Improving Cross-domain Authentication over Wireless Local Area Networks

Hahnsang Kim
INRIA, France
hahnsang.kim@inria.fr

Kang G. Shin
University of Michigan, USA
kgshin@eecs.umich.edu

Walid Dabbous
INRIA, France
walid.dabbous@inria.fr

Abstract

As mobile users cross the border of two adjacent domains with on-going sessions, their re-authentication causes a significant impact on inter-domain handoff latency as it requires remote contact with the authentication server across domains, making it difficult to employ current authentication protocols. This paper focuses on the cross-domain authentication over wireless local area networks (WLANs) that minimizes the need for remote access. We analyze the security requirements suggested by the IEEE 802.11i authentication standard, and consider additional requirements to help reduce the authentication latency without compromising the level of security. We propose an enhanced protocol called the Mobility-adjusted Authentication Protocol (MAP) that performs mutual authentication and hierarchical key derivation with minimal handshakes, relying on symmetric cryptographic functions. We also present security context nodes (SCNs) that handle security contexts in conjunction with MAP, which allows for avoiding continuous remote contact with the home authentication server. In contrast to Kerberos which favors inter-realm authentication, MAP achieves a 26% reduction of authentication latency without degrading the level of security.

1. Introduction

Time-sensitive applications, such as Voice over IP (VoIP) or video streams, are now possible over wireless local area networks (WLANs) such as those based on the IEEE 802.11 Standard [3] thanks to their high bandwidth. WLAN technologies also allow for roaming within public/corporate buildings or university campuses. Furthermore, we anticipate that mobile users might cross the border of domains without disrupting their on-going application sessions. However, VoIP requires the completion of a handoff in less than 50 ms for acceptable Quality-of-Service (QoS) [33], including the execution of the IEEE 802.11i authentication [5] as part of a secure handoff mechanism.

The IEEE 802.11i authentication is responsible for mutual authentication and key derivation for securing WLANs via 802.1X and a four-way handshake [5]. Recent efforts on security associations have been limited to distribution of keys to access points within a domain [23]. For inter-domain handoffs, however, the authentication latency is critical to the application QoS. Success in crossing a domain boundary is contingent upon whether the involved domain administrators agree on inter-domain handoffs or not. Allowing a "visitor" to use storage resources in a domain affects the performance of users in that domain. It is required to assume that a domain has an agreement on inter-domain handoffs with each of its neighbors, and to explore several storage resource configurations that impact the authentication latency.

Minimizing the number of messages to be exchanged is important as cross-domain authentication needs to contact the remote home server, and the authentication latency increases in proportion to the round-trip time between two points involved in inter-domain message exchanges. Optimization of the authentication protocol is of utmost importance since the existing redundant combination of authentication and key negotiation functions incurs more rounds of message exchanges than necessary.

We propose in this paper an enhanced protocol for cross-domain authentication, mobility-adjusted authentication protocol (MAP) which relies on far less costly symmetric cryptography. MAP reduces the cross-domain authentication latency by (1) reducing the number of message exchanges, (2) integrating the four-way handshake of 802.11i authentication into MAP, and (3) most likely avoiding remote contacts, generating security contexts. First, MAP requires less message exchanges without compromising security or the re-authentication mechanism that make the authentication latency reduced significantly. Second, in coordination with the authenticator in an access point, MAP defines a hierarchical key derivation and generates consecutive keys along with authentication operations. This allows one to optimize the 802.11i authentication mechanism by removing the need for a four-way handshake. Last of all, we present security context nodes (SCNs), the main role of which is to perform authentication operations on behalf of

Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'0
0-7695-2369-2/05 $20.00 © 2005 IEEE

the remote home authentication server. An SCN allows for exchanging security contexts generated after the completion of authentication, so that it may avoid the need for contacting the home server required each time the mobile nodes roam in a foreign domain.

Our evaluation results show that MAP accounts for 74% cross-domain authentication latency of Kerberos [16] and 85% of the latency of the Needham-Schroeder symmetric-key protocol (NS) [26,27]. It makes an up to 53% improvement in the authentication latency which is proportional to the end-to-end domain distance until the round-trip time counts up to 100 *ms*.

The paper is organized as follows. Section 2 gives an overview of the 802.11i authentication mechanism and design requirements. Section 3 describes MAP. Section 4 considers possible threats and analyzes the security of MAP. Section 5 examines the performance via measurements and simulation. Finally, we discuss related work in Section 6 and conclude the paper in Section 7.

2. Overview of Authentication Mechanism and Requirements

In this section, we first introduce the 802.11i authentication scheme and protocols applicable to the cross-domain authentication, describe the design requirements of authentication protocols, and give an overview of BAN logic.

2.1. The IEEE 802.11i Authentication

The IEEE 802.11i authentication (after (re)association) takes two main steps. First, 802.1X authentication, where an authentication protocol like TLS [6] operates, establishes the authenticity of end-to-end principals: the mobile station (STA) user and the authentication server (AS), so-called *authentication authorization and accounting* (AAA) [25]. Successful mutual verification of each identity leads to the derivation of a pair-wise master key (PMK). This key is transferred to the AP via a secure tunnel and eventually used as a seed for a pair-wise temporary key (PTK). Second, the STA and authenticator (AUTH) (which operates in an AP) perform a four-way handshake, exchanging their nonces, so that the temporary key is obtained and then the wireless link is secured. The performance of 802.11i authentication depends on the efficiency of this authentication protocol.

2.2. Cross-domain-related Protocols

There are two protocols—Kerberos and NS—that can be effectively extended to support the cross-domain authentication.

The Kerberos protocol provides cross-realm operations. By establishing inter-realm keys, the administrators of two realms allow a mobile user to receive services in a remote realm. The mobile user receives a remote ticket granting ticket (TGT) from the ticket granting server (TGS) in the local realm. It then obtains a service granting ticket (SGT) from the remote TGS in the other realm by using the issued remote TGT. With the SGT containing a secret key, the mobile user and the authentication server can authenticate each other. The remote TGT issued once can be reused to get TGTs in the current realm within a given period of time, but each time the mobile user moves into a foreign realm, it is required to get a remote TGT again by contacting its home TGS.

The NS protocol, on which Kerberos is based, is not intended to operate cross-organization boundaries. However, it can support cross-domain authentication with minor modifications.

The principal operations of the original protocol are the following. The initiator A and its correspondent B share secret keys KA and KB with the authentication server AS, respectively. In the beginning, A obtains two copies of a pair-wise key encrypted with KB and KA by AS, respectively, while communicating with each other. Afterwards, A sends B the KB-encrypted pair-wise key. In addition, A and B exchange their nonce to authenticate each other.

