Tested Evaluation of Fast and Secure Handover in FMIPv6

Shoaib Khan$^1$, K.K. Loo$^2$, and A.K. Kiani$^1$

$^1$School of Engineering and Design
Brunel University, Uxbridge, West London, UB8 3PH, UK

$^2$School of Engineering and Information Sciences
Middlesex University, Burroughs, London, NW4 4BT, UK

Abstract: In next generation all-IP based networks, a large number of mobile nodes are expected to use MIPv6 protocol for mobility. In mobile environment one frequent cause of service disruption is handovers. When MN moves, at some point MN will reach the physical border of its current wireless network and will enter an area covered by another wireless network. In such a situation, MN must switch to the new wireless network by handover. Fast handover for MIPv6 (FMIPv6) has been developed as an extension protocol to reduce the handover latency and packet loss inherent with MIPv6. The RFC4068 for FMIPv6 is categorized as “Experimental” by IETF because it is not in compliance with MIPv6 standard. One major compliance issue is that MIPv6 requires that all the messages exchanged between MN and HA should be secured. Adding security mechanism to MIPv6 without significant delay to the overall protocol performance is a challenge itself and it is recognized by IETF’s “mishop” working group. This paper presents a modified FMIPv6 which has the security mechanism in-line with the MIPv6 standard. The proposed FMIPv6 protocol is secured without introducing any significant handover latency reduction. This work was part of FP6 project ENABLE, and the actual testbed was presented in projects final demonstration meeting at Telecom Italia (T.I.) Labs in Turino.

Keywords: MIPv6, FMIPv6, Fast Handover, Secure

1. Introduction

In the vision of Next Generation Network, it has become mandatory for Mobile Node to constantly stay connected and seamlessly roam across different networks while enjoying the plethora of the ‘all-IP’ based services. A number of network layer mobility management solutions have been proposed. Amongst such candidate technologies, Mobile IPv6 has been widely accepted in the academics and industry. MIPv6 is standardized by the Internet Engineering Task Force as a viable option for delivering the required ubiquitous mobility services across an integrated heterogeneous network.

In this paper, one of the most frequent causes of disruption – the handovers is introduced. To author’s best knowledge, in IP based network, there is no “Make before Break” style handover. One protocol that comes close to making handovers seamless is Fast Handover for Mobile IPv6 (FMIPv6) [1].

In FMIPv6, static HA and HoA masks MN movements from CN. All packets sent to MN on its HoA are received by HA, which are then forwarded to MN current location using its CoA. However, when MN reaches the physical border of its current AP and enters an area covered by another AP, the MN must perform some standard procedure before regaining its connectivity. These procedures include scanning for suitable access point, switching access point, stateless address auto-configuration, Duplicate Address Detection (DAD), and finally updating binding on Home Agent. This sequence of procedures generally involves connection disruption and significant packet loss. FMIPv6 protocol is designed with the goal to reduce the handover latency and to assist MN in selecting a suitable Access Point. This protocol allows an AR to offer services to an MN in order to anticipate the L3 handovers. This anticipation is based on L2 triggers. The procedures of FMIPv6 are explained with help of a signaling diagram shown in Figure 1.

Figure 1: Standard FMIPv6 Signaling

FMIPv6 protocol enables an MN to request information on neighboring AP’s and the subnets behind them. To do this, MN sends a “Router Solicitation for Proxy Advertisement (RtSolPr)” message. This solicitation may contain the ID of one or more APs, thus requesting subnet information corresponding to the AP. It may also contain a wildcard signifying a request for all nearby ARs. In Figure 1, when RtSolPr is received by the old access router, it replies with Proxy Router Advertisement (PrRtAdv). When MN detects that a handover is required, it sends a Fast Binding Update (FBU) to its current router (pAR). This message contains MNs’ CoA in pAR network (pCoA) and the access router
(nAR) that MN is planning to switch to. At this point pAR sends a “Handover Initiate” (HI) containing link layer address of MN and pCoA. MN can optionally include nCoA. The nAR confirms the handover with a Handover Acknowledge (HACK) message that may provide further nAR specific details. Once the HACK is received, pAR sends a Fast Binding Acknowledgement (FBAck) back to the MN, on pAR’s link. At this stage, MN is ready to switch AP. After MN moves to nAR, it sends a Fast Neighbor Advertisement (FNA) message and completes handover signaling. This type of handover, characterized by the FBAck received by the MN while it is in pARs networks, is called by the FMIPv6 a predictive handover. FMIPv6 protocol also defines a reactive handover scenario which basically represents the case where MN cannot predict a handover. In this case the FBU is sent from nAR’s link after L2 handover is completed. It is usually encapsulated in the FNA. The nAR then forwards the FBU message to pAR, which is followed by HA/HAck messages.

