

Fast, Secure Handovers in 802.11: Back to the Basis

Rodolphe Marques
IEETA / Univ. of Aveiro
Campus Univ. de Santiago
3810-193 Aveiro, Portugal
rodolphe.marques@ua.pt

André Zúquete
IT / IEETA / Univ. of Aveiro
Campus Univ. de Santiago
3810-193 Aveiro, Portugal
andre.zuquete@ua.pt

ABSTRACT

This article presents a fast, secure handover protocol for 802.11 networks. The protocol keeps the security functionalities of 802.1X but uses a new reauthentication protocol that promotes fast handovers during reassociations. The reauthentication protocol recovers the original 802.11 paradigm: authenticate first, reassociate next. Following this paradigm, we conceived two new 802.11 authentication and reassociation protocols, which allow a mobile station to perform 802.1X reauthentications before reassociations with the same functionality of a complete 802.1X authentication. Furthermore, reassociation protocols are authenticated, preventing denial-of-service scenarios that are not handled by 802.11i. Our new approach requires little from the environment, namely a new, central Reauthentication Service, for storing data used in the reauthentication of stations. The time of security-related tasks that contribute to handover delays was dramatically reduced to 1.5 ms, while an 802.1X fast resume takes more than 150 ms. Finally, our protocol addresses most design goals and problems stated by standards' working groups for fast, secure roaming in 802.11.

Categories and Subject Descriptors

C.2.1 [Computer Communication Networks]: Network Architecture and Design—*Wireless communication*; C.2.0 [Computer Communication Networks]: General—*Security and protection*

General Terms

Measurement, Performance, Security

Keywords

802.11 roaming, 802.1X authentication, fast reauthentication, fast handover

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Q2SWiner'08, October 27–28, 2008, Vancouver, BC, Canada.
Copyright 2008 ACM 978-1-60558-237-5/08/10 ...\$5.00.

1. INTRODUCTION

The widespread deployment of 802.11 Access Points (APs) is a strong incentive to the mobility of people while keeping their network connections. However, while mobility is possible, the support for fast handovers of mobile stations (STAs) is still incipient. Namely, handovers in wireless networks requiring 802.1X authentications take a non-negligible time, which create noticeable outage periods.

Currently, 802.1X is the accepted standard for implementing strong authentication and access control in large wireless networks. Furthermore, this standard enables the mutual authentication of an STA and an access network providing by one organization, even when the user managing the STA is known and authenticated by another organization. Unfortunately, this standard, by design, is not suitable for implementing fast handovers of mobile STAs. This happens because, with 802.1X, the reassociation of an STA to a different AP takes place before the mutual authentication between the STA and the AP (or network) where he got reassociated. This means that the time expended in the mutual authentication of STAs and APs will always contribute to the outage time during a handover (see Figure 1).

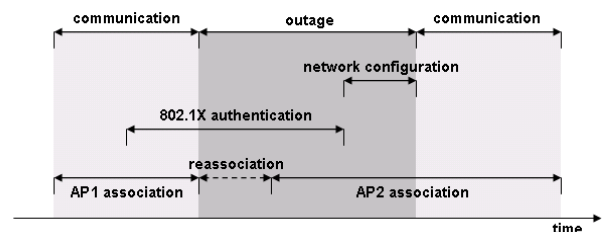


Figure 1: Outage periods after reassociations with 802.1X authentications.

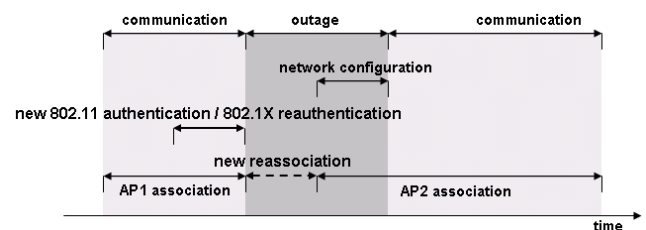


Figure 2: Outage periods after reassociations with our new 802.11 authentication and reassociation protocols.

In this article, we provide a new approach for dealing with intra-domain, mutual reauthentications of mobile STAs and 802.11 wireless networks. Our approach recovers the 802.11 principles for reassociation of mobile STAs, which are the following: first, the STA is authenticated by a candidate serving AP, while associated with a currently serving AP; then it gets reassociated to the new serving AP. With this model, outage periods during handovers are reduced to the time to perform reassociations and network configurations, but not authentications (see Figure 2).

To design the new approach, we isolated the functionalities provided by 802.1X and we implemented them in two new 802.11 authentication and reassociation protocols. The new authentication protocol performs a fast 802.1X reauthentication, though some 802.1X functionalities are delegated to the new reassociation (e.g. multicast/broadcast group key distribution). This way, we completely avoid 802.1X-related negotiations after a reassociation protocol while keeping the security features of 802.1X.

For implementing the 802.1X reauthentication we also adopted the general reauthentication architecture and key hierarchy proposed by the HOKEY (HandOver KEYing) IETF Working Group [16, 21], which uses a local HOKEY service and key hierarchies starting in the 802.1X EMSK (Extended Master Session Key). Our proposal is also inline with a TG*i* (IEEE Task Group *i*) goal: handovers to new serving APs should not require full 802.1X authentications.

This article is organized as follows. Section 2 overviews the 802.1X authentication in 802.11 networks. Section 3 presents the related work. Section 4 presents our contribution and Section 5 evaluates its security. Section 6 presents some practical considerations about the new reauthentication and reassociation protocols. Section 7 describes a prototype implementation of the two new protocols and Section 8 evaluates their performance, namely the reduction of outage delays. Finally, Section 9 presents the conclusions.