If A corresponds to STA, and B and AS are a foreign and home ASs, respectively, since the foreign AS requires a set of pair-wise keys, the home AS generates and sends a set of multiple different keys. Once receiving them, the foreign AS contacts the home AS no more and the number of message exchanges is reduced up to three.

We will use two protocols to comparatively evaluate the throughout of our protocol via simulation.

2.3. Design Requirements

The 802.11i authentication suggests several requirements that must not be compromised to secure WLANs.

Requirement 1: The STA and AS must be able to authenticate each other. Since the STA establishes a wireless connection to the AS through anonymous APs, it should be able to identify the AS, so should the AS.

Requirement 2: Successful mutual authentication leads to the derivation of a fresh key for the AS and the STA. After successful mutual authentication, a 256-bit key (i.e., PMK) is generated by the AS and the STA, and is eventually used by the STA and the AP. This key should never have been used before, and should become obsolete and be replaced whenever the STA binds with a new AP.

Requirement 3: Mutual authentication should be strong enough to be shielded from any unauthorized reception. It is not easy to demonstrate the safety of the authentication protocol, but there are theoretical approaches for this purpose. For example, formal verification methods based on

model checking, theorem proving, modal logic, and modular approach are widely used. We will show a logical proof of MAP using BAN logic in Section 4.2.

In addition to these requirements, we present the following recommendations for the authentication protocol design to help achieve fast handoffs in WLANs.

Recommendation 1: Minimizing the number of message exchanges helps improve the performance of handoffs with cross-domain authentication. We should evaluate the effect of the number of message exchanges.

Recommendation 2: Reducing the use of lightweight cryptographic algorithms helps low-power mobile terminals like personal data assistants mitigate the performance overhead of computation-intensive cryptographic algorithms.

2.4. BAN Logic

BAN logic [10] is a well-known of modal logics developed for authentication protocol analysis. It presented the proof that a simple logic could be applied to describe the beliefs of trustworthy communicating parties. It found authentication protocols in the literature containing redundancies or security flaws. BAN logic reasons that the protocol operates as correctly as it is expected. It is sufficiently effective to convince one of the correctness of the authentication mechanisms with a logical reasonings.

2.5. Prerequisite

We introduce briefly logical postulates in BAN logic that will be used for the proof of our protocol. Full details of its rules are given in [10].

- The *message-meaning* rules are applied to the interpretation of messages for shared keys

$$\frac{P \text{ believes } Q \overset{K}{\longleftrightarrow} P,\ P \text{ sees } \{X\}_K}{P \text{ believes } Q \text{ said } X}$$

  and for shared secrets,

$$\frac{P \text{ believes } Q \overset{Y}{\Longleftrightarrow} P,\ P \text{ sees } < X >_Y}{P \text{ believes } Q \text{ said } X}.$$

- The *nonce-verification* rule represents the check that a message is recent and that the sender still believes in it:

$$\frac{P \text{ believes } fresh(X),\ P \text{ believes } Q \text{ said } X}{P \text{ believes } Q \text{ believes } X}.$$

- The *jurisdiction* rule states that if P believes that Q has jurisdiction over X then P trusts Q on the truth of X:

$$\frac{P \text{ believes } Q \text{ controls } X,\ P \text{ believes } Q \text{ believes } X}{P \text{ believes } X}.$$

In addition, MAC (Hash Message Authentication Code) function, involving the use of a secret key to generate a cryptographic checksum, can also provide authentication of its source. In this aspect, we interpret $MAC(m,K)$ as a unit of the secret $< X >_Y$. It suffices to intend that Y be a secret and its presence prove the identity of whoever utters it.

3. MAP

In this section, we describe an authentication architecture that is basically analogous to the AAA one and our protocol that consists mainly of three functional modules. Each module interacts with each other for authentication and key derivation.

3.1. Architecture

Authentication operations work with three entities: an STA, the back-end AS, and the authenticator (AUTH). An STA represents the end user with a wireless LAN equipped device. The AS verifies the authenticity of each STA and provides keys to secure the wireless link. The AUTH relays traffic between the STA and the AS, which is eventually an end-point of a secure wireless link.

In addition to dealing with these entities, our protocol solves the cross-domain authentication problem by introducing a new node, called *security context nodes* (SCNs). SCNs are placed between the AUTHs and the AS. The SCN is logically distinct from the AS, although both may reside on the same physical machine or be integrated into the AS. The SCN functions as follows. After receiving a security context[1] issued by the AS, it can perform re-authentication on behalf of the AS. The SCNs are distributed in each domain so that they can mitigate the authentication latency while the STA roams around the domain. It is assumed that in case of the communication of inter-administrative domains they have a security association agreement on roaming and are securely connected to one another by sharing inter-domain keys. This combination is adaptable to the security architecture of 802.11i authentication and Wi-Fi Protected Access 2 (WPA2) [1]. The protocol describing how messages are exchanged between the SCNs is part of our future work. In this paper, we will give a rough idea of how to exchange messages between the SCNs in Section 3.2.

Figure 1 shows our MAP authentication architecture. The MAP server module on the AS, described in Section 3.5.1, is an end-point authentication protocol securely connected to the AUTHs via the SCN. The AS used in the architecture is functionally equivalent to the AAA server. The MAP security context module (SC module) in the SCN,

[1] Its contents vary with individual protocols. MAP is expected to have a set of authentication value pairs, identity (= mobile node Id), validity time, time stamp, mean time of handoff, counter and other security information.

described in Section 3.5.2, helps the AS communicate with the other MAP-support AS for cross-domain authentication. The AUTH is an authentication client as a pass-through authenticator. It relays authentication traffic from the STA to the AS and vice versa. The MAP client module in the STA, described in Section 3.5.4, is an end-point authentication party that requests authentication and eventually establishes a secure link with the attached AP.

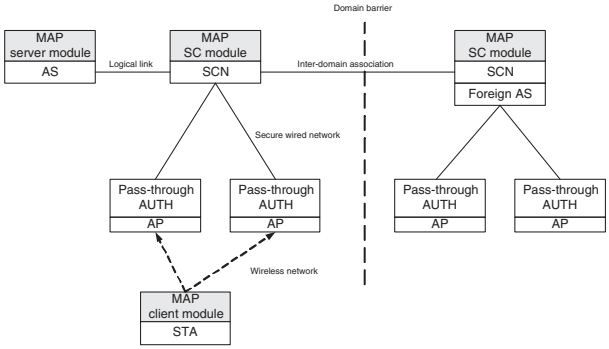

Figure 1. Authentication architecture

3.2. Communication between SCNs

The SCNs communicate with each other, based on a peer-to-peer manner. There are two ways of transferring security contexts among involved SCNs. In case of no security context cached in an SCN with which an STA has just associated, the targeted SCN fetches security context from the original SCN with which the STA associated previously; *reactive* transfer introduces latency in security context fetching. On the other hand, the original SCN may somehow forward the targeted SCN(s) the security context ahead; *proactive* transfer consists on estimations of the STA's direction and management of security contexts, the combination of which leads to a tradeoff between storage overhead and latency performance. We will elaborate on this issue in our future work.

