Secure, fast handhoff techniques for 802.1X based wireless network

Leonardo Maccari, Romano Fantacci, Tommaso Pecorella
Department of Electronics and Telecommunications - University of Florence
Telecommunication Network Lab
tel. : +390554796467 - fax : +390554796485 Florence, Italy
Email: {maccari, fantacci, pecos}@lart.det.unifi.it

Federico Frosali
Telecom Italia Lab
Via G. Reiss Romoli, 274 - 10148 - Turin, Italy
tel: +39011 2285111 - fax: +39011 2285520
Email: federico.frosali@tilab.com

***Abstract*— Wireless networks that support client mobility have to face the challenge of providing a secure, performant handoff between different access points. IEEE 802.1X [1] model provides a secure mechanism used by many standard protocols to securely generate keying material between two peer hosts when one of the two is accessing the network for first time, but that is hardly usable for reauthentication during handoff procedures without loss of performance. This paper deals with the proposal of a novel scheme to transport authentication credentials during handoff that uses a two-way only exchange with the backend authentication server maintaining the security of the system. As a high-level method it can be applied to different types of network, such as IEEE 802.11i [2] infrastructure or ad-hoc mode networks in a mesh environment.**

I. Introduction

Authentication and keying matherial generation are extremely important in wireless networks, but are also time-consuming procedures. The IEEE 802.1X standard defines a paradigm to be used in a three-party authentication that has been accepted and implemented in layer II standard such as IEEE 802.11i or the forthcoming IEEE 802.16e. This paradigm is based on the presence of three entities:

- an Authentication Server, *AS*, basically a database containing the credentials needed to accept or deny access to clients and able to carry on a complete authentication with different cryptographic mechanisms
- an Authenticator, *AA*, offering layer two connectivity to end clients
- a Supplicant, *SA*, that is the client requesting access to the network.

Initial authentication is done between the *SA* and the *AS*, through the *AA* that acts as a trasparent proxy. During authentication keying matherial is produced into this two endpoints and then moved from *AS* to *AA*, normally using UDP based RADIUS protocol. From that keying matherial *AA* and *SA* generate local keys that will be used for actual criptography and authentication. In Fig. 1 there is a typical network environment with a unique *AS* serving more than one *AA*. Clients can freeley roam between one *AA* and another but according to IEEE 802.11i standard they have to repeat 802.1X authentication for each handoff. 802.1X standard defines Extensible Authentication Protocol Over LAN (EAPOL), that can be used to vehicle specific authentication methods, such as TLS [3]; MD5 password etc. Encapsulation of TLS protocol into EAP is defined in EAP-TLS protocol [4].

This work is partially supported by National project Wireless 8O2.16 Multi-antenna mEsh Networks (WOMEN) under grant number 2005093248.

Normally, 802.1X authentication in wireless networks is constituted of the following phases:

1) detection: layer II specific message exchange used by the *SA* to detect an *AA*
2) authentication and association: layer II specific message exchange needed to create and secure layer II control channel
3) EAPOL identity message exchange: is the first exchange that involves *AS* and is used to identify *SA* to *AS*
4) method specific messages: first message indicates the method to be used, such as TLS, following messages are the body of the authentication.
5) keying matherial exchange: if during previous phase keying matherial was generated, it can be moved from *AS* to *AA*
6) key negotiation: *SA* and *AA* derive encryption keys to secure data channel.

Phase 1, 2 and 6 are not 802.1X specific but are layer II dependant.

In this paper we illustrate a handoff mechanism that reduces the number of packet exchanged in the EAP phase maintaining the security level provided by the application of 802.1X mechanisms. We also define security requirements that we consider adequate for infrastructure and ad-hoc environments.

To illustrate the proposed solution we have first to describe in a detailed way an authentication procedure in 802.1X networks. Without loss of generality, from now on we will use 802.11i terminology to indicate keys and algorithms used, and assume that *AA* and *SA* use RADIUS protocol. Fig. 2 contains a message sequence chart depicting authentication envolving the three agents of a IEEE 802.11i, using EAP-TLS protocol for authentication and key generation. In the figure packet exchange is divided into blocks, representing the phases described before.

