure 3. The Return Routability Protocol.

When *MN* wants to perform route optimization, it sends

$$HoTI = \{HoA, CN, r_H\}$$

and

$$CoTI = \{CoA, CN, r_C\}$$

to *CN*, where r_H and r_C are random values used to match responses with requests. *HoTI* tells *MN*'s home address *HoA* to

CN. It is reverse tunneled through the home agent *HA*, while *CoTI* informs *MN*'s care-of address *CoA* and is sent directly to *CN*. When *CN* receives *HoTI*, it takes the source IP address of *HoTI* as input and generates a *home cookie*

$$C_H = prf(k_{CN}, HoA|N_j)$$

and replies *MN* with

$$HoT = \{CN, HoA, r_H, C_H, j\},$$

where the index *j* is carried along to allow *CN* later efficiently finding the nonce value N_j that it used in creating cookie C_H. Similarly, when *CN* receives *CoTI*, it takes the source IP address of *CoTI* as input and generates a *care-of cookie*

$$C_C = prf(k_{CN}, CoA|N_i)$$

and sends

$$CoT = \{CN, CoA, r_C, C_C, i\}$$

to *MN*. Note that *HoT* is sent via *MN*'s home agent *HA* while *CoT* is delivered directly to *MN*. When *MN* receives both *HoT* and *CoT*, it hashes together the two cookies to form a session key

$$k_{BU} = h(C_H|C_C),$$

which is then used to authenticate the binding update message to *CN*:

$$BU = \{CoA, CN, HoA, Seq\#, LT, i, j, MAC_{BU}\},$$

where *Seq#* is a sequence number used to detect replay attack, *LT* is the proposed lifetime of the binding, and

$$MAC_{BU} = prf(k_{BU}, CoA|CN|HoA|Seq\#|LT|i|j)$$

is a message authentication code (MAC) protected by the session key k_{BU}. MAC_{BU} is used to ensure that *BU* was sent by the same node which received the *HoT* and *CoT* messages. The message *BU* contains *j* and *i*, so that *CN* knows which nonce values N_j and N_i to use to first re-compute C_H and C_C and then the session key k_{BU}. Note that *CN* is stateless until it receives *BU* and verifies MAC_{BU}. If MAC_{BU} is verified positive, *CN* may reply with a binding acknowledgement message

$$BA = \{CN, CoA, HoA, Seq\#, LT', MAC_{BA}\},$$

where *Seq#* is copied from the *BU* message, *LT'* is the granted lifetime, and

$$MAC_{BA} = prf(k_{BU}, CN|CoA|HoA|Seq\#|LT')$$

is a MAC generated using k_{BU} to authenticate the *BA* message. *CN* then creates a binding cache entry for the mobile node *MN*. The binding cache entry binds *HoA* with *CoA* which allows future packets to *MN* be sent to *CoA* directly.

An example implementation of the binding cache at *CN* is shown in Figure 4, where *HoA* is used as an index for searching the binding cache for the destination address of a packet being sent, *r_LT* is the remaining lifetime for this entry, and the sequence number *Seq#* is used by *CN* to sequence binding updates and by *MN* to match a return binding acknowledgement with a binding update. Each binding update sent by *MN* must use a *Seq#* greater than (modulo 2^{16}) the one sent in the previous binding update by the same *HoA*. It is no requirement, however, that the sequence number value strictly increase by 1 with each new binding update sent or received [2]. Note that the session key k_{BU} is not kept in the cache entry. When *MN* receives a binding update message, based on the nonce indexes *i* and *j* in the message, it re-computes the session key using k_{CN} and the list of the most recent nonce values, say $\{N_j, N_{j-1}, N_{j-2}\}$, and then verifies *BU* using the newly computed session key.

The mobile node *MN* maintains a Binding Update List for each binding update message sent by it, for which the lifetime has not yet expired. A binding update list for a correspondent node *CN* consists of *CN*'s IP address, *MN*'s home address *HoA* and care-of address *CoA*, the remaining lifetime of the binding, the maximum value of the sequence number sent in previous binding updates to *CN* and the session key k_{BU}.

Entry for *MN*: *HoA*, *CoA*, *r_LT*, *Seq#*	$k_{CN}, N_j, N_{j-1}, N_{j-2}$
Entries for other mobile nodes	

Figure 4. A binding cache implementation at *CN* in the RR protocol.