3.3. Authentication

The MAP's authentication relies on MAC algorithms [17]. The MAC values rely on shared symmetric keys, the management of which is not easy to scale in that two communication parties must somehow exchange the key in a secure way, compared to that of asymmetric-key pairs. However, on the other hand, signing and verifying public keys are very time-consuming; the MAC values are preferred over digital signatures because the MAC computation is two to three orders of magnitude faster. There is a tradeoff between scalability and CPU usage; we chose cost efficiency since it matches our design goal.

3.4. Defined Keys

We define three types of keys for the different purpose: primary key (PK), domain key (DK) and temporary key (TK). Table 1 shows a summary of the association of keys and their sharing. PK is a long-term symmetric key which may

Table 1. Defined keys

key	derivation	belonging
PK	pre-installed	AS and STA
DK	PK	SCN and STA
TK	$DK^{\dagger}$	AP and STA

be periodically updated and deployed somehow, *e.g.*, online subscription to a service provider or off-line set-up with a purchased card. It is assumed that PK has guaranteed protection against disclosure for a sufficiently long period of time. DK is a quasi-primary key in (subset of) a domain, which is derived from PK and the previous DK. The STA generates a new DK as it changes a domain/subnet; an old DK must be dethroned. In addition, $DK^{\dagger}$, a n-time-hashed DK, is defined. The purpose of introducing the key is to provide *loose coupling* of DK and TK that is derived from $DK^{\dagger}$. TK is a link key affiliated with securing a wireless link established between the STA and the AP. TK binds with the addresses of two involved physical devices.

3.5. Message Exchanges

MAP defines four and two messages for the initial and re-authentication, respectively. The *auth-req* message is sent by the client module in the STA, which triggers a negotiation on authentication and key agreement from scratch. The *auth-chal* message sent by the server, as a return message, is used for the purpose of challenging the STA, with an encrypted code used for verifying the authenticity of the AS to the STA. The *chal-res* message, sent by the STA, is a response. This message contains a nonce-response encrypted code so that the AS verifies the authenticity of the STA. The AS, in return, replies with the *auth-res* message.

Once the first authentication procedure is performed successfully, the *reauth-req* and *reauth-res* messages are exchanged for re-authentication. The *reauth-req* message sent by the STA is captured and processed by the SCN. After verifying the authentication code in the message, the SCN responds with the *reauth-res* message containing the result of authentication. Described below are the behaviors of modules involved in MAP.

3.5.1 MAP Server Module

This module is involved primarily in the initial authentication. The following procedure describes how an STA's au-

thentication request is handled.

var: $ns_{1..n}$, $nc_{1..n} := 0$;
for all i: *auth-req* of Id_i in buffer **do**
  ns_i=Refresh(ns_i);
  send *auth-chal*: $ns_i \mid \mathrm{MAC}_{PK_i}(\mathrm{Id}_i, nc', ns_i,$ "*authch*");
  nc_i=nc';
end for
for all i: *chal-res* of Id_i in buffer **do**
  $DK_{i,j-1}$ = PRF(PK_i, nc_i, ns_i);
  if $\mathrm{MAC}_{PK_i}(\mathrm{Id}_i, nc', ns_i,$ "*authres*") && $\mathrm{MIC}_{DK_{i,j-1}}$ verified **then**
    ns_i=refresh(ns_i);
    $DK^{\dagger}$=$H^{\alpha_i}(DK_{i,j-1})$
    send *auth-res*: $ns_i \mid nc' \mid DK^{\dagger}$;
    make SC_i:
    for $e = 1..n$ **do**
      $\mathrm{MAC}_{PK_i}(\mathrm{Id}_i, ns_i, DK_{i,j-1},$ "*reauth*");
      $DK_{i,j}$=PRF(PK_i, $DK_{i,j-1}$, ns_i);
      AVP_e:(Id_i, ns_i, MAC, $DK_{i,j}$) $\in \bigcup_{1..e-1}$ AVP;
      ns_i=refresh(ns_i);
    end for
  end if
end for

A MAC, including nonce nc' from the received message, is computed and sent to the STA of Id_i. The MAC allows the STA to verify authenticity of the AS. DK is computed by calculating an n-bit key generating pseudo random function (PRF)—in most cases n=128 is sufficient—with PK and the previously exchanged nonces. A MIC provides a means of verifying authenticity once the associated MAC is verified successfully. A hashed domain key, $\mathrm{DK}^{\dagger}$, is generated by applying α times a cryptographic one-way function H, equivalently $H^{\alpha}(x) = H^{\alpha-1}H(x)$ and $H^{0}(x) = x$. The α value is a sync-one shared between the STA and the AS/SCN. $\mathrm{DK}^{\dagger}$ allows DK to be hidden from authenticators. After the message exchanges, the server module creates the STA's security context that is composed primarily of the set of authentication value pairs (AVPs). It is then transferred to the corresponding SCN. The AVPs enable the SCN to conduct the re-authentication and re-keying process on behalf of the AS.

3.5.2 MAP SC Module

This is invoked on demand for re-authentication and is to be combined with the server module.

for all i: *reauth-req* of Id_i in buffer **do**
  AVP_l=(Id_i, ns_i, MAC, $DK_{i,j}$) $\leftarrow \bigcup_{l..n}$ AVP;
  if MAC && $\mathrm{MIC}_{DK_{i,j}}$ verified **then**
    send *reauth-res*: $nc' \mid ns_i \mid H^{\alpha_i}(DK_{i,j})$;
  end if
end for

The SC module first retrieves one of AVPs from the security context corresponding to Id_i and then verifies MAC$\neq$MAC$'$ or $\mathrm{MIC}_{DK_{i,j}}$(*reauth-req*)$\neq$MIC$'$. If not, it computes $\mathrm{DK}^{\dagger}$ and sends authenticator it along with the exchanged and retrieved nonces. If DK is not allowed to be re-used, the AVP is dethroned when it is notified somehow that the STA of Id_i de-associates with the current AP. If no more AVP exists, the re-authentication request is forwarded to the AS which will, in turn, handle the request from scratch. Note that the SC module does not possess any PK.