2. 802.1X IN 802.11 NETWORKS

The 802.1X architecture is formed by three types of entities: Supplicant, Authenticator and Authentication Server (AS). In a 802.11 network, the Supplicant is the STA and the Authenticator is the AP. Local AS's may use the services provided by remote AS's to authenticate Supplicants belonging to roaming users (using, for instance, a hierarchy of RADIUS servers [25]).

In 802.11 networks, the 802.1X has three phases (see Figure 3). In the first phase, the Supplicant connects to the Authenticator, which requires two 802.11 protocols: authentication and (re)association. As no secrets are shared between them at this stage, OSA (Open System Architecture) authentication is used and no effective authentication occurs.

In the second phase, Supplicant and AS mutually authenticate each other using an EAP-based protocol relayed by the Authenticator. At the end, they share two secret keys, EMSK (Extended Master Session Key) and MSK (Master Session Key), and the AS securely uploads MSK to the Authenticator. Both Supplicant and Authenticator use then MSK to derive PMK (Pairwise Master Key).

In the third phase, Supplicant and Authenticator mutually authenticate each other, using PMK and a 4-way handshake protocol (4WHP, see Figure 4), and negotiate a session key, PTK (Pairwise Temporary Key) from PMK and two nonces, N1 and N2. Along this protocol, both the STA and

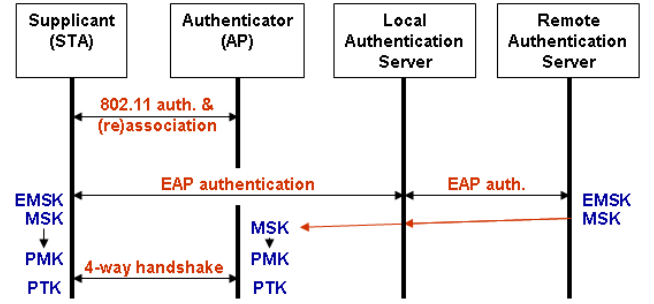


Figure 3: 802.1X phases, negotiated keys (EMSK, MSK and PTK) and derived keys (PMK).

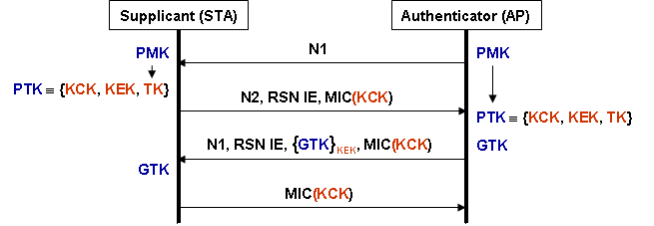


Figure 4: 802.1X third phase, 4-way handshake protocol, where Supplicant and Authenticator negotiate a fresh PTK from PMK and nonces N1 and N2.

the AP authenticate the copies of the RSN IE (Robust Security Network Information Element) values received in the 802.11 (re)association performed in phase 1. Finally, the AP also sends to the STA a Group Temporary Key (GTK) for protecting multicast/broadcast transmissions.

The PTK subdivides itself in three other keys: KCK (Key Confirmation Key), KEK (Key Encryption Key) and TK (Temporary Key). KCK is used to authenticate the last three messages of the 4WHP by means of a MIC (Message Integrity Code) computed over the messages' contents. Secret portions of these messages, namely the GTK, are encrypted with the KEK. Finally, the TK is used by Supplicant and Authenticator to encrypt and authenticate data frames.

3. RELATED WORK

The scope of our work is to achieve fast intra-domain handover by avoiding the full 802.1X authentication each time a handover occurs. There are two main approaches for tackling this problem: (i) proactive security context transfers between APs and (ii) fast recreation of new security contexts in new serving APs; our contribution follows this last one. Some hybrid approaches exist as well.

Using security context transfer between APs for fast handover is appealing because it reduces the reassociation workload required to STAs [20, 11, 22] or makes it completely disappear [26]. Some proposed architectures manage APs as physical terminals of a "central AP" managed by the network, where an STA's security context can be migrated to the AP, or APs, closest to the STA [26, 22].

However, context transfers also have disadvantages. The first disadvantage is that the network of APs must have some management infrastructure for handling inter-AP secure migration of security contexts: [20] uses a Neighbour Graph of APs and requires the establishment of security associations

between arbitrary APs; [11] uses the AS and AP neighbourhood information; [26] uses a centralized Access Controller, which takes full control of handover decisions; [22] uses a centralized CAPWAP architecture [8], which involves extra network entities and requires extra facilities for managing switches' tables. These management infrastructures raise several security issues, which are described in [3].

The second disadvantage is that, according to 802.11i [18], key hierarchies used in each AP for a given STA (starting in PMK) must be different, which means that STAs and APs must nevertheless run a 4WHP after a reassociation [20, 11, 22]. In [11], though, was proposed an alternative approach for postponing the 4WHP, by temporarily reusing a PTK distributed by the previously serving AP.

Running this protocol is also mandatory if RSN IE elements provided by different APs of the same network are different. On the other way, in systems where STAs do not notice when being served by different APs, as in [26], it is very hard, if not impossible, to deploy heterogeneous sets of APs.

The third disadvantage is that APs must proactively exchange contexts before the actual occurrence of reassociations [20, 11, 22], possibly wasting time and resources for tackling a problem that may never exist. Moreover, all this effort may not be enough; in [20, 11] the STA must run a complete 802.1X authentication whenever associating to an AP outside the Neighbour Graph of the previous AP.