In FP6 project ENABLE, FMIPv6 was selected as prime protocol for handovers. A major problem in FMIPv6 is the un-secure design of the protocol. We, as partners, were assigned the task of securing FMIPv6. The details of which are discussed in Section 3 of this paper. It was also a requirement to analyze if the added security will induce any delays to FMIPv6 protocol. For this analysis, a comparison with non-secure FMIPv6 implementation is required for which a review of existing FMIPv6 implementation is presented in next section.

2. Literature Review

In this section some of the most recent and relevant work in the field of FMIPv6 deployment and implementation is discussed. In addition to this, some recent work related to optimal selection of AP during a handover is also presented.

In [2], the author is the actual developer of FMIPv6 protocol for Linux platform. In this paper, the performance evaluation of FMIPv6 over 802.11 WLAN is very comprehensive and detailed. For comparison of secure FMIPv6, this scheme was selected as benchmark. The developed testbed is almost the exact replica of what is suggested on fmipv6.org. Similar testbed was used by author for his paper. Author achieved a handover latency of 12.4ms with buffering and 42.7ms with buffering disabled on AR, in case of a predictive handover. In case of reactive handover, handover latency is about 2.45 seconds.

In [3], a proprietary implementation is evaluated. Experimentation involves wired handovers with emulated delay of 40ms. This delay was configured for handover. These values were chosen as they are closer to that of 3G systems. Similar to handover delays, link layer trigger are preconfigured which is again not a good approach. Triggers like link down were instant in a wired network but this is not the case for 802.11 based networks.

Another performance analysis of FMIPv6 is provided in [4]. Author’s focus is on protocol overhead, wrong anticipation and buffer limitation on AR. They have shown that these vary largely depending on how close in terms of time the link layer trigger and actual link disruption is. They computed an optimal separation between them. They also studied overhead which depends on application used. If you send larger packets as in the case of VoIP or Video on demand, percentage overheads would be very low as compared to a simple web browsing data.

The [5] also designs proprietary implementation of FMIPv6. Testbed was setup in back in 2001 when FMIPv6 RFC was not finalized, but this implementation has significant impact in IETF work. This experimental setup was a bit un-realistic and has been criticized in a number of papers. The procedures followed were also not appropriate. For example, L2 trigger were generated by command line so they were not actual triggers but just emulated signals. They also neglected candidate AP discovery which has a huge impact on overall handover latency.

All of the above implementations did not consider security issues of handover. Our goal was to make it secure. The architecture of secure FMIPv6 is discussed in next section.

3. Architecture Overview of Secure FMIPv6

For security of FMIPv6 based handover, in IETF discussion it was decided to secure the Fast Binding Update (FBU) message between AR and MN. This is to protect against certain type of attacks. In the ENABLE project [6], this is exactly what was implemented. In the Figure 2, illustrates all the logical components that are required for securing FMIPv6.

In the ENABLE architecture, FMIPv6 is considered as premium service and is required to be authorized by either Mobility Service Provider (MSP) or Access Service Provider (ASP). In reference to Figure 2 for service authorization, either SP or BCA-p interface is used. SP interface is considered out of scope of ENABLE project, as it is application specific. BCA-p interface is used for Mobility service authorization. For Mobility service authorization, MN will present a certificate issued by MSA/ASA. Once MN is authorized either by MSA-AAA or by ASA-AAA server (called GSABA in ENABLE), MN is permitted to use MIPv6 service.

After the FMIPv6 service authorization, Access Router/Access Points can authenticate MN based of Shared Keys.
3.1 Component Description

In this sub-section, the components required for deployment of secure FMIPv6 are introduced.

- GSABA Server: This is responsible to facilitate authorization and authentication. Its details structure can found in ENABLE Project deliverable ZZ.
- pAR and nAR: These are service point entitles. In our case these are FMIPv6 enable access routers.
- MN: This is FMIPv6 enable Mobile node.
- BCA: It is responsible for providing necessary bootstrapping information to the MN. For FMIPv6, this information would be the IP routable address of the pAR. In our case it was co-located within GSABA.