3.5.3 Authenticator

This binds with each AP and is responsible for generating an TK with which the STA and AP establish a secure link after authentication is performed successfully.

var: *na*;
if *auth-req* | *auth-chal* | *chal-res* | *reauth-req* received **then**
  relay it;
end if
if *auth-res* | *reauth-res* received **then**
  if success in authentication **then**
    na=refresh(*na*);
    send *auth-res*: $ENC_{DK^{\dagger}}[ns' \mid na \mid nc']$;
    TK=PRF($DK^{\dagger}$, $Addr_{STA} \mid Addr_{AP}$, $na \mid nc'$);
  end if
end if

Authenticator is beyond access to DK; $\mathrm{DK}^{\dagger}$ received from the AS/SCN is used to compute TK by calculating a PRF—the key-size varies with cryptographic protocols to be used for securing a wireless link, yet it is either 256 or 512 bits. TK binds with media access control addresses of the STA and AP; de-association revokes TK and a new TK is to be recomputed.

3.5.4 MAP Client Module

This is invoked when the STA (re)associates with an AP.

var: *secret* := 0, *nc*;
if (re)associated **then**
  nc=Refresh(*nc*);
  if !*secret* **then**
    send *auth-req*: Id | *nc*;
  else
    DK_i=PRF(*PK*, DK_{i-1}, *secret*);
    send *reauth-req*: Id | MAC_{PK}(Id, *secret*, DK_{i-1}, "*reauth*") | *nc* | MIC_{DK_i};
  end if
end if
if *auth-chal* received **then**
  if MAC_{PK}(Id, *nc*, *ns*$'$, "*authch*") verified **then**
    DK_{i-1}=PRF(*PK*, *nc*, *ns*$'$);
    nc=Refresh(*nc*);
    send *chal-res*: Id | *nc* | MAC_{PK}(Id, *nc*, *ns*$'$, "*authres*") | $\mathrm{MIC}_{DK_{i-1}}$;
  end if
end if

0-7695-2369-2/05 $20.00 © 2005 IEEE

if *auth-res* | *reauth-res* received **then**
  $DK^{\dagger}$=H^{α}(DK_{i-1} or DK_i);
  $DEC_{DK^{\dagger}}[ENC[ns' \mid na' \mid nc']]$;
  if nc==nc' **then**
    secret=ns';
    TK = PRF($DK^{\dagger}$, $Addr_{STA} \mid Addr_{AP}$, $na' \mid nc$);
  end if
end if

It retains the value of *secret*. It is given by the AS after completion of the previous successful authentication. Confidentiality of the secret is guaranteed since it is transferred in ciphertext. The secret determines whether the authentication process is conducted from scratch. The α value is matched to that of the AS/SCN.

4. Security Considerations

In this section, we first examine security threats to our protocol and then, using BAN logic, we show the logical proof that MAP performs its authentication mechanism correctly as it is expected.

4.1. Possible Attacks

Key recovery attack: This relies on finding the key K itself from a number of message–MAC pairs. Ideally, any attack allowing key recovery requires about 2^k operations where k is the length of K. The adversary tries all possible keys with a small number of message–MAC pairs available. Choosing a sufficiently long key is a simple way to thwart a key search. Another simple attack is to choose an arbitrary fraudulent message and append a randomly-chosen MAC value. Ideally, the probability that this MAC value is correct is equal to $1/2^m$, where m is the number of bits in the MAC value. Repeated trials can increase the corresponding expected value, but a good implementation will be alert to repeated MAC verification errors.

Forgery attack: This attack relies on prediction of $\text{MAC}_K(x)$ for a message x without initial knowledge of K. For an input pair (x, x') with $\text{MAC}_K(x)$=$g(H)$ and $\text{MAC}_K(x')$=$g(H')$, where g denotes the output transformation and H is a chaining variable, a collision occurs if $\text{MAC}_K(x) = \text{MAC}_K(x')$. Its feasibility depends on an n-bit chaining variable and m of the MAC result. Given g that is a permutation, a collision can be found using an expected number of $\sqrt{2} \cdot 2^{n/2}$ known text-MAC pairs of at least two divided blocks each. A simple way to counter this attack is to ensure that each sequence number at the beginning of every message is used only once within the lifetime of the key.

Impersonating attack: Note that the AUTH, SCN and AS keep a security association with each other. Therefore, neither of them can be used to impersonate the other. Instead, this attack occurs between the STAs and AUTHs. This attack causes an authentication failure or misconduct of the principals. Oracle-based impersonating attacks are that the attacker exploits one of principals as an oracle to obtain cryptographic messages in a session since it has no knowledge of K. The attacker applies the obtained messages to the other principal party in another session. For example, it runs a session with an AUTH to obtain a MAC value, impersonating a legitimate STA. It runs another session with an STA and exploits the MAC value on the STA, impersonating the legitimate AUTH. This attack can be countered by exchanging nonce each other and using a sequence counter.

4.2. The Protocol Analyzed

The analysis procedure is the following: first, translate the original protocol into the idealized one and then make assumptions about the initial state. Finally, we make logical formulas as assertions and apply the logical postulates to the assumptions and assertions to come to the conclusion in beliefs held.

Translating; we extract the encrypted forms of messages from MAP communications as follows:

i-1. B $\rightarrow$ A: $< N_a, N_b >_{PK}$

i-2. A $\rightarrow$ B: $< N_b, N_a >_{PK}$

i-3. B $\rightarrow$ A: $\{N_{b'}, N_{a'}\}_{DK}$

ii-1. A $\rightarrow$ B: $< N_{b'}, A \overset{DK}{\longleftrightarrow} B >_{PK}$

ii-2. B $\rightarrow$ A: $\{N_{b''}, N_{a''}\}_{K_{ab'}}$

We have STA and SCN—the functionality of AS and AUTH is integrated into SCN for simplicity, respectively, A and B, and omit communication in clear-text. In addition, there is a slight difference by representing $(N_a \oplus N_b)$ as (N_a, N_b), which is acceptable since this means that N_a and N_b were uttered at the same time and their XOR-ed value is straightforwardly obtained.

To authenticate the party, each party verifies the MAC which requires the nonces generated by itself and the other. That is, the correct MAC can only be generated with the fresh nonces from the two. Thus, we might deem that authentication is completed between A and B if each of the two believes that the other has recently sent the nonce, and proving the sound mutual authentication is sufficiently satisfied by deriving the following:

$$A \text{ believes } B \text{ believes } N_a \text{ and } B \text{ believes } A \text{ believes } N_b$$

for initial authentication and

$$A \text{ believes } B \text{ believes} N_{a''} \text{ and } B \text{ believes } A \text{ believes } N_{b'}$$

for re-authentication.

