FI-Based Local Group Key Generation/Distribution for Mobile Multicast in a Hierarchical Mobile IPv6 Network

Jinsuk Baek, Paul S. Fisher, and Mingyung Kwak, *Members, KSII*
Department of Computer Science, Winston-Salem State University,
601 M. L. King Jr Dr, Winston-Salem, NC 27110, USA
[e-mail: baekj@wssu,edu, fisherp@wssu.edu, mkwak106@wssu.edu]
*Corresponding author: Jinsuk Baek

Received November 29, 2007; revised January 11, 2008; accepted January 14, 2008;
published February 25, 2008

Abstract

In order to securely transmit multicast data packets in a mobile environment where frequent join/leave events are a characteristic of the environment, there is a need for a new secure and efficient group key management solution. We propose a secure group key generation/distribution solution providing scalability and reliability. Using this solution, when a mobile node, which is in a multicast session, enters a new domain, the agent of the domain joins the multicast session and coordinates its data packets with the mobile node. The agent encrypts and transmits subsequent data packets to the mobile node, using a local one-time pad key. This key is generated with FI sequences, enabling the mobile node to regenerate the same data packet, based on the information sent by the agent. Our performance analysis demonstrates that the proposed solution can significantly reduce the number of key generations and distributions, when it is applied to the hierarchical mobile IPv6 network.

Keywords: Multicast, Group key, HMIPv6, FI sequences, One-time pad key

Digital Object Identifier 10.3837/tiis.2008.01.001

1. Introduction

Recently, an increasing number of mobile network applications, including Satellite TV, software distribution, stock quote streaming, multimedia conferencing, and Web cache update, require a sender to transmit the same data to a large group of receivers. These applications are classified as group communication, by contrast with traditional one-to-one communication. Multicasting, that is, the delivery of a single message to multiple recipients using the same IP address, is the only efficient and scalable solution which can support this kind of application.

The mobile multicast service can be supported by Mobile IP protocols [1][2][3][4][5][6] with two different approaches – namely, Bi-directional Tunneled multicast (MIP-BT) and Remote Subscription (MIP-RS) [1]. In order to facilitate mobile roaming between different wireless or wired access networks, Mobile IPv4 (MIPv4) [1][4][5][7] involves specialized routers called Foreign Agents (FA's) and Home Agents (HA's), to perform the registration of a mobile node (MN). On the other hand, Mobile IPv6 (MIPv6) [2] does not require dedicated routers to act as FA's. Instead, a MN can maintain its connection through the access routers (AR's). Both protocols enable a MN to maintain a permanent IP address, called the home address, while visiting different networks. While the MN is away from its home network, the HA redirects packets to the MN. Whenever an MN changes its point of attachment, both protocols require the MN to update its HA of its new location and all correspondent nodes (CN's) communicating with the MN. Even if the MN roams between subnets of the same domain, the MN has to send a binding update (BU) to its HA, which usually resides far away from the MN.

To reduce this signaling overhead caused by the handover process, the concept of Mobility Anchor Point (MAP) is introduced by the Hierarchical MIPv6 (HMIPv6) protocol [20]. Within large scale network infrastructures, it is common for several MAP's to coexist, to improve robustness and enable traffic sharing. An MAP serves as a local HA in a foreign network. Whenever an MN moves to a new subnet within the same MAP domain, it sends a BU to the MAP, instead of the HA. This significantly reduces the signaling overhead, since the MAP is usually much closer to the MN than the HA.

In addition to a fixed/wired environment, mobile wireless communication involves the processes of AAA (Authentication, Authorization, and Accounting) resolution. The associated issue is the determination of authenticity when the MN changes its point of attachment. The AAA resolution process can be performed for all the above mentioned protocols and there are multiple AAA servers associated with agent routers, which are implemented at a different level of the router hierarchy.

Another complicated and difficult issue in mobile multicasting is that of providing a good security mechanism for the ongoing multicast session. The mobile IP multicast does not manage multicast group membership in a secure way, since it only uses a 'best effort' mechanism. In a typical multicast environment, all group members should share the same group key. Hence, any host that has the multicast group key can receive a copy of the packet, because the packet is automatically duplicated at the routers.

Because of the nature of the mobile multicast environment, the join/leave events can occur more frequently. Hence, the secure multicast schemes [8][9][10] must satisfy some additional requirements. Firstly, they must support dynamic join/leave events, with a reasonable message overhead, because the group key should be changed as a result of every membership revocation. Additionally, if a new MN joins a multicast group, the group key is changed to prevent the new MN from decrypting the data that was transmitted prior to the join. This property is called backward secrecy. Also, if a MN leaves a multicast group, the group key should be changed. Otherwise, the MN leaving can continue to decrypt data using the previous group key. This property is called forward secrecy. These requirements must be addressed in a scalable manner, to be applicable to large group sizes.

In order to deal with these issues, many group key management solutions have been proposed. Currently, those protocols can be broadly classified into three categories – namely, Centralized, Decentralized, and Distributed Group Key Management Protocols. In centralized protocols [11], a single server is dedicated to managing the entire group. However, the frequent group key generation and distribution properties do not enable these approaches to be applied in a mobile wireless environment, because they result in a high workload for the central key distributor. In addition, the HA has to handle considerable traffic, in terms of overhead, to manage the mobility of the MN's, including AAA resolution. Also, it can easily suffer from feedback (ACK or NAK) implosion caused by its MN's requesting retransmission of any lost data packets. Hence, any robust solution must reduce the bottleneck at the HA for this critical activity. Decentralized protocols [12] delegate the group control to multiple sub-groups. Due to the lack of a general group key, the real-time multicast data need to be decrypted and re-encrypted by each subgroup, which impedes the performance of real-time multicasting. In distributed group key management protocols [13], there is no explicit key distribution center, and each group member can contribute to the key generation and distribution. Unfortunately, many of those protocols require many rounds of messaging to update a new group key. As a result, they are only practical for applications involving small group of multicast receivers.

We propose an efficient decentralized solution which resolves all of the above mentioned issues. Our goal is to provide a secure, scalable and reliable multicast service in an HMIPv6 network. Under the proposed protocol, each MAP router participates in the multicast session, to maintain a secure session for its MN's. This enables distribution of the tasks to the end points acting as a local server for its MN's. When the MN enters a new MAP domain, the MAP of the domain performs an AAA resolution for the MN. Once it is approved, the MAP joins the multicast session. After that, the MAP and the new MN coordinate their most recently received data packets. The MAP now encrypts and transmits the next data packets to the MN. This encryption process is performed by using a one-time pad key. The performance of the proposed solution depends on how efficiently the MN's in the domain independently regenerate the original data packet without any assistance from the MAP. In order to handle this issue, we consider a Finite Inductive (FI) system [14][15] invented by one of the authors. The FI system enables each MN to regenerate the same data packet based on the information sent by their MAP.