First block is 802.11 authentication and association, second block is EAPOL initiation, third block contains EAP-TLS specific method. Note that at this point *AS* and *SA* have created a common simmetric key named Pairwise Master Key (*PMK*) that is moved from *AS* to *AA* with the forth block. From this

common key *SA* and *AA* can derive other keys with protocol specific mechanisms, in 802.11i this is done with a 4-way handshake and a 2-way handshake represented in the fifth block. This last step derives from *PMK* two new keys, named Pairwise Transient Key and Group Transient Key (*PTK* and *GTK*); each one is used to cipher and authenticate respectively unicast and multicast traffic between *SA*'s and *AA*.

IEEE 802.1X defines clear roles that can be easily described with 802.11i terminology. Mutual authentication is a process that involves two entities, *AS* and *SA*, and its success is witnessed by the fact that both parties at the end own the same *PMK* key. *PMK* is actually never used as a cryptographic key for encryption or signature, it can be considered as an authentication token from which other cryptographic keys are generated. *AA* doesn't actively participate to *PMK* generation since it doesn't contain informations necessary to authenticate a client. Once completed the authentication phase *AS* moves the generated *PMK* to the *AA* with a specific RADIUS packet. This is possible if *AA* and *AS* already own a security association of some type, which is consituted by the symmetric RADIUS key.

When *AS* has accepted *SA* into the network it shares the *PMK* secret with *AA*, that is considered a trusted third party. From now on all communications take place between *AA* and *SA*, since *AA* is the layer II service provider. This means *AA* and *SA* need cryptographic keys to communicate, and those keys (*PTK* and *GTK*) can be derived from the *PMK* they share. *PTK* and *GTK* can be considered as keys bound to layer II links, that must be regenerated each time a new link is created while *PMK* is a *session key* testifying the authentication to the network. Compromission of *PTK* or *GTK* key means basically loss of confidentiality and authentication of frames exchanged on one link, while compromission of *PMK* means possible loss of access control over the whole network, since an attacker can use *PMK* keys to log into the network.

A handoff is the act of leaving the current access point to move towards a new one and it implies a new authentication. Infrastructure networks use a hierarchical model where one ore more Access point offers layer II connectivity to end clients, a handoff is normally needed when the client looses conectivity with the access point and must find another one. Mesh networks use a distributed ad-hoc model where every terminal is at the same time an end client and an Access Point for its neighbours. In mesh networks every machine is connected at the same time with more than one neighbour, and it plays both the roles of authenticator and supplicant. Due to the mobile and dynamic nature of such networks links are constantly falling and raising, requiring frequent handoffs. Total handoff time must consider time necessary for the exchanged depicted in fig 2 increased of detection time (time needed by a station to sense loss of conectivity), scan time (time needed to find a new access point) and in the case additional delay due to LLC layer procedures; in [5] a average of measurements of a complete 802.11 handoff time (detection + first block of figure 2) is said to vary between 58.74ms and 396.76ms depending

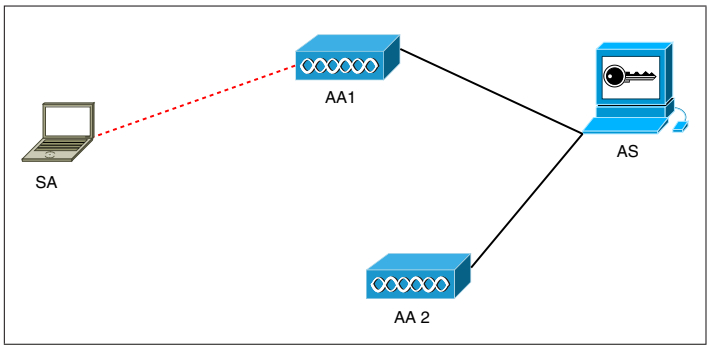

Fig. 1. 802.1X scenario, solid black lines are wired links, dotted red lines are wireless links

on hardware used. The second and forth blocks of packets are typical of a sessions of EAPOL protocol, which is embedded into 802.1X model, while third block is EAP specific method so its actual lenght varies depending on the chosen method. EAP-TLS is the shortest method providing certificate-based mutual authentication but it still needs a 6-way handshake to produce *PMK* key. It is important to note that Fig. 1 describes a very simple application of 802.1X standard, with a more complex topology such as an ad-hoc mesh network, the path from *AA* to *AS* can be much longer and could be composed with sub-optimal routing protocols for distributed networks. The actual delay introduced by the EAP specific method grows with the lenght of the path and with number of packet exchanged. This is not true for every exchange that envolves just *AA* and *SA*, which is a one-hop path.

