Fast Authentication for Mobility Support in Wireless Mesh Networks

Celia Li and Uyen Trang Nguyen
Department of Computer Science and Engineering
York University, Toronto, Ontario M3J 1P3, Canada
Email: {cli, utn}@cs.yorku.ca

***Abstract*—We propose new authentication protocols that support fast hand-off for real-time applications such as voice over IP and audio/video conferencing in wireless mesh networks (WMNs). A client and a mesh access point (MAP) mutually authenticate each other using one-hop communications. The central authentication server is not involved during the handover process. Fast authentication for roaming from one MAP to another is supported by using tickets. Our performance analysis, simulation results and security analysis show that our proposed authentication protocols are efficient and resilient to various kinds of attacks.**

I. Introduction

A WMN consists of the following major components [1]:
• *mesh clients*. They can be static (e.g., desktops, database servers) or mobile hosts (e.g., cell phone, PDAs).
• *mesh points* (MP). The MPs form a wireless mesh backbone to provide multi-hop connectivity from one mesh client to another or to the Internet. A subset of mesh points act as *mesh access points* (MAPs), connecting mesh clients to the WMN.

The current version of WMN standards 802.11s does not specify any mechanisms/protocols that support fast hand-off for mobile clients running real-time applications such as voice over IP (VoIP), newscast and tele-conferencing [2]. A mesh client has to be authenticated by an authentication server via *multi-hop wireless communications*, which may result in long delay, low reliability and thus potential service interruption. A performance study of mobile handoff delay in 802.11-based WMNs by Srivatsa and Xie [3] shows that as the number of wireless hops between two parties increases from one to five, the end-to-end delay increases from 0.15 second to 0.8 second. Since the authentication process involves several messages (e.g., nine messages in the EAP-TLS protocol used by 802.11s), the handoff latency may be several seconds long.

Our work in this paper contributes towards extending the IEEE 802.11s standards to support fast roaming for mobile clients. In particular, we focus on fast authentication during the hand-off process as well as during the initial login time. We propose ticket-based [5] authentication protocols that are efficient and resilient to attacks. The authentication server does not need to be involved in the handover authentication. Instead, mobile clients' authentications are done by mesh access points, avoiding multi-hop wireless communications. Fast authentication from one MAP to another during the hand-off process is supported using tickets [5]. Numerical analysis and simulation results show that our login authentication protocol improves the latency of 802.11s login authentication, and our handover authentication protocol supports fast authentication during the hand-off process.

The remainder of the paper is organized as follows. Related work is discussed in Section II. We describe the ticket types used in the proposed authentication protocols in Section III. In Section IV, we present our login and handover authentication protocols, along with a security analysis of the protocols. Performance evaluations of the proposed protocols are given in Section V. Section VI concludes the paper.

II. Related Work

Several authentication protocols have been proposed for wired networks such as Kerberos [5] and SSL [6]. Kerberos uses symmetric key methods, which are ideal for network environments where all services and clients are known in advance. This is usually not the case in a WMN where clients may join, leave and move freely at will.

SSL uses public key methods (e.g., public key certificates) to perform authentication, which is ideal for secure communications with a large, variable user base that is not known in advance, such as the Internet. However, public key methods are computationally intensive and space consuming, which are not suitable for resource-constrained mobile devices.

The current IEEE 802.11i and 802.11s standards do not support fast re-authentication, or fast hand-off in general, when a client moves from one MAP (or network) to another [4].

In mobile IP and cellular networks, the foreign agent/network must communicate with a client's home agent/network via multi-hop communications to authenticate the client [7], [8]. This approach, if applied to WMNs, means multi-hop wireless communications and thus potential service interruption as discussed earlier.

The objective of our proposed authentication protocols is to support fast authentication during the login time as well as the hand-off process. The protocols are built upon a new trust model [9] and different types of tickets described next.

III. Proposed Ticket Types

Our work was inspired by the concept of ticket from Kerberos [5] and a Kerberos-assisted authentication scheme proposed by Pizada and McDonald for mobile ad-hoc networks [10]. A ticket serves as a pass that a user submits

to a system/network to allow it to verify the user's identity. Tickets offer better security, more convenience and faster authentication than traditional authentication schemes using passwords [11]. Tickets are issued and managed by ticket agents who are trusted by mesh clients and mesh points including MAPs to perform such tasks. There can be several ticket agents serving a network. Tickets are used to establish the trust between mesh clients and MAPs, among MAPs, and among clients (see our previous work in [9]). The lifetime of a ticket is determined by its issuer's policy.

