Fast Re-authentication for Handovers in Wireless Communication Networks

Ralf Wienzek and Rajendra Persaud

Chair of Informatik 4, RWTH Aachen University, Aachen, Germany
{wienzek, persaud}@i4.informatik.rwth-aachen.de

Abstract. The evolution of wireless access technologies and the capabilities of today's mobile devices lead to an increasing demand of communication bandwidth. More and more packet-switched wireless access networks like Wireless Local Area Networks (WLANs) and networks based on Worldwide Interoperability for Microwave Access (WiMAX) are publicly available and operated by different providers. In order to achieve a high network coverage isolated access network providers are supposed to co-operate and to support handovers for users from access networks belonging to the same core network. Efficient authentication mechanisms are required that on the one hand exclude unauthorized users from the network and on the other hand support seamless handovers across access network boundaries. We propose a ticket-based fast re-authentication scheme that is independent from the actual authentication method and that only slightly modifies well-established standards like the Extensible Authentication Protocol (EAP) and the Remote Authentication Dial In User Service (RADIUS). As it is network technology independent, it in principle also allows fast handovers across different access network technologies.

Keywords: Wireless Networks, Authentication, Handover.

1 Introduction

Mobility is one of the main incentives for the development of wireless network technologies such as Wireless Local Area Networks (WLANs) based on IEEE 802.11 or Worldwide Interoperability for Microwave Access (WiMAX) based on IEEE 802.16. In general, a wireless network is subdivided into one or more access networks and a core network that do not need to belong to the same network operator. An access network consists of link-layer (L2) devices, whereas a core network consists of network-layer (L3) devices. The device the link layer of a mobile device attaches to is called L2 Point of Attachment (PoA) (Access Point (AP) in WLANs, Base Station (BS) in WiMAX networks), the device the network layer of a mobile device attaches to is called L3 PoA (Access Router (AR) in WLANs, Access Services Network Gateway (ASN-GW) in WiMAX networks).

In both technologies, access network mobility is in general provided by the technology itself, i.e. a handover (HO) between PoAs belonging to the same

F. Boavida et al. (Eds.): NETWORKING 2006, LNCS 3976, pp. 556–567, 2006.

access network does not require any higher-layer mobility solution. Core network mobility, i.e. a HO across access network boundaries, is in general based on the Internet Protocol (IP) or on Multi-Protocol Label Switching (MPLS) and goes along with a reconfiguration in the core network in order to redirect packets to the new PoA.

Mobility is a user-specific network service that has to be secured from attackers and therefore requires an authentication mechanism. In order to support real-time packet-switched applications such as Voice over IP (VoIP), the HO from an old PoA to a new PoA and the associated re-authentication with the new PoA has to be as fast as possible. The objective of this paper is to provide a fast re-authentication mechanism to support core network mobility.

In general, the mobile device authenticates with an Authentication, Authorization and Accounting (AAA) server in the core network. Also, authentication is reasonably performed before obtaining network-layer connectivity, i.e. before the assignment of an IP address. Therefore, an intermediate device is necessary to handle the authentication between the mobile device and the AAA server. In WLANs based on IEEE 802.11i this is the L2 PoA. In WLANs based on IEEE 802.11 only, the L2 PoA cannot act as intermediate device so that the L3 PoA has to be used.

During HO, a HO notification message has to be sent to the core network (e.g. by exploiting the Dynamic Host Configuration Protocol [1], which is an extensible protocol used for configuration purposes). We propose a predictive HO solution, i.e. the HO notification is sent to the L3 PoA the mobile device is attached to before it moves to the new access network. The advantage compared to reactive HO solutions in which HO notifications are sent to the new L3 PoA is that the disassociation from the old access network and the transfer of configuration information from the old L3 PoA to the new L3 PoA can be done in parallel.

In order to accelerate the authentication process during HO we propose a ticket mechanism providing a fast re-authentication of mobile nodes with the target access network. We define an additional RADIUS attribute and use the optional data field of EAP-Identity-Messages.

The rest of the paper is organized as follows: In Sect. 2 we summarize EAP-TLS with RADIUS as an example scenario to which the re-authentication scheme can be applied. In Sect. 3 we define the attacker model and describe our scheme in detail. In Sect. 4 required adaptations to apply the scheme to IEEE 802.11i like environments are given. Performance and security issues are discussed in Sect. 5 and the paper closes with some conclusions in Sect. 6.