Finally, the fourth disadvantage is that enabling STAs to be transparently served by different APs, as in [26], complicates the management of access networks, namely the management of link layer routing and address translation tables.

When security contexts are recreated in new serving APs, for implementing fast handovers, some optimizations must occur to reduce post-reassociation 802.1X delays.

In [1] was proposed an 802.1X pre-authentication to be performed before 802.11 reassociations. However, they do not handle fast reauthentications, they only use the basic 802.1X protocol; therefore, are less suitable for fast moving STAs, which have less time to perform authentications in candidate APs. This issue is also relevant when the moving STA loses connection with the currently serving AP before authenticating in a new one.

In [6] was proposed an architecture where an STA may communicate after being reassociated and before running the remaining 802.1X authentication phases (2 and 3). The communication is tunnelled to the previously serving AP through Dynamic Secure Tunnels. These tunnels are created on demand during reassociations, with the help of the AS, and stay afterwards for serving other STAs. However, this approach is likely to complicate APs, since they have to decrypt and validate layer 2 frames coming from the radio link and from security tunnels and, vice-versa, they have to encrypt layer 2 frames to be sent by radio or through a security tunnel. Spoofed reassociation requests may also lead APs to initiate the creation of useless tunnels.

In [13] was proposed an architecture based on a secure 3-party key distribution protocol, using a local HOKEY server [7] besides the usual 802.1X entities, Authenticator and AS. The HOKEY server reduces the duration of the second 802.1X phase, by replacing the full EAP authentication by a local ERP (EAP Reauthentication Protocol [17]) authentication between the STA and the HOKEY server. The HOKEY server uses derived EAP keying material for ERP,

uploaded by the local AS, this way eliminating roundtrips to a remote AS. However, we still have the same phases of 802.1X: phase 2, now with ERP, and phase 3 (4WHP).

In [5] was proposed an architecture where a currently serving AP provides the STA a credential that enables its fast reauthentication in future serving APs. It requires all APs to share a common secret key, K_{net} , which is used to authenticate credentials and to secretly compute keys shared between STAs and new serving APs. This approach has several security drawbacks that derive from the sharing of K_{net} among all APs: compromise of an AP leaks K_{net} , allowing the attacker to freely generate credentials and to run any number of rogue APs. Furthermore, an attacker knowing K_{net} can obtain secret keys from eavesdropped reauthentication frames, thus compromising the security of past and present communications. For all this, the K_{net} needs to be updated frequently in all APs and strong authentication protocols are must be executed after the fast reauthentication, which should only allow limited connection periods.

In [10] was proposed an hybrid approach, where there is some context migration between APs and some context recreation between the STA and the new serving AP. APs are dynamically clustered using Neighbour Graphs [15], and on each cluster there is a Cluster Roaming Key (CRK) per visiting STA. This CRK is computed from the initial STA PMK and from the (current) list of cluster members. This CRK is used by the STA and serving AP to derive their own, local PMK without message exchanges. Using a per-AP PMK, an STA performs an equivalent 4WHP using only the two 802.11 reassociation request/response frames (which are authenticated). However, using only two messages requires a "self nonce generation": the STA must guess the nonce the AP will use for deriving PTK from PMK. Guessing failures require two more synchronization frames.

This proposal is very similar to ours; however, there are some fundamental differences. Similar aspects are the usage of 802.11 protocols for 802.1X-like authentication and the authentication of reassociations. The management of PMKs, however, is completely different: they have to manage clusters of APs and to provide clustering information to STAs, which require it for computing CRKs, while we do not care about AP clustering. Furthermore, they do not provide fast reassociation when the STA moves between APs belonging to different clusters. Finally, a compromised AP in a cluster is able to derive the STA's PMK used by all other APs in the same cluster (this problem also exists in [11], but with lower risk), while in our proposal the PMK used in each AP cannot be used to compute the PMK used in all other APs.

The 802.11r standard initiative [9] aims at reducing the security overhead during AP transitions. The two most obvious security-related contributions are the clarification of opportunistic key caching in APs and the elimination of the 4WHP that in the current 802.1X follows reassociations.

The 802.11r defines a new EMSK-based key hierarchy and the concept of Mobility Domains. A group of APs jointly forms a single Mobility Domain and in doing so gain access to a common, EMSK-based key hierarchy for an STA. Optimistically, an STA assumes that such key hierarchy exists in the target AP when roaming occurs (opportunistic key caching). If it does not exist, then an unexpected latency may occur while the target AP collects this information from its holder. However, the 802.11r does not specifies how or

when this key hierarchy is uploaded in advance to the APs of the same Mobility Domain and how APs fetch key hierarchies on demand. All these key management actions are implementation dependent.

Our proposal is inline with 802.11r concerning the elimination of the 4WHP after a reassociation. But our proposal goes further ahead, specifying how target APs obtain the keying material, thus making our proposal implementation independent. The key material is distributed without requiring architectural changes in the network elements to support inter-AP protocols, either vendor-specific or other standard like CAPWAP [22] from IETF. Furthermore, the APs of the same Mobility Domain obtain new key material, namely PMKs, which are unique for each AP-STA pair and not shared by all APs of the same Mobility Domain, as in 802.11r. This way, the compromise of an AP of a Mobility Domain does not compromise the security of past and future communications of STAs with other APs of the same Mobility Domain. Finally, we provide authentication for re-association protocols, while 802.11r does not.

4. OUR CONTRIBUTION

Our goal is to perform fast 802.1X reauthentications to minimize reassociation handovers, while not using the 802.1X approach for implementing them. Namely, in reauthentications we want to achieve exactly the same results of 802.1X but with the shortest possible set of frame exchanges after initiating a reassociation with another AP.