Discussion: In the RR protocol, the two cookie exchanges verify that a mobile node *MN* is alive at its home address *HoA* and care-of address *CoA*, respectively. The eventual binding update messages are protected using a keyed hash with the session key k_{BU} obtained by hashing the concatenation of the two cookies C_H and C_C. Therefore, security of the binding update messages hinges on the secret sharing of k_{BU} between *MN* and *CN*, which in turn hinges on the secrecy of at least one of the two cookies.

The IETF MIPv6 documents [2, 3] stated that the motivation for designing the RR protocol was to have sufficient support for mobile IP, without creating major new security problems. It was not the goal of the Mobile IP Working Group to protect against attacks that were already possible before the introduction of IP mobility. The protocol does not defend against an intruder who can monitor the *CN-HA* path. The argument was that such intruders would in any case be able to mount an active attack against *MN* when it is at its home location.

However, the design principle of the RR protocol, i.e., defending against intruder who can monitor the *CN-MN* path but not the *CN-HA* path, is fundamentally flawed since it violates the well known "weakest link" principle in security. After all, one has no reason to assume that an intruder will monitor one link and not the other, especially when the intruder knows that monitoring a given link is particularly effective to expedite its attack. While it is true that intruders are able to mount active attacks when a node is at home in the base IPv6, we demonstrate below that it is much easier to launch redirect attacks in MIPv6 than in the base IPv6.

First, let's consider the session hijacking attack shown in Figure 2(a). In the case of the base IPv6 without mobility (which is equivalent to the mobile node *MN* at its home link in MIPv6), to succeed in the attack, the intruder must be constantly present on the *CN-HA* path. In order to redirect *CN*'s traffic intended for *MN* to a malicious node, the intruder most likely has to get control of a router or a switch along the *CN-HA path*. Furthermore, after taking over the session from *MN*, if the malicious node wants to continue the session with *CN* while pretending to be *MN*, the malicious node and the router need to collaborate throughout the session. For example, the router tunnels *CN*'s traffic to the malicious node and vise versa.

In the case of MIPv6, the effort committed to break the RR protocol to launch a session hijacking attack could be considerably lesser. Assume that MN_1 and *CN* in Figure 2(a) are having an on-going communication session and the intruder wants to redirect *CN'*s traffic to his collaborator MN_2. The intruder monitors the *CN-HA* path (i.e., anywhere from MN_1's home network to *CN*'s network) to obtain *HoT*, extracts the home cookie C_H and sends it MN_2. Upon receiving C_H, MN_2 sends a *CoTI* to *CN* and *CN* will reply with a care-of cookie C_C. MN_2 simply hashes the two cookies to obtain a valid session key, and uses the key to send a binding update message to *CN* on behalf of MN_1. The binding update will be accepted by *CN* which will in turn direct its traffic to MN_2.

Next, consider the malicious mobile node flooding attack shown in Figure 2(b). In the base IPv6 without mobility, perhaps the best example of flooding attack is the DDoS attack in which a multitude of compromised systems attack a single target. There are many ways to launch a malicious mobile node flooding attack against a victim (which can be either a node or a network) in MIPv6. For example, the malicious node starts some traffic intensive sessions with correspondent nodes and moves to the victim's network or the border between the victim network and the outside world. It then runs the RR protocol to redirect traffic from the correspondent nodes to the victim's network by sending them binding update messages. The malicious mobile node does not need any special software or networking skill to launch this attack.

Due to the stateless nature of the RR protocol, it is easy for an intruder to cause havoc to, say B2C operations. Imagine a correspondent node provides on-line services to many mobile clients. The intruder can simply eavesdrop on the RR protocol messages to collect cookies on the border between the correspondent node and the Internet. The intruder then randomly hashes pairs of cookies to form session keys, and sends binding update messages to the correspondent node. This will cause redirection of traffic to randomly selected mobile clients and eventually bring down the services of the correspondent node.

Finally, we point out that the IETF MIPv6 specification limits the lifetime of a RR authorized binding to a maximum of 420 seconds [2]. This will have performance implications. Imagine having a time sensitive session between a mobile node and a correspondent node where the mobile node must perform the RR protocol every 420 seconds or less. Quality of communication will definitely suffer if the RR protocol cannot be executed in a timely manner due to congestion or malfunction of the home agent, home link or the *CN-HA* path.