Our solution has many advantages over the existing traditional solutions. We believe that our FI-based approach can be applied to the HMIPv6 infrastructure, including MAP's and ARs. Firstly, a leave event does not affect the entire multicast group, in terms of traffic overhead and a new key generation/distribution, because it is not necessary to change the global group key. Secondly, the key management tasks can be delegated among the multiple MAP's, and the local group keys can be efficiently generated and managed at the MAP's by using the FI system. In addition, the transmission error recovery task can be performed at the MAP's, by requiring them to allocate some buffer space in order to store the recently received data packets that can later be retransmitted to its MN's. This approach enables the MAP and its MN's to have the same data packets. It must be noted that this decentralism enables us to achieve one of the objectives, that of a scalable multicast server.

2. Preliminaries

In this section, firstly, we describe the basic operations of HMIPv6 network; secondly, we provide a brief review of how the multicast service can be supported in Mobile IP networks; and finally we discuss some cryptography techniques that can be applied to group key generation.

2.1 Hierarchical Mobile IPv6

To provide continuous network coverage for MN's, these MN's should remain connected to the network, regardless of their location. This requirement creates a conflict between two mobility objectives. First, an MN should change its IP address in order to enable correct packet routing. At the same time, it cannot change its IP address without breaking all its existing connections. Mobile IP solves these mobility problems by using two IP addresses: a *permanent home address* assigned at the home network, and a *temporary care-of address* (CoA), representing the current location of the MN. Whenever an MN obtains the new IP address from a foreign network, the binding between the two addresses should be transparently maintained.

In MIPv4, there are two specialized routers known as mobility agents that maintain this mobility binding and tunneling; the home agent (HA) in the home network, and the foreign agent (FA) in the visited network. MIPv4 supports a seamless handoff for MN's within the scope of *unicast* delivery, through the cooperative support of these two agents. MIPv6 has been proposed to compensate for the lack of available IP addresses and eliminate several other disadvantages of MIPv4. It does not require special routers to act as FA's. Enhanced features like neighbor discovery and address auto-configuration enable the MN to function in any IPv6 network environment. On the other hand, MIPv6 is sufficiently extensible to support unforeseen future needs, by the introduction of extension headers. Further improvements are included for the support of route optimization and a lowering of routing bandwidth overhead. AAA procedures can be performed based on the DIAMETER extension for MIPv6 protocol.

Unfortunately, both protocols have a major disadvantage, due to their high handoff latency. Every time an MN moves into a new FA or AR subnet, a sequence of signaling occurs. The MN configures its new IP address and updates its HA and CN, by sending BU messages. HMIPv6 establishes the concept of MAP to reduce the signaling overhead during the handoff procedure. An MAP exempts the MN from having to transmit expensive BUs to its HA and CN, provided its movement is limited to the same MAP domain. The MAP acts as a local HA within the visited domain. It can be implemented at any level in a hierarchical network of routers, including the AR. Whenever a MN moves to a new subnet within the domain of its associated MAP, it sends a BU to the MAP, instead of the HA. Because the HA is typically further away than the local MAP, the handover overhead is reduced significantly.

Two new IP addresses are established to employ HMIPv6: A *Regional Care-of Address* (RCoA) and a *Local Care-of Address* (LCoA). The RCoA is an address in the MAP's domain. The LCoA is the on-link address configured in a MN's interface, based on the prefix advertised by its default AR. When an MN enters a new administrative domain it is informed about the presence of MAP's by collecting *Router Advertisement* messages from the AR. Once the MN has selected the best fitting MAP using MAP selection algorithms [16][17][18], it sends to the chosen MAP, a local BU containing its RCoA and LCoA. When the MAP accepts the binding request, it will create a binding by storing the IP addresses in its binding cache and answer with a *Binding Acknowledgement* (BA). After the MN receives the BA of its MAP, it sends a BU to its HA, containing its RCoA. Following a successful registration, a bi-directional tunnel between the HA and the MAP is established. All packets sent by the MN will be tunneled to the MAP. All packets addressed to the MN are intercepted by the HA and forwarded to the MN's RCoA. The MAP will intercept the packets and route them to the MN by using the corresponding LCoA.

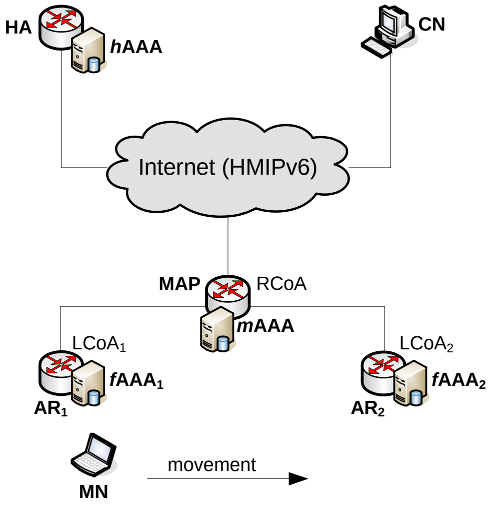

Fig. 1. AAA Architecture for HMIPv6

The main advantage of HMIPv6 is the fact that a MN does not have to send a BU to its HA whenever it moves to another subnet within the same MAP domain. Because the MN will still be bound to the same MAP, its RCoA is unmodified and only its LCoA is changed. Therefore, the MN must send a BU to its MAP. There is no need to update its HA, because the HA is only aware of the MN's RCoA. HMIPv6 also supports route optimization by delivering packets between CN and MN, using the shortest possible path.

In the HMIPv6 network, the MAP can be utilized to perform access control of MN's, and to interact with the AAA protocol. For example, the AAA server in the MAP domain (*m*AAA) can accelerate a handoff process by using the MN's security credentials, which will enable it to verify whether a newly entered MN is allowed access to the network. The *m*AAA also interacts with an AAA server in the home network of the MN (*h*AAA) in performing the AAA process for newly entered MN. The most straightforward scenario would be as follows: An *m*AAA of the MAP domain can store the MN's security credentials after the MN is allowed network access. During the subnet domain handoff, the *m*AAA could pass the MN's security credentials to the *f*AAA located in the new AR's domain, to avoid performing the AAA process involving the *h*AAA and CN of the MN, whenever the MN moves to a different subnet. **Fig. 1** shows an example of the network architecture for AAA services in HMIPv6 network.