Our proposal for implementing fast and secure handovers is to use two existing 802.11 protocols, authentication and reassociation, with extensions capable of providing all the functionalities of an 802.1X reauthentication. By doing so, we are able to completely skip phases 2 and 3 of 802.1X, using only its first phase for reauthentication, key distribution and reassociation (c.f. Figure 3).

We assume that 802.1X reauthentications can only occur after an ordinary 802.1X authentication. This previous authentication is responsible for producing and distributing secret material that will be used to carry on one or more fast reauthentications. This approach is inline with the fast reauthentication goals presented in [7].

According to Section 2, we have the following requirements for a fast 802.1X reauthentication:

1. Install a fresh, secret PMK in both STA and AP;
2. Use PMK and two nonces to produce a new PTK;
3. Confirm a common knowledge of PTK;
4. Exchange authenticated RSN IE capabilities;
5. Send a confidential GTK from the AP to the STA.

We handle the two first requirements in a new 802.11 authentication protocol and the two last requirements in a new 802.11 reassociation protocol (see Figure 6). The third requirement will be handled along both protocols.

Just to clarify the reader, we will use a modified 802.11 authentication protocol to implement the new, fast 802.1X reauthentication protocol. This is a completely new approach, since currently 802.11 and 802.1X authentications are totally independent from each other. Therefore, in the text we may use either the expressions “*802.1X reauthentication*” or “*new 802.11 authentication*” to refer to this protocol, depending on the context.

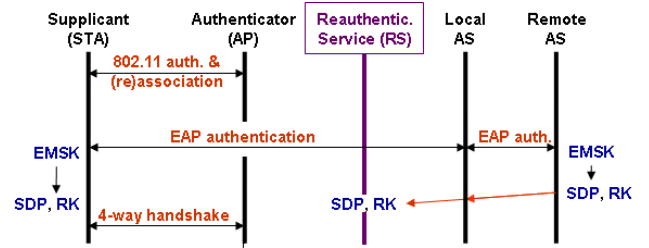


Figure 5: Reauthentication Service: integration with the 802.1X architecture and secret material uploaded to it (SDP and RK) by the local AS upon a successful EAP-based authentication within 802.1X.

4.1 Reauthentication Service

The Reauthentication Service (RS) is a new service that we use for handling fast 802.1X reauthentications. Following the terminology of 802.11r, the RS is the enabler of a Mobility Domain, which is formed by all APs that know how to reach and securely interact with the RS for handling STA’s reauthentication requests.

The RS replaces the AS for reauthentications. We consider that in each domain managed by a local AS there is an RS, which receives secret reauthentication material from the former. We assume that the RS is able to authenticate AS messages and these can only be understood by the RS.

The RS can be implemented in two different ways: (i) as part of the local AS; or (ii) as an independent, HOKEY server [16]. Note that if RS is part of the local AS, and not a separate service/host, the secure communication between them as no extra management overhead.

The APs use the RS instead of the AS for handling STA’s 802.1X reauthentication requests. We assume that there are security associations between all APs and the RS, similar to the ones that exist between APs and the local AS. These security associations are fundamental to (i) authenticate RS messages to APs and to (ii) enforce the confidentiality of keys provided by the RS to the APs. Using the terminology of 802.11r, the Mobility Domain of an STA is the set of APs that have a security association with the local RS. Again, if the RS is part of the local AS, and not a separate service/host, the 802.1X architecture guaranties these assumptions, because it uses a security association between each AP and the local AS.

4.2 Initial 802.1X authentication

As previously stated, for the new reauthentication protocol we assume that some secret material was produced by a previous, ordinary 802.1X authentication. Such material is a key and a related unique identifier: **Reauthentication Key (RK)** and **STA Digital Pseudonym (SDP)**.

These two values are computed by both Supplicant and AS during an ordinary 802.1X authentication and uploaded by the latter to the RS (see Figure 5). The RK will be used to generate a fresh PMK for each reauthentication request tagged with SDP.

After an 802.1X authentication, a Supplicant shares a secret EMSK with the AS that authenticated it. We will use this key to derive the values of RK and SDP as follows:

$$\begin{aligned} \text{RK} &= \text{PRF-X}(\text{EMSK}, \text{“802.11 authentication”}) \\ \text{SDP} &= \text{PRF-X}(\text{RK}, \text{ID}) \end{aligned}$$

where the $\text{PRF-X}(Y)$ represents the first X bits computed over Y by a pseudo-random function for key expansion defined in [12] and where ID is the identification of the Supplicant provided to the AS during its authentication. According with [21], RK is a Domain Specific Root Key and SDP is a Domain Specific Usage Specific Root Key.

We use the SDP for uniquely identifying an authenticated session of an STA instead of its MAC address. The reason for doing so is that an SDP cannot be spoofed by an attacker, as it derives from $EMSK$, while a MAC address can be spoofed. This way, an attacker cannot use spoofing attacks to install in the RS a new RK for a victim STA.

Given the security assumptions of Section 4.1, the upload of SDP and RK from the AS to the RS is as protected, in terms of secrecy and integrity control, as the upload of MSK from the same AS to an AP (Authenticator).

4.3 New 802.1X reauthentication protocol

The new 802.1X reauthentication protocol uses the basic structure of 802.11 authentication protocols, while carrying extra data in frames' payload. We use only two 802.11 frames, one Authentication Request and one Authentication Response, for performing the reauthentication on the wireless medium, though the complete protocol involves two extra messages between the AP and the RS (see Figure 6):

STA→AP	Auth. Req., $SDP, \{K\}_{RK}, N1, MIC(K)$
AP→RS	$SDP, \{K\}_{RK}, N1, MIC(K)$
AP←RS	$SDP, N3, PMK$
STA←AP	Auth. Resp., $N2, N3, \Delta T, MIC(KCK)$

where $PMK = \text{hash}(K, N3)$ and KCK is computed from $PMK, N1$ and $N2$ as in 802.1X (cf. Section 2).