2 Authentication Schemes

Mobile nodes (MN) that want to use core network services are required to authenticate themselves with the core network. When a MN enters an access network it first attaches itself to the L2 PoA. The authentication procedure with the core network is initiated either by directly contacting the L3 PoA managing that

access network or in combination with an authentication required for associating with the L2 PoA. An example for the first case is an IEEE 802.11 WLAN. The L3 PoA can be contacted e.g. by assigning a temporary IP address with restricted access rights to the MN as proposed in [2] or by establishing a Point-to-Point connection using the Point-to-Point Protocol over Ethernet (PPPoE) [3]. An example for the second scenario is an IEEE 802.11i AP that authenticates a MN with the help of an AAA server. Our scheme covers both scenarios. We define it for the first, simpler scenario in detail, and describe necessary adaptations for IEEE 802.11i environments in Sect. 4.

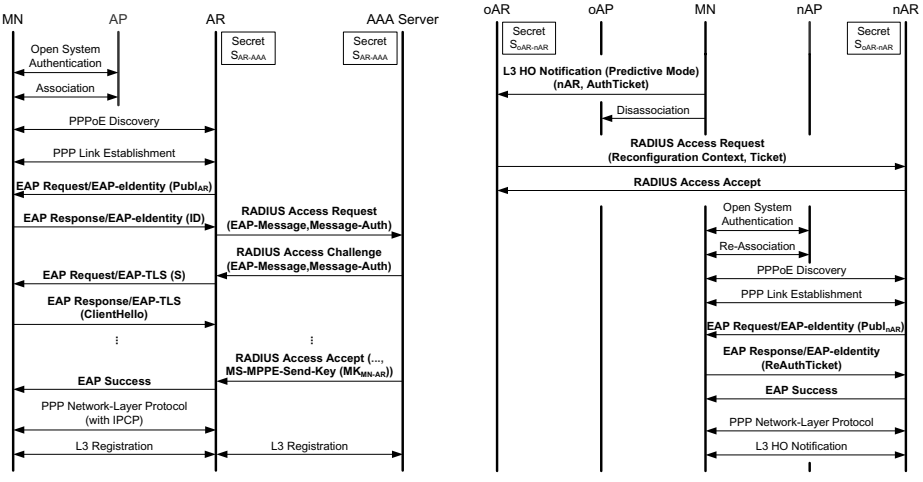
For security and maintenance reasons, the deposited authentication credentials used for authenticating a user are not directly available to the L3 PoA but rather centrally stored on an AAA server. The L3 PoA relays the authentication messages of both the MN and the AAA server and, at the end, is informed by the AAA server whether it should allow the MN to access the network or not. Message authentication for L3 signalling traffic is implemented by using key material established when the MN and the network (represented by the AAA server) authenticate each other. As the L3 PoA has to be able to create and verify authenticated messages, the AAA server transmits the necessary key material to the L3 PoA over a secured channel.

The Extensible Authentication Protocol (EAP) [4] and the Remote Authentication Dial In User Service (RADIUS) [5] are two widely-used protocols for authentication purposes. EAP provides a flexible framework allowing arbitrary authentication mechanisms. In our scenario it is used for the communication between the MN and the AAA server in which the L3 PoA acts as authenticator in pass-through mode. RADIUS is used to transport information between the AAA server and the L3 PoA. As an example scenario, to which our re-authentication scheme can be applied, in the following the authentication procedure based on Transport Layer Security (TLS) [6] is briefly described for an IEEE 802.11 environment.

The PPP EAP TLS Authentication Protocol as defined in [7] provides mutual authentication between an EAP client (here: the MN) and an EAP server (here: the AAA server) and allows to establish key material to be used for a subsequent secure communication. The L3 PoA (here: the AR) acts as the EAP authenticator. In [7] a specific EAP method called EAP-TLS is standardized that defines the transport of TLS messages within EAP messages.

The message flow for the registration of a MN with an access network is depicted in Fig. 1 (a). After associating with the AP, a link between the MN and the AR is established, e.g. by running PPPoE.

An EAP-Identity-Request is issued by the AR that is answered by the client with an EAP-Identity-Response containing an identifier. According to [4], EAP-Identity-Requests can optionally contain data to be displayed to the user and, additionally, data to be used as initialization of subsequent authentication methods. We exploit this option to implement our fast re-authentication procedure. Therefore, in Fig. 1 (a) the EAP-Identity request is already denoted by EAP-eIdentity.



(a) Startup (b) Handover

Fig. 1. Startup and Handover for IEEE-802.11-based access networks using EAP-eIdentity and EAP-TLS authentication (authentication-relevant messages in bold)

The additionally transported public key $Publ_{\mathrm{AR}}$ is an optional parameter and is ignored during startup.

As the AR has no access to the credentials for authenticating a MN, it acts as a RADIUS client and contacts the AAA server acting as RADIUS server. In [8] two attributes (`EAP-Message` and `Message-Authenticator`) are introduced to support an authenticated EAP message transport within RADIUS packets. The authentication is based on a secret key $S_{\mathrm{AR-AAA}}$ that has to be established between the AR and the AAA server by other means.