Three types of tickets are used in our authentication protocols: client ticket, MAP ticket and transfer ticket. They are needed for mutual authentication between a client with a MAP when the client signs in the network, or roams to another MAP.

A. Client Tickets

A client applies for a client ticket from a ticket agent. The trust between a client and a ticket agent is established through their public key certificates issued by a central authority.

Following is the structure of a client ticket:

$$\boxed{T_C = \{I_C, I_A, \tau_{exp}, P_C, Sig_A\}}$$

- T_C: client ticket issued by ticket agent I_A.
- I_C: ID number of the client that is given this ticket.
- I_A: ID number of the ticket agent who issued the ticket T_C.
- τ_{exp}: expiry date and time of ticket T_C.
- P_C: public key of client I_C, which is used by a MAP to verify the signature signed by the client in the login authentication protocol (see Section IV-A).
- Sig_A: digital signature of ticket agent I_A, which gives a recipient reason to believe that the ticket was created by ticket agent I_A, and that it was not altered in anyway.

B. MAP Tickets

The operator of a mesh network applies for MAP tickets, one per MAP, and distributes them to the MAPs in the network. The operator is also responsible for requesting and distributing new MAP tickets before the current MAP tickets expire. Following is the structure of a MAP ticket:

$$\boxed{T_R = \{I_R, I_A, \tau_{exp}, P_R, Sig_A\}}$$

- T_R: MAP ticket issued by ticket agent I_A.
- I_R: ID number of the MAP that is given this ticket.
- I_A: ID number of the ticket agent who issued ticket T_R to MAP I_R.
- τ_{exp}: expiry date and time of ticket T_R.
- P_R: public key of MAP I_R, which is used by clients to verify the signature of beacons message sent by MAP I_R (see Section IV-B).
- Sig_A: digital signature of ticket agent I_A.

C. Transfer Tickets

A transfer ticket is used to establish the trust relationship between a MAP and a client when a client roams from one MAP to another. When a client device C first logs in to the network, it submits its client ticket to a nearby MAP M_1, which will authenticate the client. If the authentication succeeds, M_1 becomes the *home MAP* of C. (We borrow the terminology from mobile IP.) M_1 issues to C a nonce (a number used only once) for C to compute a shared secret key K_{MAC} and a transfer ticket. See step (1) in the diagram shown in Fig. 1, which shows the messages exchanged between the MAPs and client. When C roams to another MAP M_2, which we call a *foreign MAP*, it submits the transfer ticket to M_2 for authentication. The transfer ticket proves to the foreign MAP that client C has been successfully authenticated by its home MAP. The structure of a transfer ticket Θ_C is as follows:

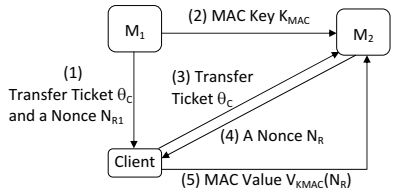


Fig. 1. Information exchange between a client and MAPs

$$\boxed{\begin{array}{l}\Theta_C = \{\mu, V_{K_{MAC}}(\mu)\}, \text{ where} \\ \mu = \{I_R, I_C, I_A, \tau_{exp}, MAC_{alg}\}\end{array}}$$

Message μ stores the information of the client, home MAP and ticket agent as follows:

- I_R: ID number of the MAP who issues this transfer ticket.
- I_C: ID number of the client who owns this transfer ticket.
- I_A: ID number of the ticket agent who issued C's client ticket.
- τ_{exp}: expiry date and time of this ticket.
- MAC_{alg}: message authentication code algorithm.

A foreign MAP will use the MAC algorithm indicated by field MAC_{alg} and value $V_{K_{MAC}}(\mu)$ to verify the authenticity and integrity of the transfer ticket Θ_C. Value $V_{K_{MAC}}(\mu)$ is the message authentication code produced by the MAC algorithm when applying key K_{MAC} to message μ. The operation of and the need for a MAC algorithm are explained below.