First, the STA generates a nonce $N1$ and a random key K . Then it sends an Authentication Request to the AP, containing its SDP, K encrypted with $RK, N1$ and a MIC computed with K . The AP forwards all these values to the RS, which uses SDP to find the STA's RK . Once knowing RK , it extracts K , validates the MIC and, if valid, generates a nonce $N3$, hashes it with K for producing PMK and sends the PMK to the AP, together with $N3$. The final Authentication Response contains two nonces, $N2$ and $N3$, the first generated by the AP and the second by the RS, and a MIC computed with KCK . The STA uses $N3$ to compute PMK , and this key, together with $N1$ and $N2$, to compute the shared PTK (and its components KCK, KEK and TK) as in 802.1X. The KCK component of PTK is then used to authenticate the received message.

At the end of this protocol, STA and AP share a fresh PTK . Its freshness is ensured by random values and nonces provided by all three protocol players: K and $N1$ (from the STA), $N2$ (from the AP) and $N3$ (from the RS).

As 802.11 Authentication Requests can be replayed, attackers could install new, fresh PMK and PTK keys in APs on behalf of spoofed STAs, creating two different DoS problems: (i) APs could get flooded by old, useless security associations for STAs that may no longer be around and (ii) attackers could interfere with security associations that are currently being used by STAs, by installing new keys. Fortunately, both problems can be solved very easily by using a monotonic, sequence counter for producing the nonce $N1$. With this strategy, the RS can detect and refuse replays of reauthentication requests for a given SDP ; all it has to do

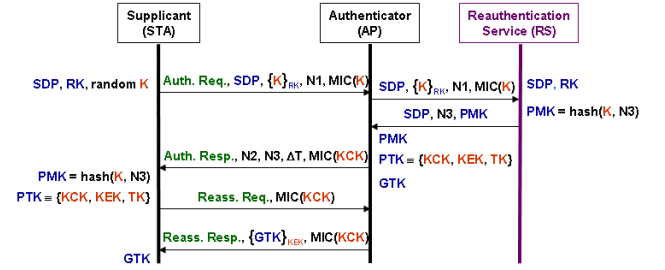


Figure 6: New 802.1X reauthentication and 802.11 reassociation protocols and interoperation with RS.

is to keep and check the previous, valid value of $N1$ used in the last successful reauthentication request of that SDP .

At the end of this protocol, the STA believes that the AP knows the new keys, PMK and PTK , because the Authentication Response is authenticated with KCK , a part of PTK . Therefore, STA can be sure the AP is genuine (belongs to the target network), otherwise it could not have received the key PMK from the RS. On the contrary, the AP is not yet convinced that the STA knows PMK and PTK . This proof will be provided latter, during an association or re-association. Nevertheless, the AP may consider the STA authenticated, because of the positive reply from the RS.

The ΔT in the Authentication Response indicates the period the AP will keep the security context (authenticated state and secret keys PMK and PTK) negotiated with the STA. If the STA does not get associated to the AP before ΔT , the AP removes the security context without any warning. This simple mechanism prevents APs from getting flooded with security contexts, while giving STAs some feedback about the lifetime of their security contexts' caching in APs. Furthermore, APs are free to manage independently their security contexts, namely they can use different caching periods.

A mobile STA should use this reauthentication protocol with all neighbour APs as soon as they become suitable targets for roaming. This can take place during AP scanning/probing phases, when the STA looks for APs more interesting than the one actually serving it [19, 14, 24]. Note, however, that while scanning/probing phases may occur frequently to better decide when to roam to another AP, authentication with each neighbour AP needs to be performed once for some time, which depends on the AP security contexts' caching period (conveyed to the STA by the ΔT field in the 802.11 Authentication Response).

4.4 New 802.11 reassociation protocol

The new 802.11 reassociation protocol has two differences regarding the current one: (i) authenticates exchanged RSN IE values and (ii) sends a group key GTK from the AP to the STA. This is achieved with some extra fields in the 802.11 Reassociation Request and Response frames (see Figure 6):

STA→AP	Reass. Req., $MIC(KCK)$
STA←AP	Reass. Resp., $\{GTK\}_{KEK}, MIC(KCK)$

The MIC values in both request and response authenticate RSN IE values exchanged in the payload. Furthermore, the MIC in the request proves to the AP that the STA effectively knows PMK and PTK .

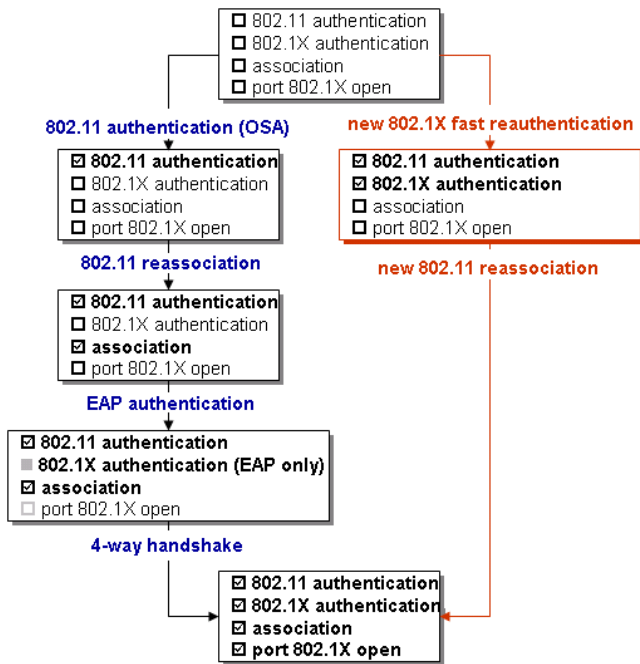


Figure 7: AP state diagram concerning 802.11 and 802.1X authentication, 802.11 association and 802.1X port state. The actual state diagram is on the left; the state diagram of our 802.1X fast reauthentication and 802.11 reassociation is on the right.

