Per-Connection Return Routability Test in Mobile IPv6

Hengky Susanto and B.G. Kim

Department of Computer Science,
University of Massachusetts at Lowell
{hsusanto, kim}@cs.uml.edu

Abstract – As a mobile node switches from one subnet to another, Mobile IPv6 requires that it authenticate the new Care of Address of the new subnet with the corresponding node. The authentication process, called Return Routability (RR) Test, is required in every handoff and can delay the handoff process by a round trip delay between mobile and corresponding nodes. In this paper, two mechanisms are proposed to reduce the latency involved in RR tests and binding association. They eliminate triangular routing delay in RR tests by using multiple authentication tokens or easily regenerative tokens, so that the round trip delay in performing RR tests can be eliminated.

1. Introduction

In a wireless network, a mobile node (MN) acquires a new care of address (CoA) from the new access point. MN needs to inform the new CoA both to the Home Agent and the corresponding node (CN). In order to prevent an imposter of MN from generating a spurious CoA, a new CoA is protected by an authentication procedure, called the Return Routability (RR) test.

In mobile IPv6 [1], MN in required to send two messages carrying the new CoA to CN: one via the home agent and the other directly to CN. In response, CN generates tokens and send them in different paths: one via the home agent and the other directly to MN. When two response messages are received, MN combines both tokens to formulate a new message encryption key. Since both tokens are known to CN, it is able to formulate the identical message encryption key.

It can easily be seen that an RR test incurs at least a round trip delay (a longer delay between the one via the home agent and the other between MN and CN) each time a handoff takes place. In order to reduce a handoff delay, the previous access point is proposed to act as the home agent until the CoA is authenticated [2]. Another proposal in [3] allows an access point to forward information to its neighbor as soon as it learns that MN has detached itself from its network. In [4], a fast handoff scheme in Session Initiation Protocol (SIP) is suggested. However, SIP is an application-layer process and is not adequate to handle handoff at the network layer. Using higher priorities for handoff messages is considered in [5].

In order to expedite the authentication process of MN, MN is proposed to perform authentication with neighboring access points prior the handoff operation [6]. This is valid when all access points share information with each other. In [7], RR test is performed in the old access point prior to handoff so that data can be continued to be forwarded to CN. However, MN and CN will perform another RR Test with the new access point after the handoff. Another study in [8] suggests that RR Test can be initialized via old and new access points instead of MN, such that the delay involved in the authentication process can be reduced. However, MN still needs to be involved in a later stage of authentication and gains in handoff delays may not be substantial.
In this paper, we propose a per-connection RR test scheme, where handoff-related information is exchanged at the beginning of a connection between MN and CN. RR tests are not required to be performed in subsequent handoffs, thus eliminating round trip delays in per-handoff RR tests. In authentication pool scheme, a pool of authentication tokens is generated at the initial connection time and is encrypted and sent to CN. In subsequent handoffs, one of tokens in the pool is used to encrypt Binding Update (BU) message. A more elaborate scheme is to generate a pool of authentication tokens locally at MN and CN so that MN is not required to send a pool of tokens to CN. By reducing delays in handoffs, the proposed schemes are expected to be particularly beneficial to real-time connections.

The remainder of the paper is organized as follows. Details of mobile IPv6 RR test are described in the next section. In Sec. 3, two approaches in per-connection RR tests are described. A brief remark on performance implication is included. Finally, in Sec. 4, concluding remarks are presented.

2. Per-handoff RR Test in Mobile IPv6

After a successful handoff, MN informs the new CoA to the home agent and CN in a Bind Update (BU) message. In order to prevent an impersonator to generate a spurious BU message, an authentication process, called Return Routability (RR) test, is performed in mobile IPv6 [1]. Prior to sending a BU, MN launches two messages to CN to initiate the RR test as shown in Fig. 1. Home Test Init (HoTI) is sent via Home Agent (HA) whereas Care of Test Init (COTI) is sent directly to CN. CN responds to both messages by sending Home Test (HoT) and Care of Test (CoT) in response to HoTI and COTI, respectively. HoT is sent via HA and CoT is sent directly to MN.

The critical feature of authentication is that HoT and CoT contain home keygen (HK) and care of keygen (CK) tokens, respectively. BU message is encrypted based on a binding management key $K_{hm}$, which is derived from CK and HK tokens.