II. Security issues in handoff strategies

During a handoff *SA* never leaves the network, so it should not be forced to repeat the generation of the *PMK* key, but it needs to create new cryptographic keys that it will use with the new access point to protect the link. The challenge resides in finding a fast and secure mechanism to move *PMK* key into the new access point.

To define security requirements we have to define first what kind of attacker we are trying to protect from. One crucial point is to decide if the attacker can be internal or we just want to consider protection from external attacks. An attack can be carried from the inside by a compromised machine or by a malicious user (being it a *SA* or a *AA*, or both like in a distributed mesh network where every machine plays both roles). As a *AA* the attacker can access a number of *PMK* legally generated and a RADIUS key that makes him a trusted party to the *AS*. We believe into the importance of building a system robust to attacks carried from the inside for two main reasons: first is that end users cannot generally be trusted, but in a distributed mesh environment end users are also access point for other users, so untrusted internal machines can elaborate sensitive informations; second is that even in infrastructure networks access point compromission is made easy by plenty of *exploits* that can be found on internet.

Consequently the solutions that we outline will be able to

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

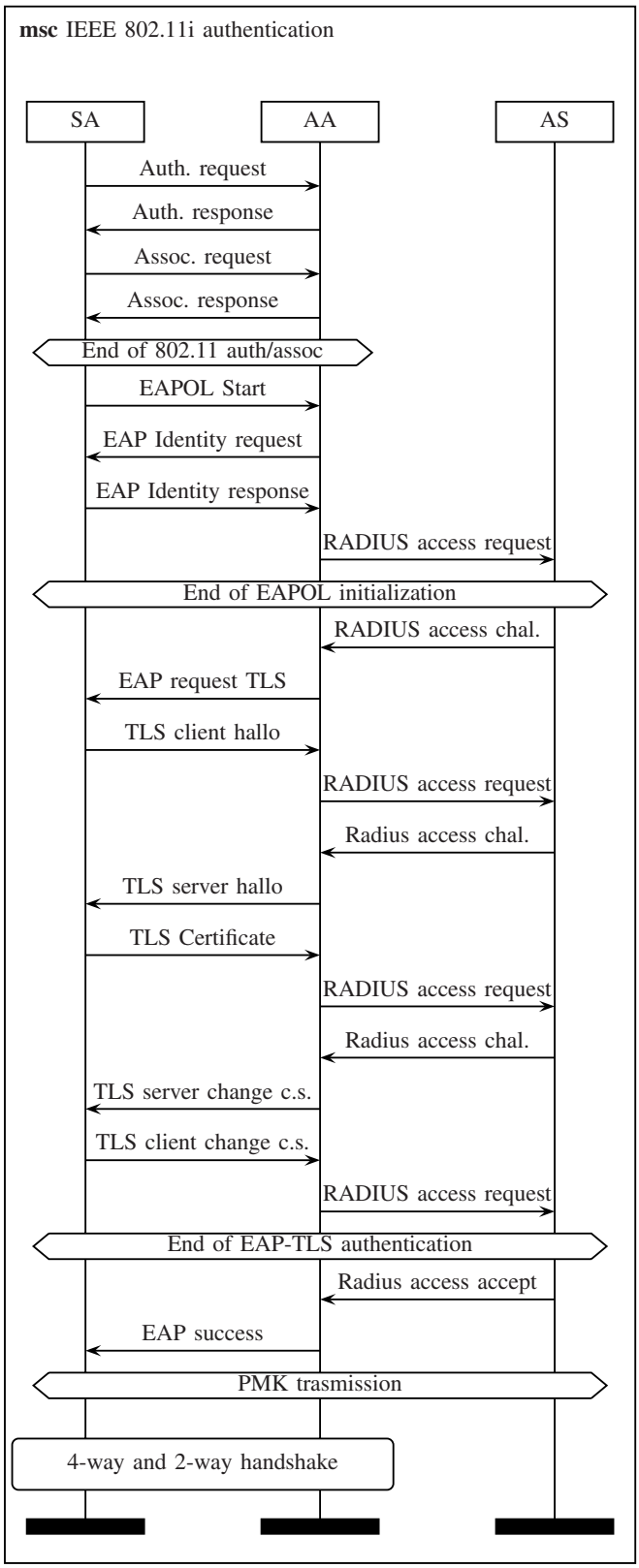

Fig. 2. Packet exchange sequence for authentication in 802.11i networks

maintain the security level reached by the 802.1X protocol against attacks coming from the outside, but also in presence of an internal attacker we plan to satisfy the following two security features:

1) access control must be respected. Compromission of a *AA* containing valid *PMK* keys must not have the consequence of loss of access control for the *AS*. Obviously a compromised machine could hide an illegal network directly connected to it and this is unavoidable but we want to avoid the possibility of generating more valid authentication tokens starting from the compromised one.
2) *PMK* secure distribution. A compromised machine must not be able to access other *PMK* keys. We want to limit the possibility of a *AA* to request *PMK* to its neighbors or to the *AS*. This limits the possible disclosure of informations due to compromission of an access point, since from the knowledge of a *PMK* key the attacker may easily obtain *PTK* and *GTK* keys and decipher sniffed traffic, carrying an on-line or even an off-line attack.

To stress the importance of this second feature, we might imagine an attack lead against a mesh network by multiple machines, some of which are actively trying to compromise at least one access point while the others collect traffic. If from a compromised access point the attacker can easily collect other *PMK* keys he can use sniffed traffic to generate *PTK* and *GTK* keys, making the attack retroactive. In [6] a multifence approach to wireless security is described, indicating that security in mobile ad-hoc networks must be guaranteed by multiple agents cooperating (authentication methods, firewall, IDS...) in proactive and reactive mode. Under this point of view we consider extremely important to slow down the propagation of attacks.