The actual EAP TLS authentication procedure uses a public key infrastructure based on certificates. Within three round trips, both the MN and the AAA server submit a random number, their public key along with a certificate chain proving the key's authenticity, and signatures computed over the messages exchanged so far to authenticate themselves. Furthermore, the MN sends a premaster secret, i.e. a random number of appropriate length, encrypted with the server's public key. The master key, i.e. the shared key material between the AAA server and the MN, is computed from these random numbers and the premaster secret.

If the EAP TLS authentication procedure is successfully finished the AAA server will send a RADIUS-Access-Accept packet to inform the AR that the MN is authenticated and can be given access to the network. This packet also contains the established master key encapsulated e.g. into `MS-MPPE-Send-Key` and `MS-MPPE-Recv-Key` attributes defined in [9]. In order to particularly protect this message, the whole communication between the AR and the AAA server should be protected by e.g. IPsec. Finally, the AR issues a EAP-Success-Message to the MN which then gets configured and registered with the core network.

3 Fast Re-authentication

As indicated in Fig. 1 (a) the conventional authentication procedure consists of several round trips between the MN and the AAA server. Furthermore, when using public key certificate chains the verification of certificates can become a time-consuming task. As packet-switched multimedia applications like video streaming or VoIP benefit from a minimal link downtime during HO, the re-authentication scheme has to be as fast as possible.

On the other hand, malicious nodes have to be prevented from misusing the re-authentication method in any way. Therefore, the new key material between a MN and its new AR should be as strong as the old one.

3.1 Attacker Model

Attackers the network has to be protected from are assumed to be mobile stations located within communication range of a MN and/or AP and trying to retrieve valuable information by eavesdropping the channel and injecting forged messages. We only consider the signalling traffic between a MN and its AR and do not address security issues on the transport or application layer as appropriate measures are available (e.g. IPsec, VPN, SSL, etc.). Furthermore, we assume that the fixed infrastructure components like ARs and AAA servers are not compromised, behave according to the protocol, and do not collude with malicious MNs. AAA servers accept RADIUS requests only from registered ARs and the communication between an AR and its AAA server is encrypted and authenticated. This can be achieved by configuring both peers with a shared secret key or by using mutually authenticated IPsec channels.

In our attacker model the attacker has the following capabilities:

- **Eavesdropping:** The attacker is able to retrieve all data packets sent over the air interface. If an encryption mechanism is in use, the attacker can only read the packet's content if it is in possession of the decryption key.
- **Forging:** The attacker is able to compile arbitrary packets and has full control over the packets' headers and data parts.
- **Simulating infrastructure devices and network services:** An attacker is able to simulate wireless access points, base stations, service gateways, etc. as long as the needed information is publicly available or has been gathered in previous attacks.
- **No resource limitations:** The attacker does not suffer from power or computational limitations that battery driven devices normally have to deal with.

Without an appropriate protection of the signalling traffic between a MN and its AR the above defined attacker would have several possibilities to attack the network. It could send forged HO requests in the name of other mobile nodes to the AR and may disconnect that node from the network. Furthermore, it could request core network services in the name of other nodes and by this gather sensitive information or trigger reconfiguration procedures. If the attacker intercepts a regular HO request it could try to connect to the destination network,

pretend to be the node that requested the HO and get unauthorized network access. As the attacker is able to simulate infrastructure components, it could also pose as the new AP/AR, trick the mobile node to connect to it, and perform at least some form of denial of service or at worst gather sensitive information. As HOs between access networks of different providers are supported it might be interesting for an old provider to further eavesdrop the signalling communication between the MN and the new provider after the HO.

Consequently, when a mobile node M performs a HO from an access network managed by the access router OA into an access network managed by NA, several requirements have to be fulfilled by the underlying re-authentication scheme:

- NA has to recognize M as being authorized to access the network. Furthermore, in order to preserve higher level security associations based on constant addresses, M should be assigned the same network address in the new access network as it had been configured with in the old one.
- The new key material for a secured channel between M and NA has to be exchanged in such a way that (1) NA cannot determine the key used between M and OA and (2) OA cannot derive the new key used by M and NA.
- Unauthorized nodes are not able to exploit the protocol to get network access or gather any other advantages.
- The delay caused by the authentication procedure has to be low.

3.2 Re-authentication Scheme

For our re-authentication scheme we assume that a secret key $S_{\mathrm{MN-oAR}}$ has been established between the MN and the old AR that can be used to exchange secured messages. Generally, this key is exchanged during startup. Furthermore, we assume the existence of a protected channel between the old AR and the new AR based on the key $S_{\mathrm{oAR-nAR}}$ established via manual configuration or by using e.g. the Internet Key Exchange protocol. In practice, an AR will probably not have very many neighbor ARs so that a manual configuration might be feasible. From the security point of view we have authenticated relationships between the MN and the old AR and between the old AR and the new AR. Our goal is to establish such a relationship between the MN and the new AR.