Making assumptions; we then write the following assumptions:

(1) A *believes* $A \overset{PK}{\Longleftrightarrow} B$, (2) B *believes* $A \overset{PK}{\Longleftrightarrow} B$,

(3) A *believes* $A \overset{DK}{\longleftrightarrow} B$, (4) B *believes* $A \overset{DK}{\longleftrightarrow} B$,

(5) A *believes* $A \overset{DK'}{\longleftrightarrow} B$, (6) B *believes* $A \overset{DK'}{\longleftrightarrow} B$,

(7) A *believes* $fresh(N_a)$, (8) B *believes* $fresh(N_b)$,

(9) A *believes* $fresh(N_{a'})$, (10) B *believes* $fresh(N_{b'})$,

(11) A *believes* $fresh(N_{a''})$, (12) B *believes* $fresh(N_{b''})$,

(13) A *believes* $fresh(N_{b'})$, (14) A *believes* $fresh(N_{b''})$,

(15) A *believes* B *controls* $N_{b'}$,

(16) A *believes* B *controls* $N_{b''}$.

The assumptions (1) and (2) are made from the fact that A and B initially share a secret, PK. (3), (4), (5) and (6) are derived from the fact that only A and B can generate a shared key only if the sound authentication is achieved. The assumptions from (7) to (12) show that A and B, each other, believe that nonces generated by themselves are fresh; freshness of nonces is hold by verification of MAC and MIC associated with the nonces. The nonces, $N_{b'}$ and $N_{b''}$ also play a role of secrets since they are transferred in encryption. Thus, A can believe that B has generated the nonces that have not been used in the past. It leads to the assumptions (13) and (14), and also (15) and (16) which indicate that A trusts B to generate the secret.

Reasoning; we analyze the idealized version of MAP by applying logical postulates presented in Section 2.5 to the assumptions.

A receives Message i-1. The annotation rule yields that A *sees* $< N_a, N_b >_{PK}$ holds afterward. With the hypothesis of (1), the message-meaning rule for shared secrets applies and yields A *believes* B *said* (N_a, N_b). Breaking conjunctions produces A *believes* B *said* N_a. With the hypothesis of (7), we apply the nonce-verification rule and yield A *believes* B *believes* N_a. On the other hand, B receives Message i-2 and the following result is obtained in the same way as that of Message i-1, via the message-meaning and nonce-verification rules with the hypothesis of (2) and (8), respectively, B *sees* $< N_b, N_a >_{PK}$ and B *believes* A *believes* N_b. This concludes the analysis of Message i-2. The analysis of Message i-1 and i-2 confirms that MAP performs mutual authentication successfully.

A receives Message i-3 and the annotation rule yields that A *sees* $\{N_{b'}, N_{a'}\}_{DK}$ holds afterward. The message-meaning rule for shared keys with the hypothesis of (3) via breaking conjunctions yields the following: A *believes* B *said* $N_{b'}$, and A *believes* B *said* $N_{a'}$. Taking the former, with the hypothesis of (13) and (15), the nonce-verification and jurisdiction rules apply, respectively and yield A *believes* B *believes* $N_{b'}$, and A *believes* $N_{b'}$, respectively. Taking the latter, the nonce-verification rule with the hypothesis of (9) yields A *believes* B *believes* $N_{a'}$. This concludes the analysis of Message i-3. This Message sounds like redundant since authentication is completed from Message i-1, yet this Message is essential not because it is for authentication, but because it is for the transmit of a secret, nonce $N_{b'}$.

B receives Message ii-1 and the annotation rule yields that B *sees* $< N_{b'}, A$ and $\overset{DK}{\longleftrightarrow} B >_{PK}$ holds afterward. By applying the message-meaning rule for the secrets with (2) via breaking conjunctions, we obtain as follows: B *believes* A *said* $(N_{b'}, A \overset{DK}{\longleftrightarrow} B)$, and B *believes* A *said* $N_{b'}$. The nonce-verification rule with the hypothesis of (10) yields that B *believes* A *believes* $N_{b'}$. On the other hand, A receives Message ii-2 and the annotation rule yields that B *sees* $\{N_{b''}, N_{a''}\}_{DK'}$ holds afterward. By applying the message-meaning rule for the shared keys with the hypothesis of (6) via breaking conjunctions, we obtain A *believes* B *said* $N_{b''}$ and A *believes* B *said* $N_{a''}$. Taking the former, the nonce-verification and jurisdiction rules with (14) and (16), respectively, yield A *believes* B *believes* $N_{b''}$, and A *believes* $N_{b''}$. Taking the latter, nonce-verification with (11) yields that A *believes* B *believes* $N_{a''}$. The analysis of Message ii-1 and ii-2 confirms that MAP also achieves mutual re-authentication.

5. Performance Evaluation

We evaluate the efficiency of MAP via experimentation and simulation, contrasting it with other protocols. We first discuss the simulation process and then analyze the MAP's performance benefits.

5.1. Simulation Methodology

The probe phase, discovering the next AP in WLAN handoffs, takes a long latency (ranging from 50 *ms* to 350 *ms*), depending on different vendors [21]. Even if the recent effort in [32] to reduce the latency by 84%, the large variance is an obstacle to highlight the effectiveness of our protocol on a real test bed. We therefore use simulation based on experimental data. We assume that network traffic is stable with small variations, *e.g.*, the latency of establishing a (re)association with an AP including the probe phase is 30 *ms* with 3% jitter, and the round-trip time (RTT) between two communicating servers across a domain is about 20 *ms* with 4% jitter. In addition, the RTT between the AP and SCN/AS is less than 3 *ms*. We use these values throughout the simulation. In cryptographic computations, we conducted an experiment using three machines: Linux v.2.4.19 iPAQ 206MHz ARM processor with 64MB (iPAQ), Linux v.2.4.2 Mobile Pentium 366MHz processor with 128MB (MP2) and Linux v.2.4.23 Intel Xeon 3Ghz bi-processor with 2GB (Xeon). We compiled codes [11] in gcc v.3.3 with an option of Level-1 optimization.

Table 2 shows the computation throughput of symmetric-key and public-key algorithms, respectively. With these

Table 2. Throughput of hash/symmetric and asymmetric algorithms

Alg.\Pow.	iPAQ	MP2	Xeon
SHA-1	15.8 Mbps	18 Mbps	104.9 Mbps
SHA-256	3.4 Mbps	9 Mbps	64.0 Mbps
SHA-512	0.2 Mbps	4.3 Mbps	24.8 Mbps
MD5	15.8 Mbps	41 Mbps	290.9 Mbps
AES-128	2.7 Mbps	10 Mbps	80 Mbps
RSA enc.	15.1 Kbps	138.9 Kbps	625 Kbps
RSA dec.	0.9 Kbps	4.6 Kbps	21.6 Kbps
RSA sig.	0.9 Kbps	4.4 Kbps	21.2 Kbps
RSA ver.	15.1 Kbps	138.9 Kbps	625 Kbps

measurement data, we numerically calculate the time to perform each authentication protocol while ignoring the overhead of running applications for simplicity.

5.2. The Simulation Model

Figure 2 shows the simulation model we used. Each AS constructs a domain consisting of an SCN and several APs. The SCN and AS may reside on the same machine as mentioned before. The domains are connected with secure links (between SCNs), based on an inter-domain roaming agreement.