![Fig. 1. Return Routability Test](image)

HK and CK tokens are generated as follows. Let $K_{cn}$ denote the secret key of CN. Let ‘nonce’ denote a unique number which is guaranteed not to be repeated. HK is generated first by forming a concatenated string of home address of MN (Haddr), ‘nonce’ and ‘0’, and then hashing it by the secrete key of CN. Namely,

$$HK = \text{First}(64, \text{HMAC\_SHA1}(K_{cn}, (\text{Haddr} | \text{nonce} | 0)))$$

where HMAC\_SHA1 is a cryptographic hash function and function ‘First’ truncates a message to the first 64 bits. HMAC implants the secret key into the data and then compute a hash value from it. HMAC\_SHA1 is a particular function which uses SHA1 cryptographic hash function in the calculation of an HMAC [1,9,12].

Although, the output from the hash function is 96 bits, the reduced token size of 64 is sufficient to protect it from spoof [1]. The author claims that the attacker has to send a large number of messages before attacker can successfully launches the blindly spoof if the attacker is able to intersect the connection and recognize which packets carry the Binding Update. However, the security strength of binding update is beyond the scoop of this paper.

Similarly, CK token is generated by

$$CK = \text{First}(64, \text{HMAC\_SHA1}(K_{cn}, (\text{Caddr} | \text{nonce} | 0)))$$
CK = First(64, HMAC_SHA1(K_cn, (CoA|nonce| 1))).

HK and CK tokens are sent to MN in HoT can CoT messages, respectively. Like HoTI and CoTI messages, HoT and CoT messages are delivered in different paths. HoT is sent via HA. HA forwards the HoT message to MN through a secure tunnel, ensuring a secure delivery. CoT message, on the other hand, is sent directly to MN.

Upon receiving HK and CK tokens, MN hashes both keys to form a binding management key K_bm as follows.

K_bm = SHA1(HK, CK).

The binding management key, K_bm, is then used to encrypt the BU message to CN. Since CN has copies of HK and CK, it is also able to generate K_bm and can authenticate that BU indeed was generated from the intended MN.

3. Per-Connection RR Test Schemes

Two schemes presented in this section perform RR test at the beginning of a connection between MN and CN, and eliminate per-handoff RR test in subsequent handoffs. By reducing the delay involved in RR test, each handoff performance can be improved.

3.1. Authentication Pool Scheme

When a connection is established between MN and CN, a normal RR test is performed as in the previous section. From the initial RR test, both MN and CN generate the identical binding management key K_bm. When MN sends a BU to CN, MN includes, a pool of n token which is encrypted by K_bm.

A pool of tokens is constructed by repeatedly applying the hash function as follow:

TK_i = HMAC_SHA1(K_bm, M),  \(i = 1, 2, \ldots, n\)

where M = (Haddr |nonce, | CNaddr), namely concatenated string of home address (Haddr), nonce, and CN’s address (CNaddr). Uniqueness of tokens among TK_i is provided by nonce.

After connection is successfully established, both MN and CN have identical pool of tokens. The sequence of tokens to be used in successive handoffs may be in the order of tokens as it is embedded in the initial BU message or may be randomly selected. Random ordering of authentication tokens will require synchronization between MN and CN to determine which token will be used next. In the paper, we assume that tokens in the pool are used sequentially, for simplicity.

Suppose that MN is ready to perform the i-th handoff since the initial connection. MN generates a new binding key so that

K_bm^i = SHA1(TK_i, M),

where M = (previous CoA| HoA| TK_i-1 ). Note that the encryption key K_bm^i is derived from the previous CoA and the previous token of TK_i-1. Upon receiving a new BU message, CN can derived the encryption key used for the BU message from the previous CoA, HoA and the previous token of TK_i-1. This allows CN to be able to decrypt the BU message for the handoff.

When a connection between MN and CN lasts for a long period of time, all the tokens in a pool may be used. Depending on the likelihood spoofing, the same set of tokens may be reused or it may have to be replaced as soon as the entire pool is exhausted.

When a pool of tokens needs to be replaced, MN generates a new set of tokens and may send it in BU message. Since the new pool of tokens is sent over the network, it poses a certain degree of security concerns even though it is encrypted. One possible remedy is to embed the new tokens in randomly selected data messages as MN monitors how quickly tokens in the pool are being used. This approach continually replenishes used tokens and makes it sure that fresh tokens are always available when a handoff is to take place. However, MN and CN
needs to be synchronized to recognize tokens embedded in data messages.

3.2. Functionally Token Scheme

In the authentication pool scheme, MN generates a set of tokens, which is included in the BU message. When the size of pool is large, the size of BU message may be increase, resulting in slightly longer transmission time of the initial BU message. Moreover, the initial BU message containing the pool of tokens may be intercepted. Although BU message is encrypted, there may be concerns that it may be compromised while tokens in the pool are being used.