3.2 The CGA Protocol and Its Analysis

The Cryptographic Generated Address (CGA) protocol for secure binding update, first proposed in [14], is under discussion in the IETF Mobile IP Working Group according to [13]. In IPv6, a 128-bit IP address is divided into a subnet prefix and an interface identifier. The home addresses of all the mobile nodes associated with a home link share the same home link subnet prefix and are differentiated by their unique interface identifiers.

Protocol Operation: Each mobile node *MN* has a public/private key pair P_{MN} and S_{MN} in a digital signature scheme. *MN*'s home address is given by $HoA = \{HL|II\}$, where *HL* is the *n*-bit home link subnet prefix and *II* is the (128-*n*)-bit interface identifier. The *II* field is obtained by taking the left-most (128-*n*) bits of the hash function output $h(P_{MN})$. A binding update message from *MN* to a correspondent node *CN* is given by

$$BU = \{CN, CoA, HoA, Seq\#, LT, P_{MN}, 128\text{-}n, SIG_{MN}\},$$

where

$$SIG_{MN} = \mathrm{S}_{MN}(CoA|CN|HoA|Seq\#|LT|P_{MN}|128\text{-}n)$$

is *MN*'s digital signature generated using its private key S_{MN}. Upon receiving the *BU*, *CN* computes $h(P_{MN})$, compares the left most (128-*n*) bits of $h(P_{MN})$ with the right most (128-*n*)-bit *II* in *HoA*, and verifies the signature using the public key P_{MN}. If the hash value matches the value of *II* and if the signature verification is positive, *CN* will accept the binding update message.

Discussion: The hash function *h*() here acts as a "one-to-one" mapping from a public key value to an interface identifier; it binds a public key value with an interface identifier. Since it is computationally hard to either find the private key or forge a digital signature given the public key, a match of $h(P_{MN})$ with *II* in *HoA* as well as positive verification of the signature on *BU* proves that *BU* was generated by the mobile node whose interface identifier portion is *II* and who knows the private key S_{MN}. This is the only assurance a correspondent node gets from *BU*. As a consequence, the protocol is able to provide good protection against the session hijacking attack provided the number of bits in *II*, (128-*n*), is large enough. If (128-*n*) is small, an intruder can randomly generate pairs of public and private keys, hash the public keys and look for a match to a target node's *II*. Once a match is found, the intruder is able to impersonate the target node and forge binding updates. The computational complexity of this brute force attack is on the order of $o(2^{(128-n)})$.

On the other hand, since this protocol does not provide any proof on the authorization of *MN* to use the particular *HoA*, it is not able to protect against the malicious mobile node flooding attacks. Actually, an intruder can just generate a public/private key pair, hashes the public key to form a home address, signs a binding update message which contains a victim's address as *CoA*, and sends it to a correspondent node. The correspondent node will accept the binding update and start sending traffic to flood the victim node.

The CGA protocol is computational intensive since every binding update message requires the mobile node to generate a digital signature and the correspondent node perform a verification of digital signature.

4. OUR SECURE BINDING UPDATE PROTOCOL

Our protocol employs public key cryptosystems in order to provide strong security and good scalability. There are two important design considerations in protocols using public key cryptosystems. The first is performance since public key cryptosystem operations are computationally intensive. Portable devices with constraint computational power, such as PDAs and cellular phones, are predicted to account for a majority or at least a substantial fraction of the population of mobile devices. It is crucial to keep the amount of public key cryptosystem operations in mobile devices to the absolute minimum.