The BAA asserts the authorization statements. Based on the MN’s profile, which is available in the authorizing domain, such statements are generated by the BAA. These statements and parameters need to be conveyed to the MN. In addition the BAA needs to authenticate the MN.

3.2 Interface Description

In this sub-section, the logical interfaces required for the secure FMIPv6 implementation are introduced.

- The Bootstrapping Target Protocol (Tp-p): This protocol is used between the BTs (i.e. pAR and nAR) and the BCA in the GSABA Proxy/GSABA Server to exchange FMIPv6 service related information and authorize the BT to provide FMIPv6 service to the MN. This information could be encoded in Attribute Value Pair (AVPs) or in XML. For implementation AVPs were used.
- The Bootstrapping Protocol (BCA-p): This protocol is used to convey bootstrapping information to the MN and inform the authorization decision taken by the BAA and BAA Proxy. The information could be encoded in AVPs or in XML. Candidate transport protocols are EAP/PANA, DHCP, HTTP, and TLS.
- The Bootstrapping Agent protocol: This is used between the BAA Proxy and BAA to exchange information between the BAA entities to base decisions on and to deliver them to the BCA. The information could be encoded in AVPs or in XML.
- The Service Related Protocol (SP): This protocol runs between the MN and BT (i.e. the pAR and nAR). Ideally, this is a service specific protocol (i.e. FMIPv6 signaling in this case) and should be left largely unmodified if the interface is therefore indicated in a dashed line in Figure 2.

3.3 Message Flow

In this sub-section, detailed message flow of the overall GSABA architecture with FMIPv6 is presented. The aim is to show how different entities interact to provide the service bootstrapping. This message flow is divided into two parts. First part describes signaling involved during FMIPv6 service initiation and the second part describes signaling involved during a handover.

Assumptions made for this message flow are mention below.

- Mobility service provider is capable of providing FMIPv6 server.
- The GSABA Server is in the service authorization domain and can make authorization decisions.
- MN has already completed bootstrapping and has successfully connected to one of the Access Point.

3.3.1 Message Involved during FMIPv6 Service Authorization Phase

This sub-section presents details of message flow of the overall GSABA architecture with FMIPv6 as shown in Figure 3.

![Figure 3: Mobility Service Authorization](image-url)
• GSABA Address Discovery: The MN authenticates with Mobility Service Authorizer (i.e. the home domain which authorizes the network access) in order to get authorized for network access. (This is done completely independent of the FMIPv6 service bootstrapping). It is assumed that the initial network access authentication uses EAP based authentication. At this stage MN know GSABA server address.

• Key Generation: During bootstrapping a new key is generated by the MN and the AAA server, called GSABA key that might be derived from EMSK generated after a successful EAP method authentication (some guidelines for further key derivation by using EMSK as a root key. This key is used for the protection of the communication between the MN and the GSABA AAA server (i.e., the BCA-p interface).

• BCID Generation: the MN and the GSABA AAA generate a new identifier called “Bootstrapping Client IDentifier” (BCID).

• ABIREQ Message: Services request done via the ABIREQ (Authorization and Bootstrapping Information Request) message. The minimum set of parameters included in the ABIREQ is the BCID, the identifier of the service that the MN wants to use (Service Request ID – SRID), and the corresponding identifier intended to be used on the SP interface (Service ID – SID). The SRID could either be a service name alone or, additionally a bootstrapping identifier (SID) to be used for accessing the service.

• ABIRES Message: Upon receiving the ABIREQ message, the GSABA Proxy sends an ABIRES message with the authorization decision and minimum information conveyed to the MN about the service points (e.g. the IP address or FQDN of pAR). Additionally, the ABIRES can contain a key and an identifier (SID) to be used for accessing the service.

• HK Key Derivation: With the information provided in the ABIRES, The FMIP service is bootstrapped as usual. For securing FMIPv6 signaling messages, a handover key (HK) is derived from the GSABA key shared by MN and GSABA proxy. Using the HOKEY process, discussed in detail in the next section.

3.3.2 Message Involved during FMIPv6 Handover

In this sub-section the actual message flow for a FMIPv6 handover is described. This flow is shown in Figure 4.