When client C moves into contact with a foreign MAP M_2, to prepare for a handover to the new MAP, C submits the transfer ticket issued by M_1 to M_2 for authentication (step (3) in Fig. 1). This handover authentication requires the following additional cryptography operations and keys:

- A shared key[1] between M_1 and M_2, which allows M_1 to *securely* send a message r containing the ID of client C and the secret key K_{MAC} to M_2 (step (2) in Fig. 1). (This K_{MAC} key is the same key that client C obtains during the login authentication process; see step (1) in Fig. 1.)
- Before sending the transfer ticket to client C, the home MAP M_1 applies the MAC algorithm [12] to message μ to produce a message authentication code denoted by $V_{K_{MAC}}(\mu)$. M_1 then

[1]Independently of authentication, a shared key is required between any two communicating MAPs in a mesh network, for encrypting/decrypting packets exchanged between them to combat attacks such as eavesdropping. This is called "key management" in wireless networks [13]. Our proposed authentication protocols simply use that shared key and the available key management protocol.

combines message μ and $V_{K_{MAC}}(\mu)$ to form the transfer ticket to be sent to C.

• Upon receiving both the message r sent by M_1 and the transfer ticket sent by C, M_2 verifies the authenticity and data integrity of the transfer ticket Θ_C by applying the MAC algorithm to message μ in Θ_C using the key K_{MAC} to produce a MAC. If this MAC matches $V_{K_{MAC}}(\mu)$ stored in the transfer ticket, then M_2 concludes that the ticket submitted by C is authentic. (M_2 will also verify the identity of C in the handover authentication protocol described in Section IV-B, and illustrated by steps (4) and (5) in Fig. 1.)

IV. The Proposed Authentication Protocols

We propose two authentication protocols, one for the initial login into a network and the other for subsequent roaming (handover).

A. Login Authentication Protocol (LAP)

The trust between a client and a MAP is established via their client and MAP tickets. Following are the order of the messages to be exchanged in the protocol and explanation:

(1) $R \longrightarrow C$: T_R
(2) $C \longrightarrow R$: $E_{P_R}(M_C)$, where $M_C = \{T_C, N_{C_1}, N_{C_2}\}$
(3) $R \longrightarrow C$: $E_{P_C}(M_R)$, where $M_R = \{N_{R_1}, N_{R_2}\}$
(4) $C \longrightarrow R$: $V_{K_{MAC}}(N_{R_2})$
(5) $R \longrightarrow C$: $\{V_{K_{MAC}}(N_{C_2}), \Theta_C\}$

(1) A MAP R periodically broadcasts beacon messages which contains its MAP ticket *to inform mesh clients and neighboring MAPs of its presence and ID*. Client C verifies the digital signature of the ticket agent A who issued the MAP ticket T_R using A's public key. C also verifies other information in the MAP ticket such as the ID of the ticket agent and the ticket expiry date.

(2) If the above verifications are successful, C extracts the MAP's public key from the MAP ticket T_R (see Section III-B) and generates a message M_C which contains C's client ticket T_C and two nonces N_{C_1} and N_{C_2}. C then encrypts the message using the MAP's public key ($E_{P_R}(M_C)$) and sends the encrypted message to the MAP R.

Upon receiving the message, R decrypts it using its private key, and verifies the digital signature of the ticket agent who issued the client ticket T_C (using the ticket agent's public key). R then verifies other information recorded in the client ticket T_C such as the ID of the ticket agent who issued T_C and the ticket expiry date.

(3) If the above verifications succeed, MAP R retrieves the client's public key from ticket T_C (see Section III-A), and generates a message M_C containing two random numbers N_{R_1} and N_{R_2}. R then encrypts message M_C using the client's public key ($E_{P_C}(M_R)$), and sends the encrypted message to client C. C will decrypt the message using its private key to retrieve N_{R_1} and N_{R_2}. Both the client and the MAP then calculate their shared MAC key K_{MAC} by applying a hash function H (e.g., SHA-1 [14], SHA-2 [15], MD5 [18]) to the message $\{N_{C_1}||N_{R_1}\}$, where the operator $||$ denotes a concatenation, and N_{C_1} and N_{R_1} are the random numbers generated in steps (2) and (3) above. That is, $K_{MAC} = H(N_{C_1}||N_{R_1})$.