The second consideration is the mechanism used to securely bind a subject's name with its public key since they have significant impact on the entire system architecture and operation. Such a binding is typically achieved using public key certificates issued by a trusted certification authority, or *CA* in short. A public key certificate at the minimum consists of a subject's name, its public key, valid time interval and *CA*'s digital signature on the above data items. In the MIPv6 environment, a mobile node could be issued a public key certificate with its home address as the subject's name. However, having public key certificates with IP addresses as subject's names is a bad practice for several reasons. First, IP addresses are often obtained by DNS (Directory Name Service) look-up and DNS does not provide a secure way of mapping names to IP addresses. Second, IP addresses 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) [15]. Third, IP addresses are leased to an interface for a fixed length of time. When an IP address's lease time expires, the association of the address with the interface becomes invalid and the address may be reassigned to another interface elsewhere in the Internet. There might be various reasons for keeping IP addresses' lease time short, such as for privacy protection. For devices which functions as client devices, reference [16] recommends to change their IP addresses periodically to prevent eavesdroppers and other information collectors from correlating the clients' seemingly unrelated activities over an extended period of time. Therefore, it is very difficult in practice for *CA*s to keep track of correct associations between IP addresses and all devices' interfaces in a consistent and timely manner, not to mention issuing and revoking public key certificates for them.

Subnet prefixes for home links, however, are much more trackable and manageable. First, a home link subnet prefix is normally much more persistent than a mobile node's home address. Second, the number of home links is significantly smaller than the number of mobile nodes. Third, subnet prefixes are managed by system administration staff who can do a much better job in keeping track prefix changes than keeping track of which IP address is associated with which individual mobile node.

Motivated by the above observations, our protocol is designed to possess the following features:

1. It performs one-way authenticated key-exchange between *MN* and *CN* where *MN* authenticates itself to *CN* and the exchanged session key is used to secure binding update messages from *MN* to *CN*.
2. It employs public key cryptosystems and is secure against powerful adversary who is able to launch both passive (e.g., eavesdropping at multiple points) and active (e.g., man-in-the-middle) attacks.
3. It is easy to manage and scalable. Instead of issuing public key certificates containing home addresses as subject names for individual mobile nodes, we issue public key certificates containing home link subnet prefixes as subject names for home links.
4. No public key cryptographic operations are performed at mobile nodes. MIPv6 assumes that home agents are trusted by mobile nodes as well as correspondent nodes and that communications between mobile nodes and their home agents are protected with pre-established security associations; home agents function as trusted security proxies for mobile nodes in the protocol. They testify the legitimacy of mobile nodes' home addresses, facilitate authentication of mobile nodes to correspondent nodes, and establish shared secret session keys for them.

System Setup: A home link is associated with a public/private key pair P_H and S_H in a digital signature scheme. The private key S_H is kept by a home agent *HA* in the home link, probably inside in a tamper-resistant hardware cryptographic processing device. The home link obtains a public key certificate,

$$Cert_H = \{HL, P_H, VI, SIG_{CA}\}$$

from a certification authority *CA*, where *HL* is the home link subnet prefix, *VI* is the valid duration of the certificate, and SIG_{CA} is *CA*'s signature on *HL*, P_H and *VI*. We assume correspondent nodes can obtain *CA*'s public key via various means, such as by embedding or configuring *CA*'s public key into MIPv6 implementations. Embedding *CA*s' public keys has been the approach in the tremendously successful SSL/TLS protocol [15], where popular browsers embed dozens of public key values of well-known *CA*s. The protocol also uses the Diffie-Hellman key exchange algorithm to arrive at a mutual secret value between parties of the protocol. 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 Z_p*. To keep our notation compact, we will write g^x mod *p* simply as g^x. Since generation of large primes in real time can be very time consuming, we assume that the values of *p* and *g* are agreed upon before hand by all the parties concerned.

Protocol Operation: As in the RR protocol, all the protocol messages here are carried within IPv6 "Mobility Header" which allows protocol messages to be piggybacked on any existing IPv6 packets. The protocol messages exchanged among a mobile node *MN*, its home agent *HA* and its correspondent node *CN* are shown in Figure 5. In the protocol, the existence of and operations performed by *HA* are transparent to both *MN* and *CN*. As far as *MN* is concerned, it sends message *REQ* to and receives *REP* from *CN*. Similarly, from *CN*'s point of view, it receives *COOKIE0*, *EXCH0* and *CONFIRM* from and sends *COOKIE1* and *EXCH1* to *MN*.

The use of cookies during the key exchange is a weak form of protection against an intruder who generates a series of request packets, each with a different spoofed source IP address and sends them to a protocol party. For each request, the protocol party will first validate cookies before performing computationally expensive public key cryptographic operations. For details on cookie generation and validation, please refer to [17].