• Step 1: Detection of new AR: MN performs periodic wireless scans. With L2 trigger, MN starts looking for a candidate access router.

• Step 2: RtSolPr: At this point MN send a RtSolPr to pAR to resolve one or more Access Point Identifiers for subnet-specific information

• Step 3: PrRtAdv: In reply to PrRtAdv, pAR send PrRtAdv message to MN. This message contains subnet-specific information.

• Step 4: nCoA Creation: From Subnet specific information about neighboring APs, MN creates nCoA.

• Step 5: HKReq: Using this nCoA, MN generates HKReq Message. This message is sent to nAR1. Aim of this message to set up security association with candidate access router. In enable project, it was decided that MN should establish Security Association with all available candidate access router before the actual switch.

• Step 6: nCoA Check: After HKreq is received by nAR1, it should check if this newly created address (nCoA) can be used. If yes, then it proceeds to next step.

• Step 7: SIReq: After having received HKReq and nCoA validation, it forwards this HKReq message to GSABA using SIReq message.

• Step 8-9: HK and SIRsp: The GSABA checks the MAC contained in the HKReq from the MN and if successful, derives a new handover key for new candidate access router and returns SIRsp message including the result code and new handover key as well as AAA nonce. This SIRsp message is sent to nAR1.

• Step 10-12: HKRsp: Handover key is contained in SIRsp message. This key would be used to authenticate MN during a fast handover. Once this key is known, nAR send the HKRes to MN.

• Step 13: Selection of nAR2: By now MN has established security association with all the candidate routers and is ready to switch to any of the available candidate access router. Final selection of candidate Access router is based on L2 trigger.

• Step 14: FBU: Once the candidate access router is selected, MN send FBU message to pAR. These messages are already secure by previous handover key (pHK).

• Step 15: HI Message: This is a standard FMIP message, indicating that MN is about to Initiate handover. This message is send to nAR by pAR.

• Step 16: HAck message: If nAR is ready to accept connection from MN, it will acknowledge the HI message with HAck message.

• Step 17: FBAck message: Once the pAR has received acknowledgment of HI message, it sends FBAck to MN. Again, this message is secure by pHK.

• Step 18: L2 Switch: At this stage, MN performs actual L2 switch by changing its active access point to nAR.

• Step 19: Once this step is complete, FNA message to nAR. This message is protected by handover key that was created in step 12.

• After step 19, the handover is complete.
3.4 FMIPv6 and HOKEY

The current Extensible Authentication Protocol (EAP) keying framework is not designed to support re-authentication and handovers without re-executing an EAP method. So when MN moves from one Access Point to another, MN requires re-executing the network authentication procedures. This often causes unacceptable delays and latencies. IETF working group “hokey” was created to design a mechanism where this re-authentication can be performed without all the latencies and delays. In standard FMIPv6 there is no concept of authentication; the assumption is that all wireless networks have open access. In commercial deployment, it is not feasible for MSP/ASP to provide services with open access network. So in the standard form, FMIPv6 is not deployable for MSP/ASP. In ENABLE framework, it is mandatory for MN to re-authenticate itself after each handover. To resolve the deplorable delays created by EAP based re-authentication, HOKEY re-authentication procedures are followed. In this sub-section the Handover Keying (HOKEY) re-authentication procedures are introduced.

In HOKEY, handover master key (HMK) is pre-shared between MN and AAA and should not be used to protect any data. Handover key (HK) and handover integrity key (HIK) are derived from HMK. HK is used to secure the messages exchanged between MN and AR while HIK is used to secure the signaling between the MN and AAA.

In GSABA, GSABA key is used as the HMK to derive the HK as well as the HIK. The HIK is used to protect the data between MN and GSABA proxy (i.e. ABE/RES/ABIREQ message) via the BCA-p interface and therefore is in fact the BCA Key. The HK can be derived by:
- \( HK = \text{gprf} + (\text{HMK}, \text{MN Nonce} \mid \text{AAA Nonce} \mid \text{MN ID} \mid \text{AR ID} \mid \text{"Handover Key"}) \), or
- \( HK' = \text{gprf} + (\text{HMK}, \text{MN Nonce} \mid \text{MN ID} \mid \text{AR ID} \mid \text{"Handover Key"}) \)

The gprf+ is used to derive the HK (where ‘\(|\)’ indicates concatenation), which is an ASCII string with 12-characters and no null termination. The MN Nonce is generated by the MN and communicated to the AAA server in the HKReq message. The AAA nonce is generated by the AAA server and sent to the MN in the HKRsp message. The MN ID is the NAI of the MN and the AR ID is the IP address of the AR as seen by the MN.