These modifications could be applied as well to the 802.11 association protocol. The only difference between them is that a Reassociation Request frame contains an extra field with the MAC address of the AP formerly serving the STA.

4.5 New AP 802.11 / 802.1X state diagram

The new 802.1X reauthentication and 802.11 reassociation protocols need to be handled in the context of a new state machine within an AP. Figure 7 shows two state diagrams, one for the current 802.11/802.1X authentication and association (left) and another for our new protocols (right).

In the state diagram on the right, we can see that after the new 802.1X fast reauthentication, both AP and STA are already authenticated in terms of 802.11 and 802.1X. Then, after the new 802.11 reassociation, the STA becomes associated to the AP and the 802.1X port in the AP can immediately be open, because 802.1X authentication is already completed. This fact enables both AP and STA to start exchanging data frames without further delays¹. Therefore, after the reassociation, no security-related delays contribute to handover outage delays.

This new state diagram differs slightly from the one proposed by B. Aboba in [2] (see Figure 8), where the PMK is installed when the STA becomes authenticated and the PTK is installed when the STA becomes associated. In fact, there is no problem in anticipating the installation of PTK and there are even some benefices, such as allowing the authen-

¹Nevertheless, the AP may not immediately start sending data frames because of delays in the update of switching tables.

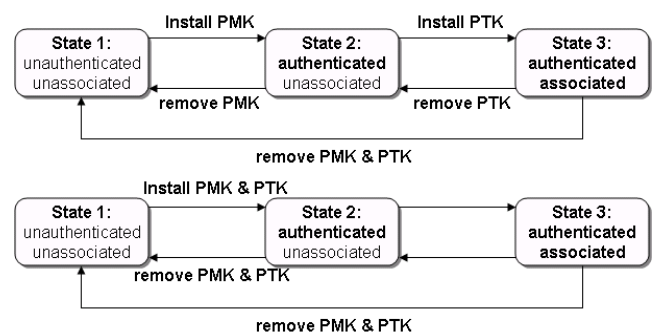


Figure 8: State diagram proposed by B. Aboba (top) and our state diagram (bottom).

tication of management frames, namely the authentication of reassociation protocol runs with KCK.

5. SECURITY EVALUATION

A security enhancement introduced by our handover protocol is that reassociation requests are authenticated. This authentication prevents attackers from creating Denial-of-Service (DoS) scenarios with spoofed reassociation requests.

The 802.11i assumptions for RSN environments state that the STA and the AP generate a different, fresh PTK for each session between the pair [18, §8.1.4 g)]. Since the exact meaning of the term *session* is not clarified in the standard, we assumed that a session may span several associations with the same AP². Under this assumption, we provide a different, fresh PMK for each session and a fresh, derived PTK. The freshness of PMK is ensured by the STA and by the RS, with K and N3, respectively; the freshness of PTK is ensured by the freshness of PMK and the two nonces, N1 and N2, provided by the STA and AP, respectively.

A TGi fast roaming goal, also stated in [3], is that compromise of one AP must not compromise past or future key material. Our proposal goes towards meeting this goal, as the PMK in each AP cannot be used to compute the PMKs used in other APs (assuming that hashing functions used to compute PMK are one-way). For fully meeting this goal, we also have to ensure that secret associations between APs and RS are independent, in the sense that compromise of one AP does not help to compromise security associations established by other APs with the RS.

Wireless eavesdroppers should not be able to cryptanalyse keys K or RK used by an STA. The cryptanalysis of K enables an attacker to eavesdrop, tamper and hijack an otherwise secure session between the STA and an AP. The cryptanalysis of RK is even more dangerous, it enables an attacker to impersonate the STA, to fool the STA with a rogue AP and to cryptanalyse K keys generated by the STA.

If K is randomly generated by an STA, then attackers have first to cryptanalyse RK to decrypt $\{K\}_{RK}$. However, the only values that attackers observe encrypted with RK are random K values, thus only exhaustive search seems usable to find RK or K. Therefore, choosing large enough RK and (random) K values, with at least 128 bits, should be enough for preventing successful exhaustive key search attacks.

²Otherwise, they would have used the term *association* instead of *session*.

The new authentication and reassociation protocols, together, are quite similar to the 802.1X 4WH protocol. The differences are that (i) the messages are sent in the opposite direction (the initiator is the STA, while in the 4WH it is the AP), (ii) the first message is authenticated, while in the 4WH it is not and (iii) PMK is negotiated by the first two messages. Therefore, assuming that PMK is secretly distributed, our protocols, together, are more secure than the 4WH, which is considered secure. Since PMK is computed from K and above we showed that eavesdroppers are not able to cryptanalyse it, then we can conclude that our two protocols are at least as secure as the 4WH.

We used the Avispa tool [23] to analyse our new protocol. Namely, we evaluated the reachability of the following goals by legitimate players: secrecy of K and GTK and authentication of received N2, N3 and RSN IE values. We found no problems in reaching these goals and we did not find attacks against the protocol.