2.2 Multicast with mobile IP networks

In Mobile IP networks, multicasting is supported by two different approaches – namely, *Bi-directional Tunneled multicast* (MIP-BT) and *Remote Subscription* (MIP-RS) [1]. In a MIP-BT, the multicast packets, destined to a MN, are first routed to the HA of the MN; the HA then encapsulates and delivers the packets to the FA of the MN. However, each packet is very susceptible to the triangle routing problem. Moreover, it does not scale well, because each MN must always subscribe to the groups of interest through the HA. As we mentioned in the previous sub-section, these issues are handled in MIPv6 [2] and HMIPv6 [6], which support route optimization and a lowering of routing bandwidth overhead.

On the other hand, MIP-RS requires an MN to re-subscribe to the multicast group on the new FA whenever it performs handoff. Hence, it provides a simple implementation, indicating an efficient accommodation for the large number of MN's. It must be noted that it suffers from an "out-of-sync" problem, since the two FAs have different one-way transit delays from a multicast sender node. When an MN moves to new FA subnet and the new FA has a shorter delivery delay than that of the previous FA, the MN experiences packet loss, requiring the lost packets to be retransmitted from the sender node or other nodes, such as an FA. The same problem can still be an issue in MIPv6 and HMIPv6, when the MN performs handoff between two MAP's which have different delays from a sender node.

2.3 Techniques in cryptography

Cryptography is a compounding of some or all techniques and applications. Cryptosystems can be classified as symmetric and asymmetric systems. Symmetric systems use the same key for encryption and decryption. Symmetric key algorithms can include stream and block ciphers. The stream ciphers encrypt the bits of the message one at a time, whereas the block ciphers encrypt a number of bits as a single unit. The symmetric algorithms yield much faster operations than asymmetric algorithms, but the mechanism it provides to enable the users to securely share the secret key is always a difficult issue. On the other hand, asymmetric systems use two separate keys. A sender uses the public key of the receiver for encryption, but his or her own private key for decryption. However, it is typically one hundred times slower than a symmetric system. In order to achieve maximum efficiency, both systems are used cooperatively. For example, a secret key is encrypted by a receiver's public key.

3. FI-based Secure Multicast

3.1 Problem statement

The important and well-known techniques in cryptographic systems, such as Digital Signature, RSA, Diffie-Hellman, and the Elliptic Curve, exploit the characteristics of some functions to compute some phrase encoded by private keys. However, most of these problems are computationally expensive. Multicast applications requiring large amounts of real-time data transmission in a wireless mobile environment, must rely on the design of a secure, scalable and reliable multicast protocol. As we mentioned in the first section, all of the exiting protocols have their own limitations in terms of scalability and message overhead.

In this section, we describe the proposed solution for providing a secure scalable multicast service, including AAA resolution in HMIPv6 networks. We also explore the use of FI technology for cryptographic systems. As we mentioned, our target infrastructure is a HMIPv6 Network. However, it must also be noted that our solution can easily be extended to the more recent Proxy MIPv6 [19][20] protocol without major modifications.

3.2 Coordination of data packets

Firstly, we describe how the lost packets can be retransmitted from the MAP's to MN's. Note that this error recovery plays an important role in enabling the MAP and its latest MN's to generate the same local group key in an independent manner, based on the FI system. Our recovery scheme guarantees that each network entity in the same domain can have the same data packet for group key generation.

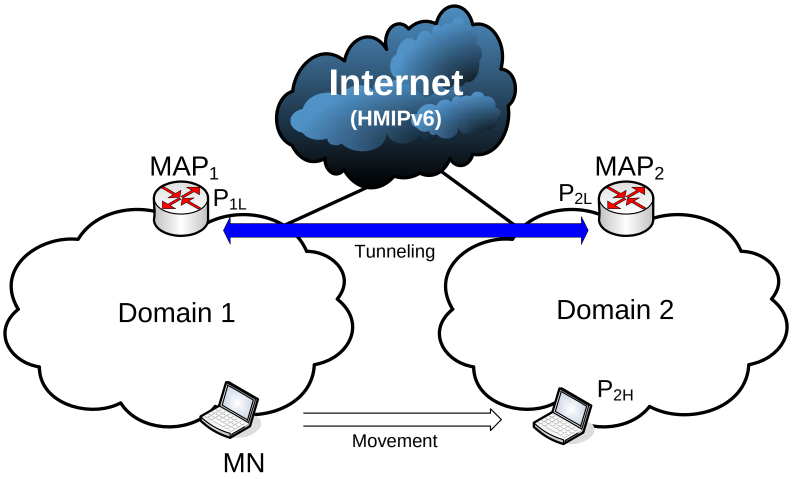

Fig. 2. HMIPv6 multicast

Fig. 2 shows the operations of the MAP's and MN in an HMIPv6 environment. When the MN visits a new MAP domain (MAP domain 2 in our example), it registers with the MAP_2 by sending a *Registration Request* message. The MAP_2 then updates its visitor list and relays the *Registration Reply* message to the MN. The MN now sends a join message to MAP_2 for its specific multicast group, with the highest packet sequence number, P_{2H}, that it has received from MAP_1. The MAP_2 then sends an IGMP-join message for the multicast group, to its first-hop multicast router. After joining the multicast group, the MAP_2 allocates some buffer space for this multicast group. This buffer is a temporary storage for multicast packets, starting from the lowest packet sequence number P_{2L} sent by a sender node. The MAP_2 is able to perform error recovery for its MN's by using the packets stored in its buffer.

Initially, the MAP_2 compares the packet sequence number of its least recently received packet, P_{2L}, with P_{2H} sent from its MN. If the MAP_2 has a higher packet sequence number, that is ($P_{2L} > P_{2H}+1$), it requests the offset packets, [$P_{2H}+1$, $P_{2L}-1$] for MAP_1, which has stored some recently received packets in its buffer, [$P_{2H}+1$, P_{1L}], for some period of time. It encapsulates the request packets and sends them to the MAP_2 using a unicast tunneling mechanism. The MAP_2 de-capsulates the received packets and saves them to its buffer. These packets might be retransmitted when its MN's request them with NAKs. If $P_{2L} = P_{2H}+1$, the MAP_2 immediately transmits the packet to the MN, starting from packet P_{2L}. If $P_{2L} < P_{2H}+1$, the MAP_2 transmits the packets to the MN, starting from packet $P_{2H}+1$. In both cases, there is no more communication between MAP_1 and $MAP_{2,}$ to compensate for the offset packets. Note that this approach will not introduce any duplicate packets caused by the out-of-sync problem.