Referring to Fig. 1 where *SA* is migrating from AA_1 to AA_2 our goal is moving *PMK* key to AA_2 in an efficient way. Describing the procedure at a high level of abstraction we can imagine AA_2 *asking PMK* to some other machine, if we decide that AA_2 may ask and receive *PMK* keys from any neighbor access point (AA_1 in this case) we implicitly accept a drawback, that is: a compromised AP might be able to inject new *PMK* keys into the network, with the aim of introducing unauthenticated clients into the network, or producing a denial of service sending invalid *PMK* keys. This violates the first security feature stated above, and deprives the *AS* of its fundamental role in access control. If *PMK* keys can be obtained only from *AS* the previous problem is avoided but still it must be granted the second security feature, that is an AP cannot indiscriminately ask and receive *PMK* keys to *AS*, but it has to somehow demonstrate that it is in need of a certain key. Our solution fullfills these conditions.

It must be noted that if *AS* is always involved into handoff procedures, handoff total time is increased, since *AA* may need a multihop communication to the *AS*. This is even more time consuming in a distributed mesh network, where the multihop path can be much longer then two hops. Thus our solution limits the packet exchange between *AA* and *AS* to only two

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

packets.

In a standard 802.11i network handoff is done via preauthentication through the current access point, that is, the *SA* must detect the next access point to associate to and complete a 802.1X authentication through the current access point that acts as a proxy. This means that a complete exchange as depicted in Fig. 2 must be carried, but it also means that the new access point must be known before leaving the current one. This limitates the mobility of the hosts.

An alternative solution is the one described in [7], based on proactive generation and distribution of *PMK* keys. Such a solution is applicable only in environments where SAs move using fixed paths, so it doesn't really fit a mobile network, moreover, *PMK* keys are exchanged between access points, and that doesn't fit our goals, as described before.

III. Proposed solution

Our solution, that we refer to as *fast authentication* is based on the following principle: when a *SA* moves towards a new *AA*, the *AA* must ask the *PMK* to the *AS*, but it also must proof that its request is legitimate. A legitimate request is done by an *AA* when it enters in contact with a *SA*, that is a machine already in possess of a *PMK* key, an illegitimate request is a request made by an *AA* that is not in contact with somebody possessing a *PMK* key. The turning point is that when doing an handoff, *SA* must give to the *AA* a *token* that *AA* forwards to *AS* to proof that the request is legitimate. The token cannot be forged by somebody not possessing the *PMK* but since it must pass through an insecure channel it must not reveal sensitive informations and must be unrepeatable. Once received the token the *AS* verifies that it was forged by an accredited machine and forwards *PMK* key to the *AA*.