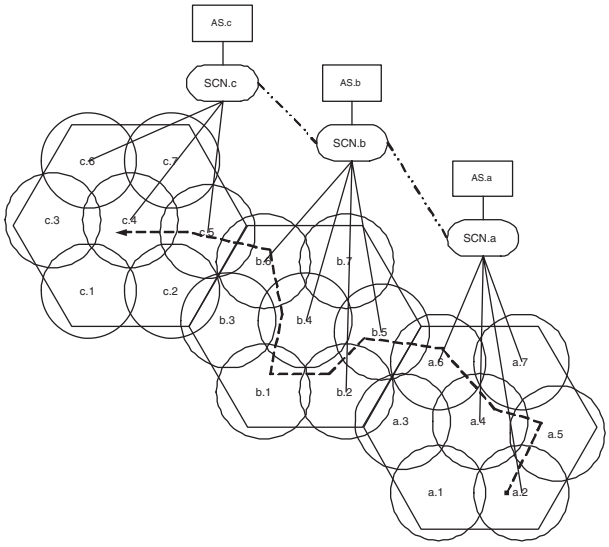

Figure 2. The simulation model for inter-domain handoffs

Handoff Pattern

Handoff pattern for STAs is basically random; the STAs cross the border after hopping a random number of times. Random pattern is sufficient to evaluate the overall efficiency performance. Nevertheless, to notice the comparative effectiveness of our protocol, we additionally set a regular handoff pattern; after association in the home domain, STAs hop three times and then cross a domain border. In visited domains, every five hops they traverse a domain.

SCN Configuration

When the STA crosses the border of a domain, there can be three system configurations according to the storage availability in the SCN of the visited domain. First, if only *relaying traffic is allowed*, the actual authentication falls in the AS/SCN of the home domain. The SCN in the visited domain serves as a relay agent. Second, if *caching security contexts is*, the foreign SCN serves as a proxy authentication server. In this case, security contexts are transferred and stored in the visited domain, which enables avoidance of contacting the home server. Third, if *pre-caching security contexts is allowed* somehow—that is, a security context is transferred to the foreign SCN before the STA arrives—then the latency of requesting the security context from the home server/SCN can be eliminated. Note that MAP can support this in conjunction with a protocol providing the localization of the security contexts.

5.3. The Simulation Results

MAP performs an optimized re-authentication procedure based on the security contexts generated after the initial authentication. It allows one to (1) compact the re-authentication procedure (with two-message exchanges, the mutual authentication is completed) and (2) avoiding contacts with the home server from the visited domain. Figure 3 clearly shows that from re-authentication, the authentication latency dramatically drops by up to 45% thanks to (1). As a regular handoff pattern, after three hops in the local domain (the first handoff corresponds to the initial authentication in the figure), the STA crosses the border of the domains at every 5 handoffs, which triggers the foreign SCN to request the security context from the home server. As a result, the latency increases in proportion to the RTT between the end-to-end points of two domains. Even if the STA roams in the foreign domain, it shows the same latency performance as in the home domain thanks to (2). In this case, the SCN in the foreign domain supports the caching allowed mode. After the 15-th handoff in the figure, the cross-domain authentication encounters the relaying allowed mode of the SCN in the visited domain, which triggers the authentication procedure to be performed in contact with the home server for

each hop in the visited domain.

Figure 4 shows the results with a random handoff pattern, illustrating the cumulative distributions of the authentication latency for three modes supporting SCN. The figure shows pre-caching and caching allowed modes to achieve more improvements in time efficiency than the relaying allowed mode which is characteristic of the legacy protocols that are unable to generate security contexts. For example, more than 70% of authentication processes in the caching allowed mode take less than 36 *ms* and more than 80% of those in the pre-caching allowed mode take less than 36 *ms*.

We evaluated the increase in storage availability via the number of authentication requests with a random handoff pattern. Figure 5 shows that the higher inter-domain handoff frequency the home SCN has, the higher its storage availability. The x-axis is the ratio of authentication request queries in inter-domain handoffs to the total number of queries, and the y-axis is the ratio of the network traffic in the foreign SCNs. Let AQ_r denote the foreign server's overhead and AQ_l denote the home server's overhead. Then, the ratio of the gain in storage availability with MAP to the overall overhead is expressed as, $1 - AQ_l/(AQ_l + AQ_r)$ which grows as the frequency of the inter-domain handoffs increases.

As shown in Figure 6 that plots the results with a random handoff pattern, the performance in authentication efficiency (caching allowed) improves up to 53% over a legacy method (relaying allowed) until the end-to-end domain distance continues to increase up to RTT=100 *ms*. In case of security context pre-cached in the visiting domain, MAP makes a 10% additional improvement with RTT=100 *ms*. Therefore, the effectiveness of MAP increases dramatically as the distance gets larger.

5.4. Comparison with Other Protocols

Figure 7 shows the cumulative CPU usage (represented in *ms*) cryptographic primitives of required in ten consecutive times of authentication in MAP and MNS and Kerberos protocols, and public-key-based Needham Schroeder (PNS) and TLS protocols. We chose one-way hash functions (MD5 [30], SHA [2]) and block ciphers (AES [4]) for symmetric-key protocols, and RSA [15] 1024-bit modulus for the public-key protocols. The symmetric-key protocols are shown to be two orders of magnitude faster thanks to the inherent advantage over modulo operations. MAP is faster than the MNS and Kerberos protocols, respectively, by 12.6% and 21.5% CPU usage gains. In view of millions of authentications conducted by a server, this is a considerable impact on the performance gain.

Regarding the number of message exchanges, MAP achieves the cross-domain authentication only with two-way handshake, the cost of which is minimal, compared to MNS and Kerberos requiring three-way and four-way handshakes, respectively. This contributes to the further enhancement of latency performance. Figure 8 shows the comparison of authentication latency of MAP with that of the MNS and Kerberos protocols while mobile nodes are hopping uniformly with a regular pattern. MAP outperforms the others in both inter- and intra-domain roaming. It accounts for 74% of cross-domain authentication latency of Kerberos and 85% of that of MNS. It reduces the intra-domain authentication latency by 5% for Kerberos and 7% for MNS.

5.5. Storage Overhead

The security context is transferred and stored in a foreign server (SCN) for cross-domain authentication. It consists mainly of a set of AVPs each of which is composed of nonce (128 bits), MAC (128 bits), DK (128 or 256 bits) and Identity (about 320 bits). In addition, a value (of 40 bits) may be reserved for security context validity and other information. The security context can be of $64 \cdot n + 45$ bytes where n is the number of AVPs. Approximately, given a 1 KB security context per STA, manipulating one million STAs requires a 1 GB storage, which is usually a small overhead to the server system.