3.3 Encryption/Decryption with FI sequences

After coordinating the most recently received data packets, the latest MN receives the encrypted data from its MAP. That is, the MAP generates secret information for packet P_{2H} +1 using a FI system, encrypts it using a local group key and transmits to its MN, until the MN leaves its domain.

Many popular techniques in cryptographic systems, such as RSA, Diffe-Hellman, and Elliptic Curve approaches, exploit the characteristics of some functions to compute some phrases encrypted by private keys. Most of these algorithms, however, are extremely expensive in terms of computational complexity. In this paper we present a new mechanism for selecting phrases in a sequence X, and producing a finite set of indices, S = [X, F, I], which can later be used to identify the sequence for the purposes of authentication. The sequences are chosen using Finite Inductive (FI) sequences, in which substrings in some packets are algorithmically generated. A discussion and application of FI sequences is found in [1]. An FI sequence S = *s*1, *s*2, …, *s*n over a set of symbols P = {*p*1, *p*2, …, *p*n} is a pattern in which some preceding subsequence *s*i, …, *s*j uniquely identifies *s*j+1, we will term *x*.

If every substring *y* = *s*i, …, *s*j uniquely identifies *x* then the pair {*y*, *x*} is called an *implicant*, the substring *y* is called the *antecedent*, and *x* the *consequent*. In these types of sequences, we can store the antecedents and expunge the consequent from the sequence, since the antecedent uniquely identifies the consequent. When consequents are eliminated from FI sequences, we produce a new sequence called a *residual* sequence which may also generate new implicants. The process of generating implicants and residuals from a sequence is called *factoring*. The set of all implicants and residuals for a sequence is called a *ruling,* the maximum length of antecedents in a ruling for a level is the *inductive base* for that level and the *inductive base* ≥ 1.

We can reconstruct the original sequence from a ruling using the implicants and the residual sequence. Recall that factoring a sequence S = *s*1, *s*2, …, *s*n produces rules of the form *x*→*y* where *y* is the consequent and *x* the antecedent. This means that if *x*→*y* is uniquely defined in the sequence, then *xy* is sub-sequence in the sequence S. We use this property to generate the original sequence from the rules and the residual sequence.

We assume the given sequence consists of hexadecimal numbers. Then, the basic operations are as follows and we show this process in the next example.

1. Use a start symbol F, which is out of range of the hexadecimal number, to designate the first symbol in the sequence. F→*x*, for some symbol *x*, which is the first symbol in the string being processed.
2. Next, if *x*→*y* is a rule for the string, then append '*y*' to '*x*' to generate *xy*. In general, if *x*i, …, *x*j→*a* is a valid rule we append '*a*' to *x*i, …, *x*j.
3. If there is no valid *x*→*y* in the implicants, we append the first symbol in the residual sequence to the subsequence that is already generated.

3.3.1 Encryption

The MAP performs factoring with the following sequence, into implicants using two levels with an inductive base of two. After that, it transmits the generated rules and residual to the MN.

$$\text{CE97E993A2} \tag{1}$$

Select a symbol F, which is not in the sequence to be factored, to designate the start of the sequence, and append F to the start of the string yielding:

$$\text{FCE97E993A2} \tag{2}$$

Factoring the string of (1) with inductive base two, we have the following rules for the first level:

$$\begin{gathered} \text{F} \rightarrow \text{C, C} \rightarrow \text{E, CE} \rightarrow \text{9, 97} \rightarrow \text{E,} \\ \text{7E} \rightarrow \text{9, 99} \rightarrow \text{3, 3} \rightarrow \text{A, A} \rightarrow \text{2} \end{gathered} \tag{3}$$

Eliminating the consequents from the sequence of (2), we have the residual sequence

$$R_1 = \text{F79} \tag{4}$$

After factoring the residual sequence (4) with the same inductive base, we next obtain the implicants and residual shown in (5) and (6), respectively.

$$\text{F} \rightarrow \text{7, 7} \rightarrow \text{9} \tag{5}$$

$$R_1 = \text{F} \tag{6}$$

From (3) and (5), we can now form **Table 1**, of Implicants with inductive base two, resulting from the factoring of the string of (2).

Table 1. Implicants at two levels, from the string of (2)

Level 1		**Level 2**	
Number	Rule	Number	Rule
1	F→C	1	F→7
2	C→E	2	7→9
3	CE→9		
4	97→E		
5	7E→9		
6	99→3		
7	3→A		
8	A→2		

3.3.2 Decryption

Based on the residual sequence of (6), F, and the implicants in Level 1 of **Table 1** (rules 1-8), MN constructs the sequence (2). It uses the following steps to generate the sequence.

1. Starting symbol: F
   - Because F→C, rule #1, generate the subsequence, ‘FC’.
   - Using rules #2, and #3, generate the subsequence ‘FCE9’
2. No rule with antecedent ‘9’ or ‘E9’ is valid in the implicants, thus use the residual sequence to obtain the leftmost residual symbol in level 2, which is ‘7’. Generate the subsequence ‘FCE97’. Delete ‘7’ from the residual sequence.
3. Use the rules 97→E (#4), 7E→9 (#5) to generate ‘FCE97E9’
4. Because no rule with antecedent ‘9’ or ‘E9’ is valid in the implicants, again use the residual sequence and borrow the leftmost symbol, ‘9’. Then, generate the subsequence ‘FCE97E99’ and delete ‘9’ from the residual sequence.

5. Using rules 99→3 (#6), 3→A (#7), and A→2 (#8), generate the subsequence 'FCE97E993A2'.

3.4 Operations for MN authentication

In this section, we demonstrate the overall processes for our proposed solution. We demonstrate how the MN can be authenticated to obtain a multicast service, including AAA resolution in a MMIPv6 network. We make the following assumptions. These are standard hypotheses used for all of the Mobile IP-based approaches [1][2][3][4][5][6][7][10][16][17][18][19][20].

- There are AAA servers in the subnets called AAAF's. Each AAAF performs three tasks including (a) enabling the MN to be authenticated, (b) generating accounting data for MN, and (c) authorizing the MN to use network resources.
- There is an AAA server in the home network called AAAH.
- An MN and its AAAH have a long-term secret key.