The token is generated by *SA* in the following way:

$$TOKEN = [RANDOM, AA_{id}, H(AA_{id}, RANDOM, PMK), PMKID] \quad (1)$$

where:

- *RANDOM* is a sufficiently large pseudo-random value to guarantee freshness of the token, a 160 bit string should fit.
- AA_{id} is an identifier of *AA* known to *SA* and to *AS*, for example in 802.11 network, the ESSID or BSSID value of *AA* could be chosen also as RADIUS login for *AA* to *AS*. Since ESSID and BSSID are sponsored in beacon frames it would be known to both *AS* and *SA*.
- *H()* is a secure hash function, such as SHA-2 functions.
- *PMKID* is an identifier of the *PMK*, used only for indexing purposes.

Once received the token *AA* encrypts it into a RADIUS message and forwards it to *AS* in a packet that we refer to as $\{TOKEN\}RADIUSKEY$, meaning that the content of the curly braces was crypted with a key named *RADIUSKEY*, that is only shared between *AA* and *AS*, RADIUS protocol header must contain also the identifier of AA_{id}. Once received the token *AS* completes the following checks:

1) it decrypts the packet with *RADIUSKEY*
2) it verifies that the AA_{id} contained corresponds to the RADIUS identifier used by *AA*
3) it searches in its archive of active *PMK* the one corresponding to *PMKID*
4) it verifies the hash function using the corresponding *PMK*.

If 2nd step fails *AS* is probably under a reply attack, since the packet could have been sniffed from a previous handshake and repeated by a malicious *SA* or *AA*. If the 3rd step fails the corresponding *PMK* might be expired, while if the hash is incorrect it might be somebody trying to forge a token without owning the *PMK* corresponding to *PMKID*. Thus the hash function used with *PMK* as an argument works as a signature. Upon a failure of one of the checks the *AS* might take countermeasures, such as triggering a full EAP-TLS authentication or simply discard the request, countermeasures are out of the scope of this article. If none of the above steps fails then *AS* can send the requested *PMK* to *AA*. Referring to Fig. 2 scheme, our mechanisms substitutes block 3 with a two-way exchange, reducing packets needed for authentication in case of handoff. In a strictly standard compliant implementation the token should be included into a specific EAP method into third packet of block 3, and response should be included in the RADIUS packet of block 4. That would reduce block 3 to a two-way exchange. In our implementation, we decided to include the token directly in EAP-Identity response packet; in the Identity field we used a separator to divide real identity from authentication token, this way we completely eliminate block 3 exchange. If an unaware *AS* receives an EAP-Identity frame containing a token, it would not recognize the separator and consider the whole filed as identity field, thus it would not find requested identity in its database and simply reject the request. Upon receival of a reject response, *SA* can trigger a complete EAP-TLS authentication.

Further enhancements to the proposed solution

There is a slight incoherence in 802.11i model that can be fixed with a minimal modification of our protocol but that achieves higher level of security. In 802.11i *PMK* key is derived as a result of a correct authentication, and then moved to *AA*. The act of moving *PMK* is implicitly modifying the roles of involved machines; as said *AS* is responsible for access control, so there is no reason why a key that is testifying the authorization state of *SA* would be moved away from *AS*. Using the terminology introduced in [8] it is sensed to separate a security association for *authentication material* (EAP-Key Security Association) from a *service* security association. The first one should be defined between *SA* and *AS*, and should testify authentication, the second should be defined between the machine providing services (*AA*) and *SA* and keying material should not be shared by the two. To achieve this separation we can use a second key, that is generated by EAP method and not exported through RADIUS protocol, called Application Master Session Key (*AMSK*). Since every EAP method generating