(4) Client C then uses the key K_{MAC} and applies a (predetermined) MAC algorithm on N_{R_2} (created in step (3)) to produce a message authentication code $V_{K_{MAC}}(N_{R_2})$, which C then sends to the MAP. Upon receiving this message authentication code, the MAP performs the same computation as C just did to produce a message authentication code $V'_{K_{MAC}}(N_{R_2})$. If $V'_{K_{MAC}}(N_{R_2}) = V_{K_{MAC}}(N_{R_2})$, then the MAP has successfully authenticated the client C, because only C has the knowledge of the shared key K_{MAC} and N_{R_2}.

(5) To allow the client to authenticate the MAP, R applies the MAC algorithm and key K_{MAC} on the random number N_{C_2} (generated by C in step (2)) to produce a message authentication code $V_{K_{MAC}}(N_{C_2})$. The MAP also creates a transfer ticket Θ_C for C, and subsequently sends a message containing both the message authentication code and the transfer ticket to C.

When this message reaches the client, C carries out the same MAC computation as the MAP did to obtain a message authentication code $V'_{K_{MAC}}(N_{C_2})$. If $V'_{K_{MAC}}(N_{C_2}) = V_{K_{MAC}}(N_{C_2})$, client C has successfully authenticated the MAP. C will use the transfer ticket Θ_C to roam in the network.

B. Handover Authentication Protocol (HAP)

When a client C wishes to move from one MAP to another, e.g., from M_1 to M_2, it first sends a request to M_1 informing it of the intention [19]. M_1 subsequently sends to M_2 a message $r = \{I_C, K_{MAC}\}$ encrypted with the shared key of M_1 and M_2 (see the foot note on the previous page). Message r contains the ID of C, I_C, and the MAC key K_{MAC} for M_2 to verify C's transfer ticket (see Section III-C). C then sends its transfer ticket to M_2 to prepare for switching to M_2. (As an alternative implementation, M_2 acknowledges the receipt of message r using a broadcast. Upon hearing the acknowledgment, C submits its transfer ticket to M_2.)

Following is the handover authentication protocol according to the order of the messages exchanged:

(1) $C \longrightarrow M_2$: $\{\Theta_C, N_C\}$
(2) $M_2 \longrightarrow C$: $\{V_{K_{MAC}}(N_C), N_R\}$
(3) $C \longrightarrow M_2$: $V_{K_{MAC}}(N_R)$

(1) Client C sends its transfer ticket Θ_C and a nonce N_C to the foreign MAP M_2. Recall from Section III-C that a transfer ticket consists of two parts: the relevant information stored in a message μ and a message authentication code $V_{K_{MAC}}(\mu)$, which is the result of applying a MAC algorithm and a MAC key to message μ. Also, M_2 receives from the home MAP M_1 a message r storing the client's ID and the MAC key K_{MAC} which M_1 used to generate the message authentication code $V_{K_{MAC}}(\mu)$. M_2 verifies the content of the transfer ticket, especially the ID of the client's ticket agent and the ticket expiry date. It then applies the MAC algorithm and the MAC key received from M_1 to message μ to output a message authentication code $V'_{K_{MAC}}(\mu)$. If $V'_{K_{MAC}}(\mu) = V_{K_{MAC}}(\mu)$,

M_2 concludes that the transfer ticket is valid (i.e., C was successfully authenticated by its home MAP).

The above verification, however, does not prove C's identity. The following steps enable M_2 to verify C's identity.

(2) M_2 uses the MAC algorithm and key K_{MAC} on the nonce N_C to produce a message authentication code $V_{K_{MAC}}(N_C)$, which M_2 sends to client C along with a nonce N_R.

When C receives the message $\{V_{K_{MAC}}(N_C), N_R\}$ from M_2, it performs the same MAC computation as M_2 did to obtain $V'_{K_{MAC}}(N_C)$. If this value matches $V_{K_{MAC}}(N_C)$, the client has successfully authenticated the foreign MAP.