As before, the first two fields in a protocol message are the source IP address and destination IP address, respectively. When *MN* wants to start route optimization operation with *CN*, it sends a route optimization request

$$REQ = \{HoA, CN, n_0\}$$

to *CN* via reserve tunneling, where n_0 is a nonce value used to match the reply message *REP*. Here we use *CN* to represent both the correspondent node and its IP address. Message *REQ* is sent to *MN's* home link via the IPsec protected secure tunnel. IPsec provides replay protection only when dynamic security association establishment is used. This may not always be possible and manual keying might be preferred in certain circumstances. For this reason, we have included n_0 to counter message replay. Upon arriving at the home link, *REQ* is intercepted by *HA* using IPv6 Neighbor Discovery [2, 18]. *HA* will not forward *REQ* to *CN*, instead, it creates a cookie C_0 and sends

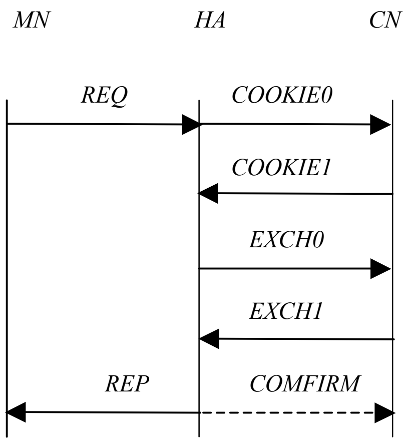

Figure 5. Message exchange in our new protocol.

$$COOKIE0 = \{HoA, CN, C_0\}$$

to *CN*. In reply, *CN* creates a nonce n_1 and a cookie C_1, and sends

$$COOKIE1 = \{CN, HoA, C_0, C_1, n_1\}$$

to *MN*. Note that the destination address in *COOKIE1* is *MN*'s home address *HoA*. As a result, this message is delivered to *MN*'s home link and intercepted by *HA* using IPv6 Neighbor Discovery. After receiving *COOKIE1*, *HA* checks on the validity of C_0, generates a nonce n_2 and a Diffie-Hellman secret value $x < p$, computes its Diffie-Hellman public value g^x and its signature

$$SIG_H = S_H(HoA|CN|g^x|\,n_1|\,n_2|TS)$$

using home link's private key S_H, where *TS* is a time stamp. This time stamp does not have to be checked by the recipient during the message exchange. It will be used to trace back the culprit should a malicious mobile node flooding attack have occurred. This point will be made clearer later. Finally, *HA* replies *CN* with

$$EXCH0 = \{HoA, CN, C_0, C_1, n_1, n_2, g^x, TS, SIG_H, Cert_H\},$$

where $Cert_H = \{HL, P_H, VI, SIG_{CA}\}$ is the public key certificate of the home link as defined before. Note that the values of n_1 and n_2 are included in the signature SIG_H in order to counter reply of old signatures and to resist chosen message attacks to the signature scheme, respectively.

When *CN* receives *EXCH0*, it validates the cookies, the home link's public key certificate $Cert_H$, the signature and importantly, checks for equality of the home link subnet prefix strings embedded in both $Cert_H$ and *HoA*. If all the validations and checking are positive, *CN* can be confident that the home address *HoA* of *MN* is authorized by its home link and the Diffie-Hellman public vaule g^x is freshly generated by *MN's* home link. *CN* next generates its Diffie-Hellman secret value $y < p$. It then computes its Diffie-Hellman public value g^y, the Diffie-Hellman key $k_{DH} = (g^x)^y$, a session key

$$k_{BU} = prf(k_{DH}, n_1|n_2)$$

and a MAC

$$MAC_1 = prf(k_{BU}, g^y|EXCH0),$$

and sends

$$EXCH1 = \{CN, HoA, C_0, C_1, g^y, MAC_1\}$$

to *MN*. Again, this message is intercepted by *HA*, which first validates the cookies, calculates the Diffie-Hellman key $k_{DH} = (g^y)^x$ and the session key $k_{BU} = prf(k_{DH}, n_1|n_2)$. *HA* then computes

$$MAC_2 = prf(k_{BU}, EXCH1),$$

and sends

$$CONFIRM = \{HoA, CN, MAC_2\}$$

to *CN*. The validity of MAC_2 is checked by *CN* and if it is valid, *CN* creates a cache entry for *HoA* and the session key k_{BU}, which will be used for authenticating binding update messages from *MN*.