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

keying material is supposed to produce a 64 Byte *MSK*, that is moved to *AA* (first 32 Bytes of *MSK* constitute *PMK* key) but also a 64 Byte *AMSK* that stays into *AS* and *SA*, we can use the whole *AMSK* or a just a portion to define a EAP-Key security association and generate a different *PMK* to send to *AA*. This way we use *AMSK* for signing the token and generating *PMK* keys. The procedure can be described in 6 steps:

1) *SA* enters for first time the network, during authentication *SA* and *AS* generate *AMSK* and *PMK*.
2) *AS* moves *PMK* to AA_1 and everything proceeds as before
3) *SA* operates a handoff procedure, and moves to AA_2
4) *SA* sends a token, signing it with *AMSK*, token is forwarded by AA_2.
5) *AS* receives the token, verifies the signature and generates a new *PMK* that we call PMK_i, relative to this handoff and that *SA* can generate as well
6) *AS* moves PMK_i to *AA*

Note that every handoff produces a new key PMK_i, so every layer II link is secured by a different key. In the previuos model every link involving the same *SA* was using the same *PMK*.

To ensure that PMK_i is known to both *AS* and *SA*, it must be generated as a hash function (or a generic pseudo random function $PRF()$) applied to *AMSK* and the *RANDOM* value, that is owned by the two machines and contained in the token.

The token takes the following form:

$$TOKEN = [RANDOM, AA_{id}, H(AA_{id}, RANDOM, AMSK), AMSKID] \quad (2)$$

and using RADIUS protocol *AS* returns a PMK_i of the form:

$$PMK_i = \{PRF(RANDOM, AMSK)\}RADIUSKEY \quad (3)$$

This approach increases the security of the system because *PMK* keys are different for every new link generated, that means that compromission of a single PMK_i leads only to disclosure of the traffic passing on a single link. For every handoff there is rekey that makes the compromised PMK_i useless. Compromission of a machine leads to disclosure of a single *AMSK* and of a certain number of PMK_i keys, limiting the impact on the network.

While achieving a much higher security level, in a mesh network this second approach implies that every machine must keep a *PMK* key per link, and not a *PMK* per machine. For extremely low resources machines the first approach might be necessary, since low memory space and CPU time is necessary.

Note that this second approach while respecting the requested security properties introduces a new security problem: since the proof of authentication is not *PMK* anymore but *AMSK*, even if compromission of a *AMSK* cannot lead to forgery of new *AMSK* it can lead to forgery of new *PMK* keys. That is, a compromised machine could use the revealed *AMSK* to forge any number of tokens, using different AA_{id} to be later used with different authenticators to generate valid *PMK* keys. To prevent this, the token should include some information to proof freshness, for example layer II specific information used in block 1 of Fig.2; since the only informations that the two machines share are layer II specific we prefer not to specify any format and leave this feature to implementors. As an example in 802.11i networks, the token might include data from the standard 802.11 authentication and association phase (such as sequence numbers, CRC, challenge text fields) that the authenticator can check before forwarding it to *AS*.

IV. Implementation

Both the solutions have been implemented in a testbed constituted of two 802.11i access point and an authentication server with the topology depicted in Fig. 1. All the machines are standard x86 processor (varying from 600MHz to 1.6GHz clock freq.) using primsII wireless NIC and hostap driver and relative applications [9], as a radius server it was chosen FreeRADIUS. The OS's are Fedora Core III GNU/Linux, using 2.6 kernel, modifications have been made to wpa_supplicant, and to the RADIUS server. During handoff procedures a wireless and wired network sniffer (Ethereal) were used on the AS machine, and the results merged in a unique file.

Handoffs were forced sending deauthentication messages from the access points, this way *SA* was forced to make multiple handoffs between the two access points. It must be noted that forcing deauthentication shortens total handoff time of the time needed by *SA* to detect link failures; after each deauthentication a 802.11 scan is triggered to find new access points. When testing standard 802.11i handoffs Fig. 2 procedure is triggered, while our solution eliminates block 3.

Data collected are not meant to be an extremely precise measure of the authentication times, we just want to show that *fast authentication* is a significant improvement to handoff performance limiting the number of RADIUS packet produced.