In addition, we assume the multicast sender node has the multicast group key and this key is safely distributed when the MN joins the multicast group. We also assume there is an AAA server in the MAP domain called AAAM, and the communication among the AAAA servers is secure, with a trust relationship. As seen in **Fig. 3**, HMIPv6 requires the following sequence of actions when the MN starts registration at an AR. Then, our proposed secure multicast service can be performed as depicted in **Fig. 4**:

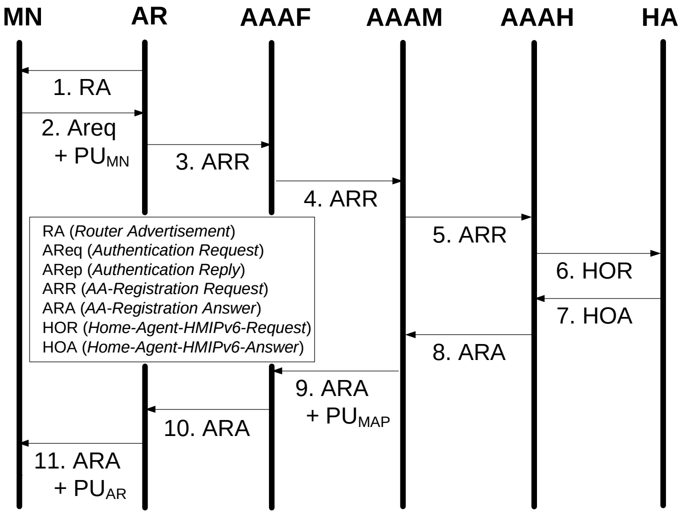
MN
AR
AAAF
AAAM
AAAH
HA
1. RA
2. Areq
+ PU_MN
3. ARR
4. ARR
5. ARR
6. HOR
7. HOA
8. ARA
9. ARA
+ PU_MAP
10. ARA
11. ARA
+ PU_AR
RA (Router Advertisement)
AReq (Authentication Request)
ARep (Authentication Reply)
ARR (AA-Registration Request)
ARA (AA-Registration Answer)
HOR (Home-Agent-HMIPv6-Request)
HOA (Home-Agent-HMIPv6-Answer)

Fig. 3. MN authentication with AAA resolution

1. The MN listens to *Router Advertisement* (RA) messages from the AR in the new administrative domain.
2. By referencing the RA message, the MN recognizes the availability of the MAP's in the current domain. The MN selects the best MAP, based on the MAP selection algorithm, and sends an *Authentication Request* (AReq) message to the AR. This message also contains MN's information, including its public key PU_{MN}.

3. When the AR receives an AReq message, it saves the public key PU_{MN}, creates an *AA-Registration-Request Command* (ARR) message, and sends it to the AAAF.
4. When the AAAF receives the ARR message, it forwards the message to the AAAM of the MAP which has been selected by the MN.
5. After receiving the ARR message from the AAAF, the AAAM saves the public key PU_{MN} and forwards the ARR message to the AAAH of the MN.
6. After receiving the ARR message from the AAAM, the AAAH should now contact HA to obtain a certification of the MN. In order to do this, it generates a *Home-Agent-HMIPv6-Request Command* (HOR) message and sends it to the MN's HA.
7. The HA processes this message and creates a key to establish an AAA *security association* (SA) with the MN. Then, it responds with a *Home-Agent_HMIPv6-Answer Command* (HOA) message.
8. After receiving a positive answer, the AAAH generates and sends an *AA-Registration-Answer Command* (ARA) message that has an authentication result to the AAAM.
9. After receiving the ARA message from the AAAH, the AAAM includes its public key, PU_{MAP} in the ARA message and forwards it to the AAAF.
10. The AAAF stores the authentication result locally and forwards the ARA message to the AR.

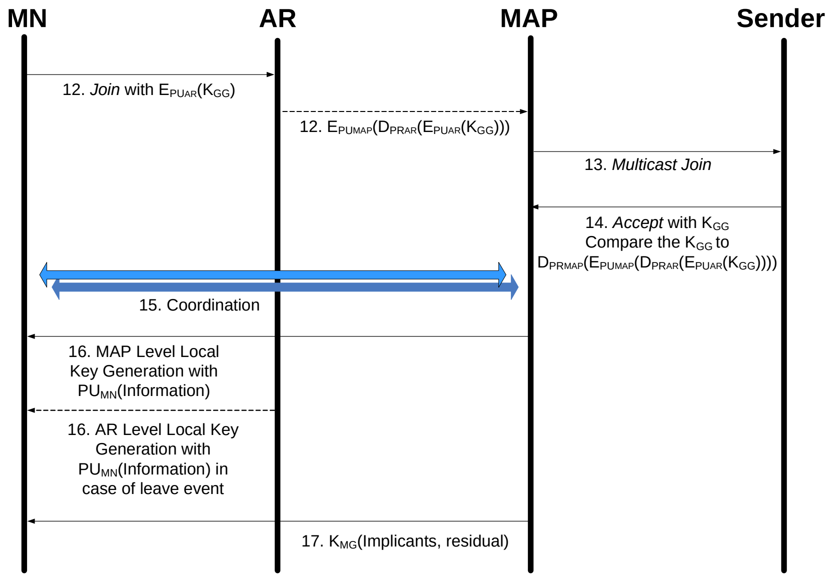
MN
AR
MAP
Sender
12. *Join* with $E_{PUAR}(K_{GG})$
12. $E_{PUMAP}(D_{PRAR}(E_{PUAR}(K_{GG})))$
13. *Multicast Join*
14. *Accept* with K_{GG}
Compare the K_{GG} to
$D_{PRMAP}(E_{PUMAP}(D_{PRAR}(E_{PUAR}(K_{GG}))))$
15. Coordination
16. MAP Level Local Key Generation with PU_{MN}(Information)
16. AR Level Local Key Generation with PU_{MN}(Information) in case of leave event
17. K_{MG}(Implicants, residual)

Fig. 4. Local group key generation and distribution

11. The AR modifies the ARA message by including its public key, PU_{AR} in the *Authentication Reply* (ARep) message and forwards it to the MN. Based on this message, the MN now obtains the authentication result from the AAAH, the established key for the SA, and a public key, PU_{MAP} and PU_{AR} from its AAAM and AR.
12. The MN sends a *Join* message, to the MAP though the AR. This message includes 1) its specific multicast group address; 2) its highest packet sequence number that it has received from its previous MAP; and 3) its global multicast group key, K_{GG}, shared with a multicast sender node. This key is encrypted with PU_{AR}, decrypted with PR_{AR} in the AR, and encrypted again with PU_{MAP}.