HK can be derived after MN has attached to an AR and HK’ can be generated during handover procedure. Thus the probability for handover error induced by security mechanism will be reduced, because it is possible for MN to leave its old link without FBU message.

There are many advantages for setting up a SA between the MN and the AR before the actual fast handover process:
- It allows leveraging the AAA infrastructure that is already in place to establish session keys for securing FMIPv6 signaling messages.
- The handover key derivation does not impact handoff latency.
- The compromise of one AR or a particular handover key does not lead to the compromise of keys shared between the MN and any other AR.

3.5 AAA and FMIPv6

To support the FMIPv6 service authorization across different
network domains there is a need for an entity, which can relay authorization messages across different network domains. Almost all Telecom Provider (POTS providers), Internet Service Providers (ISPs) and Internet Telephony Service Providers (ITSP) make use of the Authentication, Authorization. To support roaming between different access domains, AAA broker services have been deployed to accomplish peering of various providers. Such agreements symbolize business relations and have an impact on AAA message routing. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure reduces operational and deployment cost. Due to this reason, ENABLE architecture leverages and integrates with the existing AAA infrastructure.

The GSABA server contains BAA component and is responsible for making the ultimate decision of the authorizing the requested service (i.e. FMIPv6 in this case). The GSABA service will reside in the service-authorizing domain (which is typically in the MN’s home domain).

4. Experiment Test-Bed Evaluation

The testbed built for secure FMIP is illustrated in Figure 5. There are two access routers (AR), a mobile node (MN), a corresponding node (CN), Home Agent (HA) and Authentication and Authorization server.

![Figure 5: FMIPv6 Testbed](image)

4.1 Detailed Description of Nodes

The ARs were based on highly optimized Linux machines with Atheros chipset. Latest Linux kernel 2.6.23-rc3 available at that time was used. For Mobile IP, the latest developed Universal Mobile IP (UMIP) package was used. For routing Quagga Routing Suite was used. The actual routing protocol used was RIPv3. Access routers were running the modified implementation of fmip-ar daemon developed by fmipv6.org. For route advertisement, Radvd was used.

The MN was Linux based machine with Atheros chipset. Again Linux kernel 2.6.23-rc3 was used. MN was also running UMIP and the modified implementation of fmip-mn daemon.

The HA was Linux based machine with one Ethernet card. As with other nodes, kernel version 2.6.23-rc3 was used with UMP. For route advertisement, Radvd was used. For routing, Quagga Routing Suite is used. The actual routing protocol used was RIPv3.

The Authentication/Authorization server was again Linux machine. Software running on this machine were Apache (web server) with SSL support, modified FreeRadius server and Quagga routing daemon.

In Table 1, the exact specification of the each node used in testbed is described.

<table>
<thead>
<tr>
<th>Network Entity</th>
<th>Hardware</th>
<th>OS</th>
<th>WLAN card</th>
</tr>
</thead>
<tbody>
<tr>
<td>HA</td>
<td>Intel Core 2 Duo desktop</td>
<td>Linux kernel 2.6.23-rc3</td>
<td>N/A</td>
</tr>
<tr>
<td>pAR</td>
<td>Sony VAIO Laptop</td>
<td>Linux kernel 2.6.23-rc3</td>
<td>Atheros chipset</td>
</tr>
<tr>
<td>nAR</td>
<td>Siemens Laptop</td>
<td>Linux kernel 2.6.23-rc3</td>
<td>Atheros chipset</td>
</tr>
<tr>
<td>MN</td>
<td>IBM T43p Laptop</td>
<td>Linux kernel 2.6.23-rc3</td>
<td>Atheros chipset</td>
</tr>
<tr>
<td>CN</td>
<td>Intel Core 2 Duo desktop</td>
<td>Linux kernel 2.6.23-rc3</td>
<td>N/A</td>
</tr>
<tr>
<td>GSABA</td>
<td>Intel Core 2 Duo desktop</td>
<td>Linux kernel 2.6.23-rc3</td>
<td>N/A</td>
</tr>
</tbody>
</table>

4.2 Test Description

This section discusses the objectives of all the tests performed, their expected results and comparison of actual and expected results. The main objective of this testbed was to show that by adding security to FMIP, there would be no substantial increase in handover delay as compared to standard FMIP protocol. The results are presented on graphs. Exact time measurements were based on Timestamps displayed on Linux console provided by FMIP suit when it was running in debug mode along with a packet-sniffing tool, Wireshark.