Data collected in Fig. 3 are average inter arrival time values collected over 8 handoff carried with full EAP-TLS authentication. Following details must be noted:

- inter arrival time for packet 6 is significantly higher then others, this is probably due to driver or firmware issues since it goes from 1s to 0.11s without apparent reason
- our solution influences EAP packet exchange, that is the grayed out part of the table. To avoid data 6 affect comparisons we didn't include packet 6 into gray zone
- since sniffing is done in the AS, data 8 represents time needed for the packet to cross the network and reach AS, data 9 represents only elaboration time in AS, data 10 represents time to reach AA in return, data 11 represents elaboration time in AA. This four packets logical sequence is repeated for the following EAP packets
- we consider propagation time in wireless medium unrelevant compared to latency of RADIUS packets so we don't distinguish between start and arrival time of a one hop transmission
- Our testbed is an infrastructure network, so path between *AA* and *AS* is composed of a single hop, thus latency is really low. In [10] an experimental measurement of latency time in a mobile mesh networks composed of 33

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

machines shows that latency vary between 0.37 and 2.98 seconds depending on the routing protocol used. We use factor F to simulate additional lag time for a mesh network environment. This way we produce column IAT2.

		IAT (s)	%EAP	IAT2 (s)	F
1	SA -> AA IEEE 802.11 Authentication				
2	AA -> SA IEEE 802.11 Authentication	0.0513		0.0513	1
3	SA -> AA IEEE 802.11 Association Request	0.0017		0.0017	1
4	AA -> SA IEEE 802.11 Association Response	0.0920		0.0920	1
5	SA -> AA EAPOL Start	0.0057		0.0057	1
6	AA -> SA EAP Request, Identity	0.6610		0.6610	1
7	SA -> AA EAP Response, Identity	0.0022	1.26	0.0022	1
8	AA -> AS RADIUS Access Request	0.0013	0.72	0.1274	100
9	AS -> AA RADIUS Access challenge	0.0066	3.71	0.0066	1
10	AA -> SA EAP Request, EAP-TLS	0.0074	4.19	0.7448	100
11	SA -> AA TLS Client Hello	0.0247	13.87	0.0247	1
12	AA -> AS RADIUS Access Request	0.0011	0.62	0.1110	100
13	AS -> AA RADIUS Access challenge	0.0070	3.92	0.0070	1
14	AA -> SA TLS Server Hello	0.0036	2.02	0.3594	100
15	SA -> AA TLS Certificate, Client Key Exchange	0.0346	19.49	0.0346	1
16	AA -> AS RADIUS Access Request	0.0034	1.94	0.3442	100
17	AS -> AA RADIUS Access challenge	0.0344	19.33	0.0344	1
18	AA -> SA TLS Change Cipher Spec	0.0041	2.33	0.4132	100
19	SA -> AA EAP Response, EAP-TLS	0.0029	1.64	0.0029	1
20	AA -> AS RADIUS Access Request	0.0016	0.89	0.1579	100
21	AS -> AA RADIUS Access Accept	0.0300	16.89	0.0300	1
22	AA -> SA EAP Success	0.0127	7.17	1.2738	100
23	AA -> SA EAPOL Key	0.0012		0.0012	1
24	SA -> AA EAPOL Key	0.0164		0.0164	1
25	AA -> SA EAPOL Key	0.0026		0.0026	1
26	SA -> AA EAPOL Key	0.0366		0.0366	1
	Total time (s)	1.0463		4.5425	
	EAP exchange time (s)	0.1777		3.6739	
	100*(EAP time)/(total time)	16.9840		80.8793	

Fig. 3. IAT: inter-arrival time, %EAP: weight of specified packet over whole EAP procedure, F: amplification factor, IAT2: amplified values of IAT