Upon positive verification of MAC_1, *HA* also sends

$$REP = \{CN, HoA, n_0, k_{BU}\}$$

to *MN* through the secure IPsec ESP protected tunnel. After receiving *REP*, *MN* checks that n_0 is the same as the one it sent out in *REQ*. If so, *MN* proceeds to send *CN* binding update messages protected using k_{BU} as in the RR protocol. It should be noted that the *CONFIRM* message serves to confirm the key to *CN* and hence is optional.

Discussion: With a straightforward informal analysis it is easy to show that the above protocol performs a strong one-way authentication of *MN*/*HoA* to *CN* and provides *CN* with the confidence that it shares a secret session key with *MN*. A formal analysis of the protocol is beyond scope of the paper. Here we just would like to point out that the most important message is *EXCH0*. Recall that after receiving *EXCH0*, *CN* checks on the equality of the home link subnet prefix contained in both $Cert_H$ and *HoA*. This check is critical to detect man-in-the-middle attack. The signature $SIG_H = S_H(HoA|CN|g^x|n_1|n_2|TS)$ serves two purposes. First, it certifies that the Diffie-Hellman value g^x was originated by *MN*'s home agent *HA* on behalf of *MN* and second, it testifies that *HoA* is under *HA*'s (or equivalently the home link's) jurisdiction and is a legitimate home address for its mobile node *MN*. This authenticates *MN*'s *HoA* to *CN*.

Since a successful completion of the protocol allows *CN* to authenticate *MN*'s *HoA* as well as allows the two nodes to set up a secret session key for securing binding updates, the protocol prevents the session hijacking attack shown in Figure 2(a). This protocol, as any other protocols, is not able to completely prevent malicious mobile node flooding attacks. However, if a correspondent node were blamed to have bombarded a network service or site, it could present the signature $SIG_H = S_H(HoA|CN|g^x|n_1|n_2|TS)$ and point its fingers at the home agent *HA*. *HA* can subsequently nail down the mobile node *MN* which had a home address *HoA* and performed a binding update at the time specified by *TS*.

In our protocol, mobile nodes are not required to perform any public key cryptographic operations but correspondent nodes are.

Public key cryptographic operations may not a great concern if a correspondent node is a server machine. However, a correspondent node can also be a mobile node with limited computational power and battery life. In this case, public key operations supposed to be performed by the correspondent code can be off-loaded to its home agent. This scenario is depicted in Figure 6, where HA_{MN} and HA_{CN} are the the home agents of *MN* and *CN*, respectively. Since in MIPv6 it is assumed that a mobile node has a pre-established security association with its home agent, it is natural in our protocol to have HA_{MN} and HA_{CN} perform public key cryptographic operations on behalf of *MN* and *CN*, respectively. Also due to the symmetric arrangement of the entities, it is possible to perform a mutual authenticated key-exchange between *MN* and *CN* and establish session keys to secure binding updates messages in both directions.

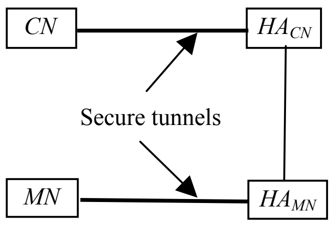

Figure 6. Scenario where *CN* is a mobile node.

Since our protocol uses strong cryptosystems, the secret session key k_{BU} established from the protocol could be used for a long period of time. This is in contrast to the RR protocol, where the protocol must be executed at least every 420 seconds even if the mobile node stays at the same foreign location.

5. CONCLUDING REMARKS

Mobile IP allows mobile nodes to have seamless communications when they change their point of attachment in the Internet and is poised to take off in a big way in the not too distant future. However, introduction of mobility into IP also brought with it new security issues and attacks, among them redirect attacks are perhaps the ones need the most urgent attention.

In this paper, we first classified redirect attacks into two types. In the first type of attacks -Session Hijacking attacks, an intruder hijacks an existing session between a mobile node and a correspondent node and redirects the correspondent node's traffic to a malicious location. In the second type of attacks - Malicious Mobile Node Flooding attacks, a mischievous mobile node sets up communication sessions with correspondent nodes, and then redirect traffic from the correspondent nodes to flood a victim node or network.