Two scenarios for a mobile node to perform handover, either Reactive or Predictive discussed as follows:

- Reactive Handovers: MN performs a Reactive handover when it is not able to predicted handover because of sudden drop/loss of signal from its attached AR. In such a handover, a long service disruption time is expected because MN would have to perform scanning and after completion of scanning it would have to attach to one of the available access point. The scanning process takes about 1.3seconds-5seconds depending on available channels and access points. For all the reactive handover testing mobile node was forced to change its point of attachment.
from one AR to another by shutting down the wireless interfaces of the pAR.

- Predictive Handovers: A predictive handover is performed when a MN is able to anticipate a handover. This predication is triggered when Signal strength of an associated link drops below a certain threshold. In such a situation, MN moves to AR with highest signal strength from the last scan. For all predictive handover testing, MN was attached to one of the access router and then reducing the Tx power of this access router, forcing it to move to nAR.

In both scenarios, MN was streaming video from CN based on RTP protocol. RTP transport protocol was used to avoid the TCP re-transmissions that could affect the actual results. Buffering was enabled on Video client to be able to perceive the handovers in the actual streaming. WireShark collected the data on MN. Timestamps were generated by FMIPv6 daemon.

4.3 Reactive Handover Test Results

In the testing, the MN was connected to one of the AR and then while MN was streaming the video, the wireless interface of that AR was switched off. MADWiFi drivers detected the link failure through missing beacons. Default value set in drivers is 7 beacons. In 802.11 the beacon interval is set to 100ms. So the device drivers detected the link loss after 700ms. The FMIPv6 daemon detected the link lost after another 350 ms and then it initiated the L2 handover. If FMIPv6 daemon had already completed the scan, it tried to connect to AR with highest signal strength. If scanning was not preformed before the link loss then FMIPv6 daemon would start scanning. This scanning time depends on a number of factors which will be discussed later. Finally, when L2 link is up, BU is sent to HA. This time again varies from 1ms-9ms.

Figure 6 shows the overall delay of reactive handover. This illustrates different stages of handover and the time each stage took. Total time for handover varies between 2.5-4 second. In this figure, the time between when pAR interface was turn off to the time when link loss was declared MADWiFi driver was about 700ms. This link lost was detected by FMIPv6 daemon in about 350ms. Once this was detected by FMIPv6 daemon it starts scanning. This scanning time depends on a number of factors which will be discussed later. Finally, when L2 link is up, BU is sent to HA. This time again varies from 1ms-9ms.

Table 2: Testbed Results

<table>
<thead>
<tr>
<th>Latency (ms)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link Loss</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Declaration by MADWiFi</td>
<td>785</td>
<td>734</td>
<td>789</td>
<td>738</td>
<td>721</td>
<td>753.4</td>
</tr>
<tr>
<td>Link Loss Detection by FMIPv6</td>
<td>351</td>
<td>362</td>
<td>373</td>
<td>341</td>
<td>336</td>
<td>352.6</td>
</tr>
<tr>
<td>L2 Scan time</td>
<td>2531</td>
<td>262</td>
<td>181</td>
<td>312</td>
<td>4</td>
<td>342</td>
</tr>
<tr>
<td>BU sent to HA</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>8</td>
<td>4.8</td>
</tr>
<tr>
<td>Total Latency</td>
<td>3669</td>
<td>372</td>
<td>297</td>
<td>421</td>
<td>448</td>
<td>3812 .</td>
</tr>
</tbody>
</table>

This experiment was repeated multiple times and the timestamps are listed in Table 2. In this table average L2 scan-time is 2.7 second. This is because all the available channels of 802.11a/b/g card were scanned. If all the channels of 802.11b were scanned then this scan time would be around 400ms. This could significantly reduce the handover latency.

4.4 Predictive Handover Test Results

The predictive handover of FMIP is the most desirable part of the protocol. Handovers performed in this mode are almost seamless with minimum packet loss and hence minimum service disruption time. Video streaming application was tested and there was just a glitch in the stream.