(3) Client C then executes the MAC algorithm using the MAC key K_{MAC} it computed in step (3) of the log-in authentication (Section IV-A), and the nonce N_R as input. The result is a message authentication code $V_{K_{MAC}}(N_R)$, which C will send to M_2. Upon receiving $V_{K_{MAC}}(N_R)$, M_2 repeats the same MAC calculation on N_R. If it obtains the same message authentication code as $V_{K_{MAC}}(N_R)$, then this proves C's identity since C is the only client who has the knowledge of the key K_{MAC}.

It should be noted that

• If the foreign MAP M_2 receives the transfer ticket Θ_C before the message $r = \{I_C, K_{MAC}\}$ from the home agent (Section III-C), M_2 will not be able to verify the validity of the transfer ticket because it does not have the MAC key K_{MAC} in order to apply the MAC algorithm to the ticket. In that case, M_2 sends back an error message to C and C who will initiate a log-in authentication instead of handover authentication. In this worst-case scenario, the handover authentication reverts back to the current practice in WMNs, i.e., repeating the login authentication with the foreign MAP. However, with careful design of message distribution (as future work) and low to moderate mobility speeds, we expect that this worst-case scenario does not happen often, and the handover authentication protocol will be used in most cases.

• After M_2 receives message $r = \{I_C, K_{MAC}\}$ from the home MAP, it also propagates this message to its neighbors to prepare for client C's future move to another MAP, say M_3. M_3 will use message r and the transfer ticket submitted by C to authenticate C as described above.

The handover authentication protocol does not use digital signatures or public key cryptography, but rather a MAC algorithm, to minimize authentication latency.

C. Security Analysis of the Authentication Protocols

In this section, we describe the countermeasures implemented in the proposed authentication protocols against the attacks listed in [20] that are relevant to our protocols.

• *Replay attack*: The attacker records messages of a successful authentication and replays these messages in an attempt to be successfully authenticated and gain access to the network. We prevent this type of attack by using message encryption and nonces. Consider an example in which an attacker attempts to impersonate a MAP by capturing and retransmitting a beacon message in step (1) of the login authentication protocol in Section IV-A. The attacker should not be able to modify the content of the original MAP ticket in the beacon message, thanks to the ticket agent's digital signature in the original MAP ticket. A client C will respond to the attacker's beacon message with a message $M_C = \{T_C, N_{C_1}, N_{C_2}\}$ encrypted using the legitimate MAP's public key. The attacker will not be able to decrypt this message since it does not have the corresponding private key. Without the knowledge of the nonces N_{C_1} and N_{C_2}, the attacker will not pass client C's verification in step (5). We can show in a similar manner that the replay of any message in the login or handover authentication protocol will fail the authentication.

• *Time-memory trade-off attack*: For a given hashed value of a password, the attacker can use partially pre-computed values in the hash space of a cryptographic hash function to guess the password. With pre-computation done offline, the time taken in the online stage is shortened at the expense of more memory required. SHA-1 [14] or SHA-2 [15], which are currently among the most secure hash functions, can be used in the hash-based MAC algorithm to prevent this type of attack.

• *Compromised MAPs*: A MAP may be compromised by an attacker which may (1) drop authentication messages to prevent clients from joining the network, or (2) grant access to unauthorized/non-paying users.

(1) Dropping valid messages deviates from the normal procedure of the authentication protocol, which requires the attacker to modify the authentication code. Software-based attestation techniques such as SWATT [17] and Pioneer [16] can be used to externally verify the contents of the memory of an embedded device (SWATT) or a CPU (Pioneer) in order to detect changes to the original code. An external verifier can detect with high probability if a single byte of the memory deviates from the expected value [17]. These techniques allow a network operator to periodically verifies the routers in its network and detect compromised nodes. Note that this attack can happen to any protocols (e.g., routing) and not just authentication. From a client's point of view, the attack consequence is similar to that of a router failure: the client times out on the authentication request, and will look for another MAP nearby to join. This type of router placement redundancy should be implemented regardless of security issues: if a MAP fails or malfunctions, nearby MAPs should be able to support its clients.