6. Related Work

There have been several studies on how to achieve fast handoffs and enhance the performance of authentication mechanisms, including WLAN protocols.

Michra *et al.* [23, 22] presented a keys distributing method by means of proactive context caching. The idea of proactive caching is for an AP to broadcast its cached context to its neighbor APs in advance by using neighbor graphs and IAPP. However, this method is limited to intra-domain handoffs since APs are required to be functionally identical.

Pack *et al.* [29] presented a pre-authentication method that skips the 802.1X authentication phase by distributing the key to a certain number of selected APs and computing the likelihood based on the analysis of past network behavior. Bargh *et al.* [8] presented the applicability of the pre-authentication method for inter-domain handoffs. However, a pre-authentication method creates a higher risk of compromising security.

Wong *et al.* [35] proposed a hybrid protocol based on a certificate containing a symmetric key signed with a public key which is suitable for wireless communications. An asymmetric method for wireless communications presented in [14] uses Diffie-Hellman key exchange combined with Schnorr signatures. In addition, there are several legacy au-

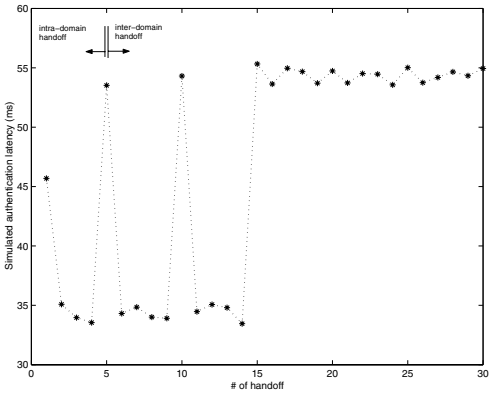

Figure 3. Authentication latency variations in different configurations of foreign servers

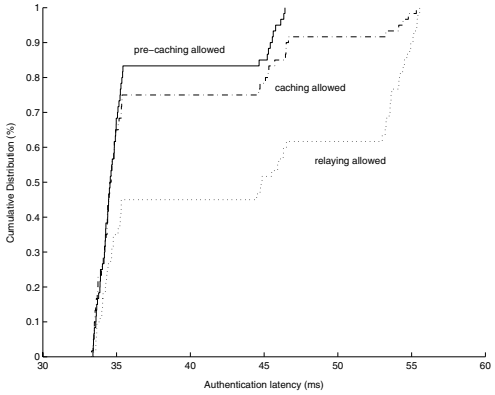

Figure 4. Cumulative distributions of authentication latency under each different configuration

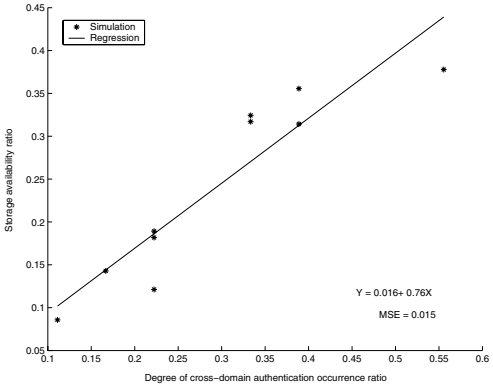

Figure 5. System storage availability affected by the inter-domain handoff authentication occurrence ratio. MSE is Mean Squared Error of the above regression function.

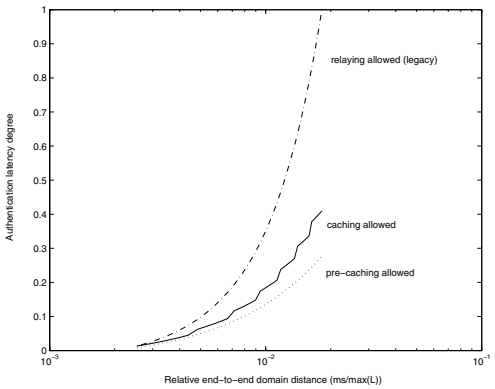

Figure 6. End-to-end domain distance vs. authentication latency: the distance is scaled down at a rate of the maximum authentication latency (max(L)).

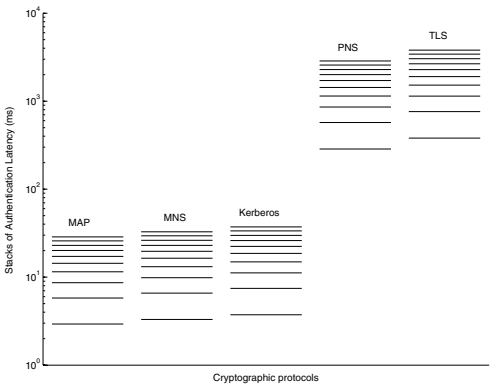

Figure 7. CPU utilization. Ten consecutive times of authentication.

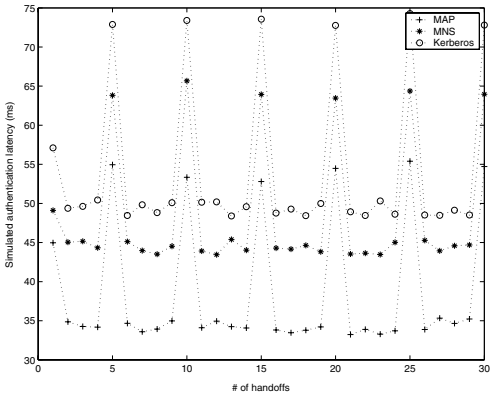

Figure 8. Latency comparison of MAP with MNS and Kerberos

thentication protocols [36, 37, 31, 10, 28, 12] for the general purpose in the literature.

There are several approaches to analyzing the security of authentication protocols. One is the formal methods that model and verify the protocol using specification languages and verification tools [20]. It consists of model checking and theorem-proving methods. Application examples [18, 24, 13, 19] demonstrated the feasibility of formally verifying the authentication protocols with general-purpose verification tools. Also proposed in [9, 34, 7] are modular approaches aiming to establish a sound formalization and a security analysis for the authentication problem.

7. Conclusions

The cross-domain authentication requires retrieval of security contexts from the server of the previously-visited or home domain. As we have shown, contacting a remote server may increase the authentication latency significantly. If security contexts are allowed to be pre-cached/transferred before the mobile user arrives, the latency can be reduced significantly: the longer the end-to-end distance, the larger the latency reduction.

In this paper we designed and evaluated a mobility-adjusted authentication protocol, MAP, by leveraging symmetric-key cryptography for cross-domain authentication and key distribution. MAP can be configured to make tradeoffs between performance and storage usage.

MAP introduces three concepts to the cross-domain authentication: (1) a re-authentication mechanism based on a two-way handshake; (2) the temporary-key generation of the 802.11i authentication; and (3) security contexts eliminating the need to contact a remote server.