In predictive mode, MN performed regular scans and whenever it detected that the SNR threshold has been reached it did a handover. The signaling diagram of a predictive style handover is illustrated in Figure 4. The exact steps were followed during actual testing. First MN sent the HKreq to pAR. On receiving this HKreq, pAR sent SIReq message to Authentication Server which replied with SIres. Once this message was received by pAR it sent an HKRes message to MN. Using Ethernet link between two ARs, MN sent HKreq to nAR. On receiving this message, nAR sent SIReq to Authentication Server which was acknowledged with SIres. Once this SIres message was received by nAR, it sent HKRes message to MN using the Ethernet link between nAR and pAR. While MN is connected to pAR it sent an FBU message to pAR. Next, pAR sent a HI message to nAR. In response to HI message HAck message was sent to pAR and this was acknowledged with FBAck message by pAR. At this point MN actually established the L2 connectivity with nAR which takes about 0.8-1.1 second. In this table average L2 scan-time is 2.7 second. This is because all the available channels of 802.11a/b/g card were scanned. If all the channels of 802.11b were scanned then this scan time would be around 400ms. This could significantly reduce the handover latency.
16ms. The final message sent by MN is FNA to nAR.

Figure 7 shows the results of the Predictive handover. It takes about 16ms for a handover complete. This figure represents the testbed where MN has either two interfaces, one for scanning and one for actual communication or FMIPv6 daemon has already performed the scanning.

In case, if MN is not a dual interface or if it has not performed scanning already then there would be an added scanning time to this graphs, which would be about 1.3 second.

Figure 7: Handover Delay for Predictive Mode

4.5 Impact of Scanning on Handover
FMIPv6 does not provide a way for MN to discover candidate access router. Sending an RtSolPr to pAR only request the MAC address of Candidate access router. It doesn’t tell MN if it is in the range of any of the Candidate access route. The only way for MN to discover a suitable AR is to perform periodic scanning.

During the tests it is observed that the scanning process takes about 1.5-3.5 second depending on the type of wireless card used and the scan procedures. Two different types of cards were tested. One was capable of 802.11a/b/g and other was capable of 802.11b/g. For the first one, scanning process would normally take up 15.2 seconds and the other one would take about 2.7 seconds. After modifying MadWiFi drivers and setting the MaxChannelTime, scan time was decreased to 4.7s and 1.3 seconds.

In case of reactive handover, if the MN has not previously performed scanning, after receiving Slres from pAR, the wireless card would go in a scanning phase. During scanning, there is a significant packet loss in case of heavy traffic. For testing video streaming was used and whenever MN would perform a scan, there was either a pause or glitch in the video. On way to resolve this issue is to have two wireless cards in MN. One wireless for scanning and one for actual data communication but this is not an optimal solution.

4.6 Comparison of FMIPv6 with Secure FMIPv6
In [1], authors have achieved a handover time of 10.42 ms for a predictive mode. On our testbed the standard time for same scenario was 14.2ms. Reason for this delay is from fact that author did modified wireless card’s firmware which was not done for this testbed. Our secure FMIPv6 predictive handover had a latency of about 16ms. The delay was mainly because all the messages exchange was secure. The secure message creation and its validation on the other end require some computations, which takes time. Overall, the overhead for added security is 1.8ms. If the firmware was modified, these results may drop to 12.22ms.

With just 1.8ms overhead it have been proved that FMIPv6 can be secured without adding any significant delay. Even for most demanding application, this 1.8ms delay would not be a problem at all.

5. Conclusion
In the paper, it has been proved that adding security to FMIPv6 will not affect the FMIPv6 in any possible way. It is still fast and secure. During our experimentation on actual testbed, a seamless handover is observed despite all this added security. In case of reactive handover, the latency was between 2.5-4 second. Major contribution to this delay was from the scanning time, which was between 1.8-3.4 seconds. In case of predictive handover, this time was about 16-17ms without including the scanning phase.

There is one major issue in protocol, which is unaddressed. This includes Candidate Access Router discover. The only way to discover Candidate Access Router is by scanning. This scanning time is the longest delay in overall handover. One possible approach that was adopted is to use two wireless interfaces on MN; one interface for scanning and one for actual data communication. Another possible solution could be to use 802.21 information sever for Candidate Access Router discover.

References