Next we reviewed and analyzed two protocols, the RR protocol and the CGA protocol, as proposed by the IETF Mobile IP Working Group. We showed that both protocols could be broken quite easily and are virtually not much use in defending redirect attacks.

We then proposed a new protocol to guard against redirect attacks. Our protocol makes use of a digital signature scheme and the Diffie-Hellman key exchange algorithm. An important contribution of our protocol is that we issue public key certificates for home links based on home link subnet prefixes, instead of issuing public key certificates for each and every mobile nodes based on their individual home addresses. We argued that such an approach makes certificate issuing, tracking, and revocation much more practical and manageable. In our protocol, a home agent functions as security proxy for its mobile nodes and testifies the legitimacy of a mobile node's home address to a correspondent node during protocol execution. Recognizing that most mobile nodes are constrained in processing power and battery life and that home agents can be easily equipped with increasingly low cost yet powerful cryptographic processing hardware accelerators, the protocol was designed to off load all the expensive public key cryptosystem operations from mobile nodes to their home agents. We showed that our protocol is secure against session hijacking attack. In addition, it is also capable of providing evidence to trace back the culprit should a malicious mobile node flooding attack has occurred.

The validity of the RR protocol authorized bindings is limited to a maximum of 420 seconds [2]. Consequently, a mobile node and a correspondent node need to perform the protocol once at least every 7 minutes. This introduces periodic overheads for the mobile and correspondent nodes and may degrade communication quality (such as introducing jitters in time sensitive communication sessions) due to congestion or slow response of the home agent. The CGA protocol requires a mobile node to generate a digital signature for every binding update it sends and the correspondent node to verify the signature for every binding update it receives. This could be a substantial computational burden if the mobile node moves quickly from one subnet to another. In our approach, the protocol needs just be performed once, say at the starting point of a communication session between a mobile node and a correspondent node. The secret session key resulted from the protocol execution can be used for authenticating binding updates for a relatively long period of time, say, throughout a communication session.

6. REFERENCES

[1] C. Perkins, "IP Mobility Support", *IETF RFC 2002*, October 1996.

[2] D. Johnson, C. Perkins and J. Arkko, "Mobility Support in IPv6", *IETF draft-ietf-mobileip-ipv6-18.txt*, 1 June 2002.

[3] A. Mankin et. al., "Threat Models Introduced by Mobile Ipv6 and Requirements for Security in Mobile Ipv6", *IETF draft-ietf-mipv6-scrty-reqts-02.txt*, May 2001.

[4] S. Thomas and T. Narten, "IPv6 Stateless Address Autoconfiguration", *IETF RFC 2462*, December 1998.

[5] J. Bound et. al, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", *IETF draft-ietf-dhc-dhcpv6-23.txt*, 22 April 2002.

[6] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specifications", *IETF RFC 2460*, December 1998.

[7] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)", *IETF RFC 2406*, November 1998.

[8] R. Rivest, "The MD5 Message Digest Algorithms", *IETF RFC* 1321, April 1992.

[9] NIST, "Secure Hash Standard", *NIST FIPS PUB 180*, May 1993.

[10] H. Krawczyk, M. Bellare and R. Canetti, "HMAC: Keyed-Hashing for Messaging Authentication", *IETF RFC 2104*, February 1997.

[11] R. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Commun. ACM*, Vol. 21, February 1978, pp. 120-126.

[12] NIST, "Digital Signature Standard", *NIST FIPS PUB 186*, May 1994.

[13] Security Group, DoCoMo USA Labs, "Mobile IP – Making it Secure and Practical", *Proceedings of the RSA Conference 2002*, February 18-22, 2002, San Jose.

[14] G. O'Shea and M. Roe, "Child-Proof Authentication for MIPv6 (CAM)", *Computer Communications Review*, April 2001.

[15] E. Rescorla, *SSL and TLS: Designing and Building Secure Systems*, Addison-Wesley, 2001.

[16] T. Narten and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", *IETF draft-ietf-ipngwg-temp-addresses-v2-00.txt*, July 2001.

[17] P. Karn and W. Simpson, "Photuris: Session-Key Management Protocol", *IETF RFC 2522*, 1999.

[18] T. Narten, E. Nordmark, and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", *IETF RFC 2461*, December 1998.