(2) To grant access to users that do not own valid tickets, the attacker would need to modify the authentication code. Thus one countermeasure is to use attestation techniques such as SWATT and Pioneer to detect changes in the authentication code, as discussed above. An alternative we propose is to use a dual authentication process. The authentications described in Section IV, if successful, give the client only short-term access to network services. The client will subsequently be authenticated by an authentication server (via multi-hop communications), while enjoying network services using the short-term access permission. After the server successfully authenticates the client, it will issue to the client a service ticket [5] that serves as a pass for the client to access network

services on a long-term basis. An illegitimate or non-paying user will not be issued such a service ticket, and will not be able to continue to use network services after the short-term access privilege expires. The dual authentication process allows both fast authentication during the handover step, and the stronger security provided be an authentication server. The effectiveness and performance of the dual authentication protocol will be evaluated in our future work.

V. Performance Evaluation

We compare our protocols with EAP-TLS [21], the public-key-based authentication protocol of IEEE 802.11s.

A. Numerical Analysis

The performance is measured in terms of

• *communication costs*, which indicate the number of messages exchanged between a MAP and a client to complete an authentication session.

• *computation costs*, which are the latencies (in milliseconds) incurred by the following security operations: encryption using public key (E_{pub}); decryption using public key (D_{pub}); generation of a digital signature (G_{sig}); verification of a digital signature (V_{sig}); computation/verification of a message authentication code (MAC); and hashing.

Table I lists the above operations, the current state-of-the-art algorithms implementing the operations, and the computation time each of these algorithms incurs [22] (the first, second and third columns, respectively). The fourth, fifth and sixth columns of Table I list the numbers of security operations the proposed login and handover authentication protocols and EAP-TLS perform, respectively. By multiplying the computation cost of each operation (from the third column) and the number of times it is executed, and summing up the costs of all operations executed by a protocol, we obtain its total computation cost as shown in the third last row of Table I. The computation cost of the login authentication protocol (97.944ms) is slightly less than that of EAP-TLS (97.962ms). But more importantly, the computation cost of the handover authentication protocol (0.09ms) is three orders of magnitude lower than that of the login authentication and EAP-TLS.

The second last row of Table I lists the number of messages exchanged in each protocol. The authentication latencies shown in the last row are the sums of computation costs and communication delays, where d is the average delay of a one-hop communication incurred by a message, and h is the number of hops between the client and the home authentication server. (Parameter h is applicable to only EAP-TLS as our handover protocol does not require a client to communicate with the home MAP during the hand-off process.) The results show that the larger the number of hops between a client's home MAP and a foreign MAP, the lower the authentication latency our protocols incur compared with EAP-TLS.

B. Simulation Results

We use the QualNet simulator version 4.5 [25]. The simulation parameters are listed in Table II. The clients are randomly placed while the MAPs are uniformly distributed in the network. To test the scalability of the protocols, we let all clients present in the network send authentication requests to their respective nearby MAPs simultaneously.

TABLE I
Computation and communication costs

Op.	Alg.	Time (ms)	Login sec. IV-A	Handover sec. IV-B	EAP-TLS
E_{pub}	RSA [23]	1.42	1	0	1
D_{pub}	RSA	33.3	1	0	1
G_{sig}	ECDSA [24]	11.6	1	0	1
V_{sig}	ECDSA	17.2	3	0	3
MAC	HMAC [12]	0.015	1	6	1
Hash	SHA-1 [14]	0.009	1	0	3
Total computation cost			97.944ms	0.09ms	97.962ms
Number of messages			5	3	9
Authentication latency			97.944+5d	0.09+3d	97.962+9dh