Functional token scheme addresses the concerns by adopting the one-time password mechanism in [10]. Let function $F(x)$ be a computationally inexpensive linear function. However, it is computationally difficult to invert $F(x)$. Some of hash functions have such property [10]. Consider repeatedly applying $F(x)$ recursively. Let $h_{1}(x) = F(x)$ and $h_{i}(x) = F(h_{i-1}(x)) = F^{i}(x)$, where $F^{i}(x)$ denotes $i$ successive applications of function $F(x)$.

From the property of $F(x)$, it is relatively straightforward to obtain $h_{1}$ from $h_{i-1}$, but not the other way around. The one-time password system first generates $\{h_{i}(x) | 1 \leq i \leq n\}$, and uses passwords in the backward sequence of $\{h_{i}\}$. Namely, the $k$-th password, $pw_{k}$, is given by

$$pw_{k} = \{h_{n-k}\}, \ 1 \leq k \leq n$$

When $n = 1000$, for example, the sequence of one-time passwords becomes $F^{1000}(x), F^{999}(x), \ldots, F^{2}(x), F(x)$. When a user logs in with the $i$-th password $pw_{i}$, the system applies the function $F(\cdot)$ to $pw_{i}$ and checks if it matches with the previous password $pw_{i-1}$. Namely, if $pw_{i-1} = F(pw_{i})$, the new password is verified. In order to verify a new password, the system is required to record only the last password used, which is substantially efficient than maintaining a table of passwords.

We adopt the one-time password system to creating a hash chain of authentication tokens. After a connection is established, both MN and CN obtain the identical $K_{bm}$ as in section 2. Now, MN and CN independently compute $h_{i}(K_{bm}) = F(h_{i-1}(K_{bm})) = F^{i}(K_{bm})$ for $1 \leq i \leq n$. MN keeps the entire table of $\{h_{i}(K_{bm}) | 1 \leq i \leq n\}$, whereas CN only keeps the last token, $h_{n}(K_{bm})$. At the first handoff, MN includes the 2nd from the last token, $h_{n-1}(K_{bm})$ in the BU message (which is encrypted by $K_{bm}$). When CN receives the BU message, it decrypts it with $K_{bm}$ and applies $F(\cdot)$ to the authentication token in the BU message to check if $F(h_{n-1}(K_{bm})) = h_{n}(K_{bm})$. If they check out, the authentication token in the BU message is indeed valid. CN then keeps the authentication token, $h_{n-1}(K_{bm})$ for the next round of handoff.

This method allows the authentication to be performed without exchanging or distributing information over network. Also, the amount of information required to be maintained at CN is kept to its minimum.

When MN and CN run out tokens, they can be replenished by generating $\{h_{i}(x) | 1 \leq i \leq n\}$ with a different seed for $x$. Generating a new seed value can be implemented in many different ways. One simple example is to combine the token kept in CN with $K_{bm}$, such that

$$x = h_{1} \oplus K_{bm},$$

assuming that the regeneration of tokens takes place when CN has the token of $h_{1}$. As in the initial connection establishment, MN and CN generate new $\{h_{i}(x) | 1 \leq i \leq n\}$ with the new seed, and MN keeps the entire sequence whereas CN keeps the last computed value. The approach eliminates needs for synchronizing when to replenish tokens and for distributing a new pool of tokens.
One possible weakness of the scheme is that each BU message is encrypted by the same binding key of $K_{bm}$. Even when $K_{bm}$ is exposed to an intruder, however, authentication token embedded in BU message provides additional level of security.

3.3. Performance

In Mobile IPv6, two round-trip delays are required at each handoff: a round-trip delay for exchanges of HOTI/HOT and COTI/COT messages, and another round-trip delay for BU message and its acknowledgement. Since HOTI and HOT messages have to be delivered via the home agent, it may take slightly longer than the other message exchanges. The two per-connection RR test schemes require two round-trip delays at the beginning of a connection between MN and CN, as in Mobile IPv6. However, all subsequent handoffs are performed with exchanges of BU messages only, thereby reducing handoff delay to a single round-trip delay.

4. Concluding Remarks

In mobile IPv6, an RR test is performed in each handover in order to authenticate the new CoA of mobile node. In this paper, two schemes are proposed to reduce the time required to authenticate the new CoA of MN. Both rely on generating a pool of authentication tokens at the initial connection establishment time. In each subsequent handoff, one of the tokens from the pool is used to authenticate MN in a BU message, so that a round trip required for a new RR test can be eliminated.

The two schemes differ in whether to exchange the pool of authentication keys or to dynamically verify the key in each handoff. MN may generate a pool of keys and transmit encrypted pool of keys to CN. Or, both MN and CN agree on a linear function with difficult inverse operation so that a pool of keys may be generated dynamically. Although more attractive, this scheme requires CN possesses some computational capability.

REFERENCES