13. Then, the MAP sends an IGMP-join message for the multicast group. It also decrypts the $E_{PU_{MAP}}(K_{GG})$ using its private key PR_{MAP}.
14. It receives the global group key, K_{GG}, from the sender node and compares it to the K_{GG}, which is submitted by the MN.
15. Once it is approved, the MAP and MN coordinate their most recently received packet.
16. The MAP sends a *MAP Group Key Generation* message to the MN. This message is encrypted with PU_{MN} and includes information for which the MAP and the MN can independently generate the same MAP group key K_{MG}. If one of the members leaves the multicast session in a subnet, this process will be done at the AR level. That is, the AR level group key K_{AG} is regenerated and distributed among the subnet group members rather than the entire MAP level group members.
17. The MAP performs an FI system for an individual or a group of packets, encrypts the Implicants and residual with a K_{MG} , and sends them to the MN.

4. Expected Performance

In this section, we analyze the performance of the proposed solution, in terms of the number of group key generations and the message overhead required to distribute the new group key. We compare the results to the traditional approaches, for which the sender node is required to change the current K_{GG} whenever a join/leave event occurs. We also demonstrate the performance when our approach is applied to the MIPv6 network and also the HMIPv6 network.

For our solution, the global multicast group key, K_{GG}, is not changed over the multicast session. This means a join/leave event does not affect the entire multicast group, in terms of traffic overhead and a new key generation/distribution. Instead, this overhead is bounded within the MAP domain. Moreover, this overhead is bounded within the subnet in the case of a leave event. The MAP level group key, K_{MG}, is also not changed, providing the MN performs the intra-domain handoff. That is, although the MN moves to the new subnet, the AR will not generate a new AR level group key K_{AG} for the new MN. Instead, it will communicate with the MN, providing the MN suggests a valid MAP group key K_{MG}. In the case of a leave event, the AR level group key K_{AG} will be regenerated and distributed only to the MN's in the same subnet. Note that, in an MIPv6 network without the MAP concept, the local group key, K_{LG}, should be generated by the AR level and changed whenever the MN enters the new subnet. We assume the sender node can handle up to m MN's in a multicast session, the average arrival rate of MN's is denoted by λ_i (=λ, if $0 \le i \le m$), and the average departure rate of MN's is denoted by μ_i (=μ, if $0 \le i \le m$). Under this condition, the probability P_k that there are exactly k MN's in a current session is given by

$$P_k = (\rho)^k \cdot P_0, \text{ for } 1 \le k \le m \tag{7}$$

where $\rho = \lambda/\mu$. Because $\sum_{k=0}^{m} P_k = 1$, we have

$$P_0 = \begin{cases} \dfrac{1}{m+1} & (\lambda = \mu) \\ \dfrac{1}{\sum_{k=0}^{m} (\rho)^k} & (\lambda \neq \mu) \end{cases}$$

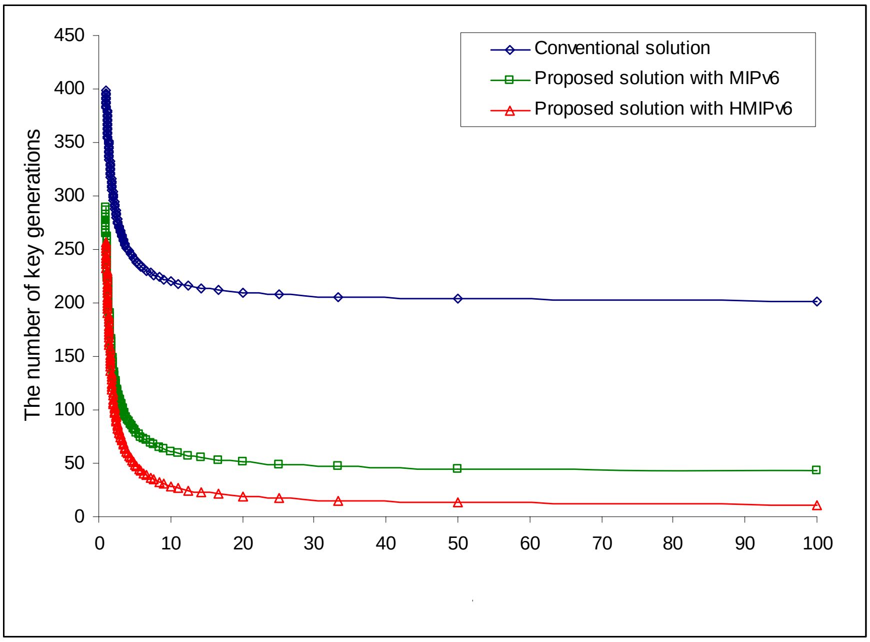

Fig. 5. The number of key generations vs. ρ

While the average number of MN's, L, in a multicast session obeys equation (8), the L can be calculated based on the different mobility conditions. Note that the effective arrival rate λ_e which is defined as $\lambda_i(1–P_m)$ is equal to λ, provided $m > \lambda$ and P_m goes to 0.

$$L = E(N) = \sum_{k=0}^{m} k \cdot P_k \tag{8}$$

$$L = \begin{cases} \dfrac{m}{2} & (\mu = \lambda, m > \lambda) \\ \dfrac{\rho}{1-\rho} - \dfrac{(m+1)\cdot(\rho)^{m+1}}{1-(\rho)^{m+1}} & (\mu \neq \lambda, m > \lambda) \end{cases}$$

On the other hand, if the average arrival rate is greater than the capacity of the sender node, then L can be represented by equation (9).

$$L = \sum_{k=0}^{\mu} (m-k) \cdot {}_m C_k \cdot (\rho)^k \cdot (1-\rho)^{m-l} \text{ , } (m < \lambda) \tag{9}$$

Next, consider the comparison of solutions, in terms of key generation and distribution overhead. As we discussed previously, conventional solutions require the sender node to change the current K_{GG} whenever a new MN joins a multicast group, to prevent the new MN from decrypting the previous data packets. The same process should be performed whenever a leave event occurs. In addition, whenever the group key is generated, the key is distributed to all MN's that are currently in the multicast group. As a result, it requires $(\lambda+\mu)\cdot T$ new group key generations and $(\lambda+\mu)\cdot T\cdot L$ new group key distributions, where T is the duration of the multicast session.