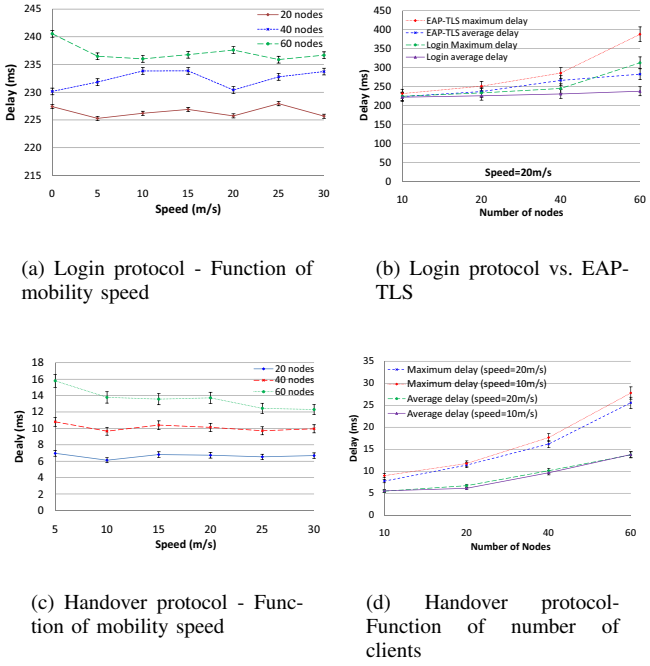
TABLE II
Simulation parameters

Experiment	Network	Clients, Mobility Speed
Fig. 2(a), login	300m x 300m, one MAP	20-60 nodes, 0-30m/s
Fig. 2(b), login vs. EAP-TLS		10-60 nodes, 20m/s
Fig. 2(c), handover	1500m x 1500m, four MAPs	20-60 nodes, 5-30m/s
Fig. 2(d), handover		10-60 nodes, 10-20m/s

The performance metric is *authentication delay* (latency), which is measured as the time between a client's transmission of an authentication request to a nearby MAP and the receipt of an acceptance confirmation. After a client sends an authentication request, it sets a timer. If it does not receive a confirmation by the time the timer expires, it will re-send the request. The authentication delay is measured starting with the first request. In all experiments, we calculate the *average authentication delay* (AAD), averaged over all mobile clients participating in the experiment. In several cases, we also keep track of the *maximum authentication delay* (MAD), the maximum value among all mobile clients. Each data point in the graphs is averaged from 10 runs using different random seeds, and plotted with a confidence interval of 95%.

The results are shown in Fig. 2. The graph in Fig. 2(a) shows the AAD of the *login authentication* protocol (LAP) as a function of clients' mobility speed. There is one MAP placed at the center of the network, serving 10-60 mobile clients. We observe that the AAD is not impacted much by the mobility speed, which is a positive attribute of the LAP. On the other hand, as the number of clients increases from 20 to 60, the ADD also increases as expected, by approximately 4% to 6%. More clients imply more authentication requests to be processed by the MAP, and more channel contention around the MAP, resulting in longer delay.

Fig. 2(b) shows the performance of the *login protocol vs. EAP-TLS* under the same network setting as above. When there are only 10 clients in the network, both protocols perform similarly. Given more than 10 clients, the workload and channel contention at the MAP increases. In these cases, the LAP offers lower AAD than EAP-TLS, because the LAP requires less messages exchanged than EAP-TLS (5 vs. 9, as shown in the second last row of Table I). In the case of 60



(a) Login protocol - Function of mobility speed

(b) Login protocol vs. EAP-TLS

(c) Handover protocol - Function of mobility speed

(d) Handover protocol- Function of number of clients

Fig. 2. Simulation results

clients, the AAD of the LAP is 16% lower than that of EAP-TLS. As the number of nodes increases, the performance gap between the LAP and EAP-TLS enlarges, consistent with the authentication latencies recorded in the last row of Table I (97.944+5d vs. 97.962+9dh, where $h = 1$). The graph also shows the MAD of both protocols. The MAD of the LAP is about 32% higher than its AAD, which we deem acceptable, and about 20% lower than the MAD of EAP-TLS.

The graph in Fig. 2(c) shows the AAD of the *handover authentication* protocol (HAP) as a function of clients' mobility speed. Four MAPs are uniformly distributed over the network, serving 10-60 mobile nodes. Again, the mobility speed does not have a big impact on the AAD of the HAP, as in the case of the LAP. Also, the more clients send requests, the higher the AAD, as expected. Note very low AADs of the HAP, ranging from 6ms-16ms, compared with the AADs of the LAP and EAP-TLS which are above 220ms.

The above observations also apply to Fig. 2(d), which shows the MADs and AADs of the *handover authentication* protocol as functions of number of clients. In the experiment with 60 nodes moving at a speed of 10m/s, the MAD of the HAP is 28ms, about twice as long as the corresponding AAD, but still very low compared with the authentication delay of EAP-TLS.