6. PRACTICAL CONSIDERATIONS

A long-standing problem of 802.11i pre-authentications is the distribution of GTK keys. In fact, running a 802.11i pre-authentication protocol between an STA and an AP distributes a GTK to the STA, but the key may change until the effective association of the STA to AP. We solve this problem in a very simple way, as GTK keys are distributed only when reassociations happen.

Unlike solutions that migrate security contexts, we negotiate a security context with each candidate AP. Therefore, we tolerate differences in security capabilities of APs, as long as they implement our two new protocols. Natural differences between APs are the set of supported cipher suites. For example, an STA may wish to use AES CCMP whenever possible and TKIP otherwise, and not all APs may support AES CCMP. In this case, our solution gracefully adapts to the resources provided by each STA.

Our protocols follow the recommendations of [7] concerning backward compatibility and impact to existing deployments (changes are required but are reduced and do not sacrifice performance). STAs can detect if a network or AP does not support fast 802.1X reauthentication by using Information Elements in 802.11 Beacon or Probe Response frames.

7. IMPLEMENTATION

We implemented a prototype of our protocols using Linux (Fedora Core 8) and the MadWifi driver, tools and daemons. This driver was updated to implement the protocols in both STAs and APs.

The main goal of this prototype was to evaluate the minimum handover latency using our new protocols. As previously explained, when a reauthentication is realized by an STA in advance, the latency depends solely on the time to perform a reassociation. Consequently, we did not implement the RS as a separate service, but as part of the AP driver, because we were not interested in measuring reauthentication delays. An external application, namely the `iwpriv` tool, was used to provide SDP-RK pairs for the driver. The driver running on a STA keeps only one pair; the driver running on a AP keeps all provided key pairs, overwriting existing ones.

The driver was extended to allow STA applications, namely the `iwpriv` tool, to perform reauthentications in arbitrary APs using a previously downloaded SDP-RK pair. After a successful reauthentication, the drivers of both STA and AP keep internally an association between the MAC address of the peer and a PTK. For the reauthentication protocol we chose a new protocol number, 2 (0 is used for OSA, 1 for Shared Key Authentication).

The driver was extended as well to allow STA applications to perform reassociations. Upon a reassociation request for an AP, given its MAC address, the driver looks for a local MAC-PTK association and, if present, starts the new reassociation protocol using PTK. After a successful reassociation, the drivers of both STA and AP update association information, namely the keys for protecting data exchanges, derived from the TK portion of PTK.

In this prototype implementation, we used AES to encrypt K and GTK, HMAC SHA-1 to compute the MICs with K or KCK and SHA-256 for computing PMK from K and N3. For the AES we used 128 bits for both keys and data blocks.

8. PERFORMANCE EVALUATION

For evaluating the minimum delay of our secure handover, we measured the delay of forced reassociations between an STA and two APs using 802.11g. For both STA and APs we used equal Linux operating systems (Fedora Core 8) and wireless network interfaces (Netgear WG511T, with the Atheros chipset). For the AP hosts we used two equal 3.2 GHz Pentium 4 desktop hosts; for the STA we used a 1.7 GHz Pentium M laptop.

From network captures with WireShark we observed that, in average, the new 802.11 reassociation protocol takes 1.5 ms. To accurately measure this delay, the STA driver sends a null data frame immediately after the reassociation; the reassociation delay is thus the interval between the observation of the 802.11 Reassociation Request and the null data frame.

The delay of 802.1X authentications was evaluated in [4] and the minimum value (150 ms) was obtained with 802.1X fast resume; the minimum values obtained for a full 802.1X and for the 4WHP were 750 ms and 10 ms, respectively. Thus, our minimum reassociation delay is at least two orders of magnitude smaller than the lowest delay with 802.1X and at least one order of magnitude smaller than the 4WHP delay. This happens because we use less frames, only 2, while 802.1X uses at least 8 and 4WHP uses 4. Furthermore, AP does not need to contact other network entities to perform an authenticated reassociation, while in 802.1X the AP has to interact with the local AS.

Comparing our performance results with the ones presented by similar contributions (the ones that totally or partially recreate security contexts in new serving APs), we clearly outperform them. In [6] an STA may have to wait for the establishment of a secure tunnel between the current AP and the previously serving AP, with the help of an AS, after a reassociation. The delay of such procedure was not evaluated but surely it is much higher than 1.5 ms, as it uses 6 messages. In [13] the authentication and key distribution with ERP and an HOKEY server after a reassociation takes more than 184 ms. In [10], which partially uses secure context migration, the handover delay is higher than 8 ms (value obtained with a simulator).

9. CONCLUSIONS

We presented a new approach for achieving fast, 802.1X-like authentications during reassociations. Our approach recovers the original 802.11 model, using authentications prior to reassociations. This way, we are able to remove reauthentication delays from handover outage delays, since they can be executed before reassociations (see Figures 1 and 2), and to authenticate reassociation protocols, which is useful against DoS attacks. Performance valuations on a prototype showed that handover delays can be reduced down to 1.5 ms.

The proposed architecture uses a key hierarchy starting in EMSK and a Reauthentication Service for dealing with fast 802.1X reauthentications of STAs. This new service can be implemented by an AS or an HOKEY server. Finally, we do not require special configuration facilities from network equipments, no topological information regarding the actual deployment of APs and no homogeneity among facilities provided by APs (e.g. homogeneous cipher suits).