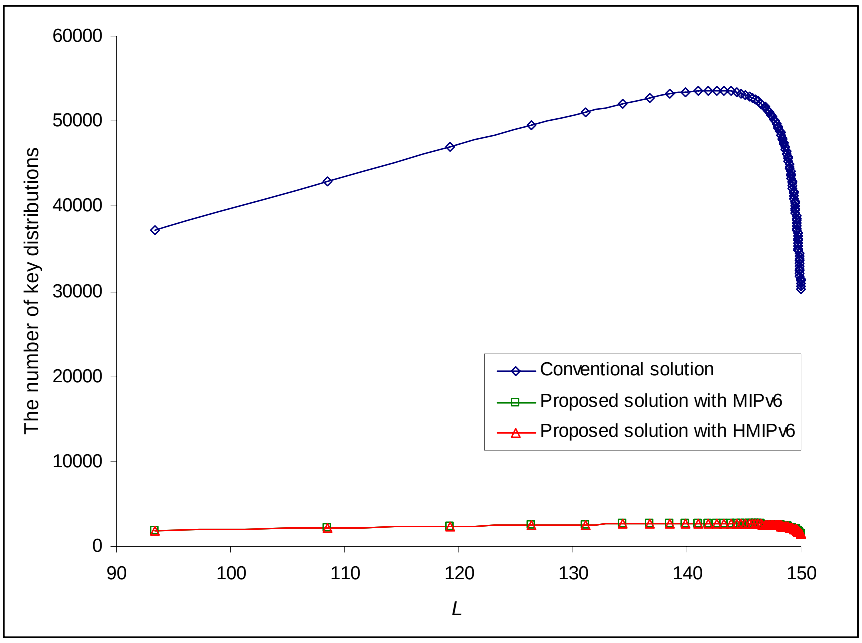

Fig. 6. The number of key distributions vs. *L*

On the other hand, the proposed solution does not require the sender node to generate a group key whenever a join/leave event occurs. In an HMIPv6 network, a multicast group is now separated into several domains and our solution requires the MAP in the domain to subscribe to the same multicast service. A sender node and the MAP's of the domains share the global group key K_{GG}. Although an MN enters a new subnet or new domain, this key will not be changed. Hence, if we assume the multicast session involves *M* different domains, the K_{GG} is fixed until the session ends; but this key should be delivered to the MAP when the new MN enters the domain of the MAP. As a result, there are *M* different messages to be delivered. On the other hand, a MAP and its MN's share local group key K_{MG}, which is generated by the FI system. Hence, MN's can decrypt data packets sent by their MAP's, by using K_{MG}. Instead of the K_{GG}, the current K_{MG}, which is relatively computationally simple to generate, is generated and distributed when a new MN joins a multicast group. As a result, each MAP needs to generate a new K_{MG} whenever a new MN enters its domain, and when the MAP is not a member of the multicast group. The same process should be performed whenever a MN leaves the domain. As the multicast session involves *M* different domains, the number of K_{MG} generations in an entire multicast session can be given by $(1+\mu/M)\cdot T\cdot M$. The local group key K_{MG} should also be delivered to the new MN. The AR level local group key K_{MG} should be delivered to the current MN's in a subnet, resulting in $(\lambda/M+\mu L/M\cdot D)\cdot T\cdot M$ delivery messages.

If we apply our solution to the MIPv6 network, the K_{GG} can still be fixed until the session ends; but this key should be delivered to the AR when the new MN enters the subnet of the AR. In addition, a FA and its MN's share local group key K_{LG}. As a result, each AR needs to generate a new K_{LG} whenever a new MN enters into its subnet or leaves the subnet. Therefore, when we assume there are *D* different subnets in a multicast session, the number of K_{LG} generations in an entire multicast session can be represented by $(1+\mu/D)\cdot T\cdot D$. This local group key should also be delivered to the current MN's in a subnet, resulting in $(\lambda/D+\mu L/D^2)\cdot T\cdot D$ delivery messages.

Fig. 5 and **6** show the results of the number of key generations and distribution messages, respectively, when we use fixed λ (= 100), T (= 2 hours), M (= 4 domains), D (= 20 subnets), m values (= 150) and dynamic μ value (=1 to 99). As can be seen, our solution for the HMIPv6 network requires much less overhead, in terms of local key generation and distribution, compared to other approaches. The number of global group keys, in our solution for the MIPv6 and HMIPv6 network, is independent of the parameter values. In particular, when we apply our solution to the HMIPv6 environment, it can dramatically reduce the number of key generations, because an MN leave event does not affect the entire multicast session. Note that the conventional solutions could introduce significant time overhead to generate a new group key, due to the complexity of group key generation algorithms. It must also be noted that our local key generations/distributions are evenly distributed over the subnets, and computationally easy to generate, because FI can run in O($n$ln(n)) time.

5. Conclusion and Plans for Future Work

We have proposed a group key management solution providing secure scalable and reliable multicast service in a mobile wireless environment. The communication between the MAP and its MN's, is performed with a local one time pad key generated by FI sequences. We conclude that this can reduce the key generation/distribution overhead, when it is deployed in a mobile wireless multicast service. This is because a) the key management tasks can be delegated among the multiple agent routers, and b) the group keys can be efficiently generated at the agent routers, by using FI systems. Also, the security solutions we are designing must be able to thwart many kinds of practical threats, to enable multicasting for many kinds of practical applications. This consideration leads to several interesting research issues.

Firstly, we conclude that, because paths between handover points are deterministic, and individual users tend to have fixed patterns of behavior, such information can be used to predict the most likely next attachment points, thus reducing the communication overhead. We are developing a scheme that will store historical information about the individual MN. That information can be used to predict the next attachment point in the network. In addition, more global information will be stored, to determine if the local data concerning a particular node fits the more general patterns of usage for mobile nodes at this attachment point. The data is stored using a specialized storage format called a finite inductive sequence (FI), which is described in this paper. Such sequences can be used to describe processes with a very short history, for example two, three or four previous steps, to yield a prediction that can be extrapolated as far into the future as desired. This prediction can be non-deterministic and based on a probabilistic determination, where the next connection points can include a few that are almost equally likely.

Secondly, we will also extend our focus on the environment where a group of mobile nodes need to be connected to the Internet through the mobile routers, which connect the subnet to the global Internet. That is, we turn our attention from host mobility to the network mobility, which manages the mobility of a group of MN's rather than mobility management of a single MN. For example, MN's in the same vehicle change their network attachment points as the vehicle moves along its path. If we apply our solution in a straightforward manner to this environment, the agent router should perform the movement prediction for an individual MN, although all of them will show the same movement pattern. This approach will result in an excessive overhead, due to the unnecessary operations. This overhead can be eliminated when a mobile router in a vehicle acts as a representative of the MN's. This is because a) mobility prediction using the FI system can be performed based on only the movement of the mobile router, and b) only the mobile router performs AAA resolution and updates its home agent. As a result, the movement is transparent to the individual MN's.