MAP is much faster than Kerberos and MNS protocols, so that it performs better in cases of long end-to-end domain distances as well as high cross-domain authentication traffic.

Acknowledgments

We gratefully acknowledge support, feedback, and fruitful discussions with Robert De Simone. We would also like to thank Thierry Parmentelat and Mathieu Lacage for technical assistance of performing experiments, Martin Abadi and Mike Burrows for their comments on applying BAN logic and the anonymous reviewers for their insightful comments.

References

[1] Wi-fi alliance. http://www.wi-fi.org/.

[2] Secure Hash Standard. In *Federal Information Processing Standards Publication 180-1*. NIST, Apr. 1995.

[3] Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Specification for Robust Security. In *ANSI/IEEE Std 802.11: 1999(E)*. ISO/IEC 8802-11, 1999.

[4] Advanced Encryption Standard (AES). In *Federal Information Processing Standards Publication 197*. NIST, Nov. 2001.

[5] Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Specification for Robust Security. In *IEEE Std 802.11i/D3.1*. ISO/IEC 8802-11, 2003.

[6] B. Aboba and D. Simon. PPP EAP TLS Authentication Protocol. RFC 2716, Oct. 1999.

[7] W. Aiello, S. M. Bellovin, M. Blaze, R. C. J. Ioannidis, A. D. Keromytis, and O. Reingold. Efficient, DoS-Resistant, Secure Key Exchange for Internet Protocols. In *Conf. on Computer and Comm. Security*. ACM Press, 2002.

[8] M. S. Bargh et al. Fast Authentication Methods for Handovers between IEEE 802.11 Wireless LANs. In *Int. Workshop on Wireless Mobile App. and Services on WLAN Hotspots*, pages 51–60. ACM, Oct. 2004.

[9] M. Bellare, R. Canetti, and H. Krawczyk. A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols. In *30th Symposium on Theory of Computing*, pages 419–428. ACM Press, 1998.

[10] M. Burrows, M. Abadi, and R. Needham. A Logic of Authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[11] W. Dai. Crypto++, http://www.eskimo.com/∼weidai/cryptlib.html.

[12] T. Dierks and C. Allen. The TLS Protocol Version 1.0. RFC 2246, Jan. 1999.

[13] Heather and Schneider. Towards automatic verification of authentication protocols on an unbounded network. In *13th Computer Security Foundations Workshop*, page 132. IEEE Computer Society, 2000.

[14] M. Jakobsson and D. Pointcheval. Mutual Authentication for Low-Power Mobile Devices. In *Financial Cryptography 2001*, Grand Cayman Island, British West Indies, Feb. 2001.

[15] B. Kaliski. PKCS #1 RSA Encryption Version 1.5. RFC 2313, Mar. 1998.

[16] J. Kohl and C. Neuman. The Kerberos Network Authentication Service (V5). RFC 1510, Sep. 1993.

[17] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104, Feb. 1997.

[18] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 1055, pages 147–166. LNCS, 1996.

[19] P. Maggi and R. Sisto. Using spin to verify security properties of cryptographic protocols. In *9th SPIN Workshop on Model Checking of Software*, volume 2318, pages 187–204. LNCS, 2001.

[20] C. A. Meadows. Formal Verification of Cryptographic Protocols: A Survey. In *ASIACRYPT: Int. Conf. on the Theory and Application of Cryptology*, volume 917, pages 135–150. LNCS, Dec. 1994.

[21] A. Mishra, M. Shin, and W. Arbaugh. An Empirical Analysis of the IEEE 802.11 MAC Layer Handoff Process. *ACM SIGCOMM: Computer Communications Review*, 33(2):93–102, 2003.

[22] A. Mishra, M. Shin, and W. Arbaugh. Context Caching using Neighbor Graphs for Fast Handoffs in a Wireless Network. In *IEEE Infocom 2004*, Hong Kong, Mar. 2004.

[23] A. Mishra, M. Shin, and W. Arbaugh. Pro-active Key Distribution using Neighbor Graphs. *Wireless Comm. Magazine*, 11 Issue 1:26–36, Feb. 2004.

[24] J. C. Mitchell, M. Mitchell, and U. Stern. Automated analysis of cryptographic protocols using murϕ. In *Symposium on Security and Privacy*, pages 141–153. IEEE Computer Society, 1997.

[25] D. Mitton et al. Authentication, Authorization, and Accounting: Protocol Evaluation. RFC 3127, June 2001.

[26] R. M. Needham and M. D. Schroeder. Using encryption for authentication in large networks of computers. *Comm. of the ACM*, 21(12):993–999, 1978.

[27] R. M. Needham and M. D. Schroeder. Authentication Revisited. *ACM SIGOPS: Operating Systems Review*, 21(1):7, 1987.

[28] D. Otway and O. Rees. Efficient and timely mutual authentication. *ACM SIGOPS: Operating Systems Review*, 21(1):8–10, 1987.

[29] S. Pack and Y. Choi. Pre-Authenticated Fast Handoff in a Public Wireless LAN based on IEEE 802.1x Model. In *IFIP TC6 Personal Wireless Comm. 2002*, Singapore, Oct. 2002.

[30] R. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, Apr. 1992.

[31] M. Satyanarayanan. Integrating security in a large distributed system. *ACM Transactions on Computer Systems*, 7(3):247–280, 1989.

[32] M. Shin, A. Mishra, and W. Arbaugh. Improving the Latency of 802.11 hand-offs using Neighbor Graphs. In *ACM Mobisys 2004*, Boston, Jun. 2004.

[33] R. Shirdokar, J. Kabara, and P. Krishnamurthy. A QoS-based Indoor Wireless Data Network Design for VoIP. In *Vehicular Technology Conf. (VTC'01)*, volume 4. IEEE, Oct. 2001.

[34] D. S. Wong and A. H. Chan. Efficient and Mutually Authenticated Key Exchange for Low Power Computing Devices. In *ASIACRYPT: Int. Conf. on the Theory and Application of Cryptology and Information Security*, volume 2248, pages 272–289. LNCS, 2001.

[35] D. S. Wong and A. H. Chan. Mutual Authentication and Key Exchange for Low Power Wireless Communications. In *MILCOM 2001*, pages 39–43, USA, Oct. 2001. IEEE Press.

[36] T. Y. C. Woo and S. S. Lam. A Lesson on Authentication Protocol Design. *ACM SIGOPS: Operating Systems Review*, 28(3):24–37, 1994.

[37] Y. Zhang, C. Wang, J. Wu, and X. Li. Using SMV for cryptographic protocol analysis: a case study. *ACM SIGOPS: Operating Systems Review*, 35(2):43–50, Apr. 2001.