10. REFERENCES

- [1] B. Aboba. IEEE 802.1X Pre-Authentication. IEEE 802.11 TGd draft 802.11-02/389r0, June 2002.
- [2] B. Aboba. IEEE 802.11i: A Retrospective, 2004. www.ieee802.org/1/files/public/docs2004/11i-Retrospective.ppt.
- [3] B. Aboba, D. Simon, and P. Eronen. Extensible Authentication Protocol (EAP) Key Management Framework, Nov. 2007. draft-ietf-eap-keying-22.
- [4] A. Alimian and B. Aboba. Analysis of Roaming Techniques. IEEE 802.11 WG document 802.11-04/0377r1, 2004.
- [5] T. Aura and M. Roe. Reducing reauthentication delay in wireless networks. In *Proc. of the 1st Int. Conf. on Security and Privacy for Emerging Areas in Communication Networks (SECURECOMM '05)*, pages 139–148, Washington, DC, USA, 2005. IEEE Computer Society.
- [6] J. Chen, Y. Tseng, and H. Lee. A Seamless Handoff Mechanism for DHCP-Based IEEE 802.11 WLANs. *IEEE Comm. Letters*, 11(8):665–667, Aug. 2007.
- [7] T. Clancy, M. Nakhjiri, V. Narayanan, and L. Dondeti. Handover Key Management and Re-Authentication Problem Statement. RFC 5169, IETF, Mar. 2008.
- [8] S. Govindan, H. Cheng, Z. H. Yao, W. H. Zhou, and L. Yang. Objectives for Control and Provisioning of Wireless Access Points (CAPWAP). RFC 4564, IETF, July 2006.
- [9] R. Greenlaw and P. Goransson. *Secure Roaming in 802.11 Networks*. Elsevier, 2007. ISBN-13 978-0-7506-8211-4.
- [10] C.-M. Huang and J.-W. Li. An IEEE 802.11 Fast Reassociation and Pairwise Transient Key establishment Based on the Dynamic Cluster Method. In *Works. of Comp. Networks and Wireless Communications, Int. Comp. Symp. (ICS 2006)*, Taipei, Taiwan, 2006.
- [11] M. Kassab, A. Belghith, J. Bonnin, and S. Sassi. Fast Pre-Authentication Based on Proactive Key Distribution for 802.11 Infrastructure Networks. In *1st ACM Works. on Wireless Multimedia Networking and Performance Modelling (WMuNeP'05)*, Montreal, Canada, Oct. 2005.
- [12] C. Kaufman. Internet Key Exchange (IKEv2) Protocol. RFC 4306, IETF, Dec. 2005.
- [13] R. Marin, P. J. Fernandez, and A. F. Gomez. 3-Party Approach for Fast Handover in EAP-Based Wireless Networks. In *Proc. of OTM Confs., 2nd Int. Symp. on Information Security (IS'07)*, pages 1734–1751, Vilamoura, Portugal, Nov. 2007. Springer. LNCS 4804.
- [14] A. Mishra, M. Shin, and W. A. Arbaugh. An empirical analysis of the IEEE 802.11 MAC layer handoff process. *Computer Communication Review*, 33(2):93–102, 2003.
- [15] A. Mishra, M. H. Shin, J. N. L. Petroni, T. C. Clancy, and W. A. Arbaugh. Proactive key distribution using neighbor graphs. *IEEE Wireless Communication*, 11(1):26–36, Feb 2004.
- [16] M. Nakhjiri and Y. Ohba. Derivation, delivery and management of EAP based keys for handover and re-authentication. IETF HOKEY WG Internet-Draft, Nov. 2007. draft-ietf-hokey-key-mgm-01.
- [17] V. Narayanan and L. Dondeti. EAP Extensions for EAP Re-authentication Protocol (ERP). IETF HOKEY WG Internet-Draft, Nov. 2007. draft-ietf-hokey-erx-08.
- [18] L. S. C. of the IEEE Computer Society. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 6: Medium Access Control (MAC) Security Enhancements. IEEE Std 802.11i, July 2004.
- [19] S. Pack and Y. Choi. Fast Inter-AP Handoff using Predictive-Authentication Scheme in a Public Wireless LAN. In *IEEE Networks Conf. (Joint IEEE ICN 2002 and IEEE ICWLHN)*, Aug. 2002.
- [20] A. R. Prasad and H. Wang. Roaming key based fast handover in WLANs. In *IEEE Wireless Communications and Networking Conf. (WCNC 2005)*, volume 3, pages 1570–1576, Mar. 2005.
- [21] J. Salowey, L. Dondeti, V. Narayanan, and M. Nakhjiri. Specification for the Derivation of Root Keys from an Extended Master Session Key (EMSK). IETF HOKEY WG Internet-Draft, Nov. 2007. draft-ietf-hokey-emsk-hierarchy-02.
- [22] B. Sarikaya and X. Zheng. CAPWAP Handover Protocol. In *IEEE Int. Conf. on Communications (ICC'06)*, volume 4, pages 1933–1938, June 2006.
- [23] T. A. Team. *Automated Validation of Internet Security Protocols and Applications (AVISPA) v1.1 User Manual*, June 2006.
- [24] H. Velayos and G. Karlsson. Techniques to reduce IEEE 802.11b MAC layer handover time. Technical Report TRITA-IMIT-LCN R 03:02, Kungl. Tekniska Hogskolen, Stockholm, Sweden, Apr. 2003.
- [25] K. Wierenga and L. Florio. Eduroam: past, present and future. In *TERENA Networking Conf.*, Poznan, Poland, 2005.
- [26] L. Zan, J. Wang, and L. Bao. Personal AP Protocol for Mobility Management in IEEE 802.11 Systems. In *Proc. of the 2nd Ann. Int. Conf. on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS'05)*, pages 418–425, Washington, DC, USA, 2005. IEEE Computer Society.