Finally, we will develop a testing protocol, together with a simulation, for determining the efficacy of the proposed protocol. It is a modification to the present protocols, to significantly improve performance by reducing overhead. We will also actually implement such a protocol, which can be loaded into wireless network devices based on the availability of the equipment. In addition, we will consider temporary *ad hoc* networks, such as those used by the military and those used in disaster situations. We will be able to move the base stations and eliminate the fixed MN movement patterns between BS's, to determine how this protocol will actually support such *ad hoc* situations.

References

[1] E. Gustafsson, A. Jonsson, and C. Perkins, "Mobile IP Regional Registration," *IETF RFC 2002*, March 2001.

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

[3] F. Le, B. Patil, C. Perkins, S. Faccin, "Diameter Based Protocol," *IETF RFC 3588*, November 2004.

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

[5] C. Perkins, "Route Optimization in Mobile IP," *IETF Draft*, draft-ietf-mobileip-08.txt, September 2001.

[6] H. Soliman, C. Castelluccia, K. El Malki, and L. Bellier, "Hierarchical Mobile IPv6 Mobility Management (HMIPv6)," *IETF RFC* 4140, August 2005.

[7] E. Lee, J. Baek, and S. Huang, "A Dynamic Mobility Management Scheme for VoIP Services," *In Proc. of the IEEE ITNG*, pp. 340–345, April 2006.

[8] T. Hardjono, B. Cain, and N. Doraswamy, "A Framework for Group Key Management for Multicast Security," *IETF draft*, draft-ietf-ipsec-gkmframework-03.txt August 2000.

[9] T. Hardjono, B. Cain, and I. Monga, "Intra-Domain Group Key Management Protocol," *IETF draft*, draft-ietf-ipsec-intragkm-00.txt, September 2000.

[10] S. Mittra, "Iolus: A Framework for Scalable Secure Multicasting," *In Proc. of ACM SIGCOMM '97*, pp. 277–278, September 1997.

[11] Q. Ke, Z. Mingtian, L. Naiqi, H. Yujie, and G. Jiandong, "A Novel Group Key Management Based on Jacobian Elliptic Chebyshev Rational Map," *Lecture Notes in Computer Science*, 4672, pp. 287–295, September 2007.

[12] J. Hur, Y. Shin, and H Yoon, "Decentralized Group Key Management for Dynamic Networks Using Proxy Cryptography," *In Proc. of 3^{rd} ACM Workshop on QoS and Security for Wireless and Mobile Networks*, pp. 123–129, October 2007.

[13] I. Chatzigiannakis, E. Konstantinou, V. Liagkou, and P. Spirakis, "Agent-based Distributed Key Establishment in Wireless Sensor Networks," *In Proc. of WoWMoM 2007*, pp. 1–6, June 2007.

[14] J. H. Case, and P. S. Fisher, "Long Term Memory Modules," *Bulletin of Mathematical Biology*, 46(2): 295–326, Springer New York, March 1984.

[15] D. Minton, P. S. Fisher, “Method and Apparatus for a Dynamic Data Correction Appliance,” US Patent Office Award, February 2007, Patent Number 7,180,947.

[16] R. Chellappa, A. Jennings, and N. Shenoy, “A Comparative Study of Mobility Prediction in Fixed Wireless Networks and Mobile Ad hoc Networks,” *In Proc. of the IEEE ICC*, pp. 891–895, May 2003.

[17] J. Ficher, and J. Baek, “An Advanced Model for Adaptive MAP Selection in HMIPv6 Networks,” *in progress*.

[18] S. Pack, M. Nam, T. Kwon, and Y. Choi, “An Adaptive Mobility Anchor Point Selection Scheme in Hierarchical Mobile IPv6 Networks,” *Elsevier Computer Communications* (COMCOM), 29(16): 2066–3078, June 2005.

[19] V. Devarapalli, S. Gundavelli, K. Chowdhurym, and A. Muhanna, “Proxy Mobile IPv6 and Mobile IPv6 Inter-working,” *IETF Draft*, draft-devarapalli-netlmm-pmipv6-mipv6-00.txt, October 2007.

[20] S. Gundavelli, V. Devarapalli, K. Chowdhury, and B. Patil, “Proxy Mobile IPv6,” *IETF Draft*, draft-ietf-netlmm-proxymip6-07.txt, November 2007.

Jinsuk Baek is Assistant Professor of Computer Science at the Winston-Salem State University (WSSU), Winston-Salem, NC. He is the director of Network Protocols Group at the WSSU. He received his B.S. and M.S. degrees in Computer Science and Engineering from Hankuk University of Foreign Studies (HUFS), Korea, in 1996 and 1998, respectively and his Ph.D. in Computer Science from the University of Houston (UH) in 2004. Dr. Baek was a post doctorate research associate of the Distributed Multimedia Research Group at the UH. He acted as a consulting expert on behalf of Apple Computer, Inc in connection with Rong and Gabello Law Firm which serves as legal counsel to Apple computer. His research interests include scalable reliable multicast protocols, mobile computing, network security protocols, proxy caching systems, and formal verification of communication protocols. He is a member of the IEEE.

Paul S. Fisher is R. J. Reynolds Distinguished Professor of Computer Science at the Winston-Salem State University (WSSU), Winston-Salem, NC. He is the director of High Performance Computing Group at the WSSU. He received his B.A. and M.A. degrees in Mathematics from University of Utah and his Ph.D. in Computer Science from Arizona State University. He has written and managed more than 100 proposal efforts for corporations and DoD involving teams of 1 to 15 people. He worked as consultant to the U.S Army, U.S Navy, U.S Air Force and several companies over the years. In the 1990’s he commercialized an SBIR funded effort and built Lightning Strike, a wavelet compression codec, then sold the company to return to academe. His current research interests include wired/wireless communication protocols, image processing and pattern recognition.

Mingyung Kwak was born in Seoul, South Korea. He is an exchange student at the Winston-Salem State University (WSSU) through the 7+1 program conducted by Hankuk University of Foreign Studies (HUFS). He worked for Wake Forest University Baptist Medical Center as an Intern. He is a member of the Network Protocol Research Group at the WSSU and is expected to receive his B.S. degree in Computer Science in 2008. His areas of concentration are computer networks, network security, and programming languages.