VI. Conclusion

We propose new authentication protocols to support fast hand-off in IEEE 802.11s networks. A client and a MAP mutually authenticate each other using one-hop communications. Fast authentication for roaming from one MAP to another is supported by using transfer tickets. The authentication server is not required to participate during the handover authentication process (but only after the client has joined the new MAP if dual authentications are implemented). Our numerical analysis and simulation results confirm that the proposed LAP and HAP outperform the EAP-TLS protocol of IEEE 802.11s. They are also resilient to various kinds of attacks. In our future work, we will extend the proposed protocols to support multiple network operators and multiple ticket agents, and evaluate the performance of the dual authentication approach.

References

[1] I. Akyildiz and X. Wang, Wireless Mesh Networks, Wiley, 2009.

[2] X. Wang and A. O. Lim, "IEEE 802.11s Wireless Mesh Networks: Framework and Challenges," Ad Hoc Networks Journal (Elsevier), vol. 6, no. 6, pp. 970-984, 2008.

[3] A. M. Srivatsa and J. Xie, "A Performance Study of Mobile Handoff Delay in IEEE 802.11-Based Wireless Mesh Networks," 2008 IEEE International Conference on Communications, pp. 2485-2489, 2008.

[4] IEEE, "Draft Amendment: ESS Mesh Networking," IEEE 802.11s Draft 1.00, 2006.

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

[6] D.Wagner and B.Schneier, Analysis of the SSL 3.0 Protocol, USENIX Workshop on Electronic Commerce Proceedings, pp. 29-40, 1996.

[7] Y. Jiang, C. Lin, X. Shen and M. Shi, "Mutual Authentication and Key Exchange Protocols for Roaming Services in Wireless Mobile Networks," IEEE Trans. on Wireless Comm., pp. 2569 - 2577, 2006.

[8] M. Buddhikot, et al., "Design and Implementation of a WLAN/CDMA 2000 Interworking Architecture," IEEE Comm. Magazine, 2003.

[9] Celia Li and U. T. Nguyen, "Fast Authentication for Mobile Hosts in Wireless Mesh Networks," Technical Report, Dept. of Computer Science & Engineering, York University, April 2010.

[10] A. A. Pizada and C. McDonald, "Kerberos Assisted Authentication in Mobile Ad-hoc networks," Conf. on Australian Computer Science, 2004.

[11] D. P. Jablon, "Password Authentication Using Multiple Servers," Topics in Cryptology, pp. 344-360, 2001.

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

[13] A. J. Menezes, P. C. Oorschot and S. A. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1996.

[14] C. S. Jutla and A. C. Patthak, "Is SHA-1 Conceptually Sound?" Cryptology ePrint Archive, Report 2005/350, http://eprint.iacr.org/, 2005.

[15] P. Hawkes, M. Paddon and G. Rose, "On Corrective Patterns for the SHA-2 Family," Cryptology ePrint Archive, Report 2004/207, 2004.

[16] A. Seshadri and M. Luk, "Verifying Code Integrity and Enforcing Untampered Code Execution on Legacy System," 20th ACM Symposium on Operating Systems Principles, 2005.

[17] A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla, "SWATT: Software-based Attestation for Embedded Devices," IEEE Symposium on Security and Privacy, 2004.

[18] B. D. Boer and A. Bosselaers, "Collisions for the Compression Function of MD5," Eurocrypt Conference, 1993.

[19] P. Goransson and R. Greenlaw, "Secure Roaming in 802.11 Networks," Elsevier, 2007.

[20] G. Horn, M. Martin and C. Mitchell, "Authentication Protocols for Mobile Network Environment Value-Added Services," IEEE Trans. on Vehicular Technology, pp. 383-392, 2002.

[21] D. K. K. Baek and S. W. Smith, "A Survey of WPA and 802.11i RSN Authentication Protocols," Technical Report, Department of Computer Science, Dartmouth College, 2004.

[22] M. Long, "Energy-efficient and Intrusion Resilient Authentication for Ubiquitous Access to Factory Floor Information," IEEE Trans. on Industrial Informatics, pp. 40-47, 2006.

[23] R. Rivest, A. Shamir and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems," Communication of the ACM, pp. 120-126, 1978.

[24] ECDSA, FIPS 186-3, Digital Signature Standard (DSS), 2009.

[25] QualNet simulator, http://www.scalablenetworks.com/.