		IAT (s)	%EAP	IAT2 (s)	F
1	SA -> AA IEEE 802.11 Authentication				
2	AA -> SA IEEE 802.11 Authentication	0.0392		0.0392	1
3	SA -> AA IEEE 802.11 Association	0.0017		0.0017	1
4	AA -> SA IEEE 802.11 Association	0.0807		0.0807	1
5	SA -> AA EAPOL Start	0.0020		0.0020	1
6	AA -> SA EAP Request, Identity	0.4417		0.4417	1
7	SA -> AA EAP Response, Identity	0.0064	35.97	0.0064	1
8	AA -> AS RADIUS Access Request	0.0017	9.64	0.1727	100
9	AS -> AA RADIUS Access Accept	0.0063	35.42	0.0063	1
10	AA -> SA EAP Response	0.0034	18.97	0.3400	100
11	AA -> SA EAPOL Key	0.0009		0.0009	1
12	SA -> AA EAPOL Key	0.0250		0.0250	1
13	AA -> SA EAPOL Key	0.0089		0.0089	1
14	SA -> AA EAPOL Key	0.0273		0.0273	1
	Total time (s)	0.6454		1.1530	
	EAP exchange time (s)	0.0179		0.5255	
	100*(EAP time)/(total time)	2.7770		45.5813	

Fig. 4. Comparison between *fast authentication* and EAP-TLS results

In Fig. 4 we collected the same average values for *fast authentication* and in Fig. 5 there is a brief comparison between results. As expected *fast authentication* needs has much better relative performances than EAP-TLS, and absolute gain significantly grows with factor F. It must also be noted that latency measurements made in [10] are done on paths that have an average length between 1.18 and 2.47 hops and without encryption. In indoor environment even networks composed of less machines could produce longer path and encryption algorithms will increase latency time. Thus in a real mesh environment EAP exchange time has a greater impact over total handoff time, so *fast authentication* ie expected to improve absolute performance even more.

	EAP TIME	EAP TIME (2)
EAP-TLS (s)	0.1777	3.6739
Fast Auth. (s)	0.0179	0.5255
Gain (s)	0.1598	3.1484
Gain (%)	89.91	85.7

Fig. 5. EAP TIME: total time needed for EAP exchange in handoff procedure, EAP TIME (2): EAP time amplified wiht factor F

V. Conclusion

In this paper we analyzed 802.1X authentication and its application in handoff techniques. We outlined security features of possible implementation of handoff methods and selected a set of security features that we consider adequate for application in service networks working in infrastructure and in mesh mode.

We projected and realized a novel method for authentication during handoff that respects the security model we proposed while reducing the number of packets needed. *Fast authentication* reduced number of packets exchanged with EAP protocol, thus reducing total handoff time. Our testbed was formed by a wireless infrastructure network with multiple access points but the impact of our solution is expected to be even higher in a mesh environment, where RADIUS packets travel over multi-hop paths.

As future work we plan to enhance hostap set of application to implement a 802.1X mesh testbed, since at the time of writing there is no working solution of mesh network using 802.1X and import and test the proposed handoff scheme in a real mesh network.

References

[1] Institute of Electrical and Electronic Engineers, Inc., *IEEE Standard for Local and metropolitan area networks Port-Based Network Access Control*, IEEE Std., 2001.

[2] ——, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 6: Medium Access Control (MAC) Security Enhancements*, IEEE Std., 2004.

[3] T. Dierks, C. Allen, W. Treese, P. Karlton, A. Freier, and P. Kocher, "The tls protocol version 1.0," RFC 2246, 1999.

[4] B. Aboba and D. Simon, "Ppp eap tls authentication protocol," RFC 2716, 1999.

[5] A. Mishra, M. Shin, and W. Arbaugh, "An empirical analysis of the IEEE 802.11 MAC layer handoff process." [Online]. Available: citeseer.ist.psu.edu/567990.html

[6] Y. Hao, L. Haiyun, Y. Fan, L. Songwu, and Z. Lixia, "Security in mobile ad hoc networks: Challenges and solutions," *Wireless Communications, IEEE*, vol. 11, pp. 38 – 47, 2004.

[7] A. Mishra, M. H. Shin, j. Nick L. Petroni, T. C. Clancy, and W. A. Arbaugh, "Proactive key distribution using neighbor graphs," *IEEE Wireless Communications*, 2004.

[8] I. E. W. Group, "Extensible authentication protocol (EAP) key management framework," draft RFC version 7, 2005.

[9] [Online]. Available: http://hostap.epitest.fi/

[10] R. S. Gray, D. Kotz, C. Newport, N. Dubrovsky, A. Fiske, J. Liu, C. Masone, S. McGrath, and Y. Yuan, "Outdoor experimental comparison of four ad hoc routing algorithms," in *Proceedings of the ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, October 2004.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.