The message sequence of our fast re-authentication scheme is depicted in Fig. 1 (b). The MN sends a HO notification message to its current AR. It contains information about the destination access network and an authentication ticket *AuthTicket* consisting of a sequence number used to prevent replay attacks and the actual authentication information $(MN\_ID,\ nMK)_{SK}$ to be forwarded to the new AR.

$$AuthTicket := \{SeqNo, (MN\_ID, nMK)_{SK}\}_{S_{\mathrm{MN-oAR}}} \tag{1}$$

nMK is the new, randomly chosen master key between the MN and the new AR. MN_ID is the ID of the MN and is used to bind nMK to that particular MN. In order to disguise nMK from the old AR the authentication information is encrypted with the randomly chosen key SK. The *AuthTicket* is encrypted and authenticated by using the current session key $S_{\mathrm{MN-oAR}}$.

When the old AR receives the *AuthTicket* it verifies the sequence number and forwards a *Ticket*, as an attribute of a RADIUS-Access-Request packet, over the previously established secured channel to the new AR.

$$Ticket := \{MN\_ID, (MN\_ID, nMK)_{SK}\}_{S_{\mathrm{oAR-nAR}}} \tag{2}$$

The old AR prepends the *MN_ID* of the MN with which it shares $S_{\mathrm{MN-oAR}}$ to the authentication information. By this, the new AR later can verify that the *MN_ID* provided by the MN is the same as the one the MN has used to authenticate with the old AR. Along with the *Ticket* a reconfiguration context is submitted, i.e. information necessary to reconfigure the MN in the new access network. The new AR stores the *Ticket*, starts a timer, and waits for the MN to enter the network and request a fast re-authentication. If the MN does not show up before the timer runs out, the *Ticket* will be deleted. In order to re-authenticate itself to the new access network, the MN has to prove its knowledge of *nMK* stored in the corresponding *Ticket*. Furthermore, the new AR has to be enabled to extract *nMK* from the *Ticket* while preventing the old AR from doing the same.

After the conventional open system authentication with the new AP and the link establishment with the new AR, e.g. via PPPoE, the new AR requests the MN to authenticate itself. At this point in time, the new AR does not know, whether it has to handle a conventional or a fast re-authentication. As in the startup example, the new AR sends an extended EAP-Identity-Request called *EAP-eIdentity* that contains its public key $Publ_{\mathrm{nAR}}$. A MN can answer to this request with a conventional EAP-Identity-Response, thus initiating the conventional authentication scheme as described in Sect. 2. Alternatively, it can provide the re-authentication ticket *ReAuthTicket* and get authenticated immediately[1].

$$ReAuthTicket := \{MN\_ID, oAR, (SK)_{Publ_{\mathrm{nAR}}}, HMAC_{nMK}(Publ_{\mathrm{nAR}})\} \tag{3}$$

The *MN_ID* in the *ReAuthTicket* aims at identifying the correct *Ticket* from the list of currently pending tickets at the new AR. The address of the old AR *oAR* enables the new AR to contact the old AR in case that the correct *Ticket* has not yet been received. The secret key *SK* used to encrypt *nMK* in (2) is encrypted with $Publ_{\mathrm{nAR}}$. The old AR is therefore prevented from getting knowledge of *SK* by eavesdropping the *ReAuthTicket*. The last component is a hashed message authentication code (HMAC) computed over $Publ_{\mathrm{nAR}}$ and seeded with *nMK*. On the one hand it is generated to prove knowledge of *nMK* and on the other hand to indicate that it is generated to answer an EAP-eIdentity-Request with $Publ_{\mathrm{nAR}}$. The new AR uses its private key to extract *SK* from (3) and uses *SK* to decrypt the new master key *nMK* and the encrypted *MN_ID* from (2). By verifying that both *MN_IDs* from (2) match, it is ensured that the MN has

[1] Although the EAP standard [4] states to send EAP-Success-Messages "after completion of an EAP authentication method (Type 4 or greater)", sending an EAP-Success-Message already after receiving an EAP-Identity-Response (Type 1) is not explicitly forbidden.

transmitted the *MN_ID* it has used to authenticate itself with the core network. A successful verification of the HMAC provided in (3) proves that the issuer of the *ReAuthTicket* knows *nMK*. As the *Ticket* comes from a trusted peer (the old AR) to which the MN had authenticated itself, the MN is successfully authenticated with the new AR and the IP address stored in the reconfiguration context can be assigned to the MN with ID *MN_ID*. Finally, the core network is informed about the performed HO operation and the new AR can delete the used ticket from its list of pending tickets.

Although intended to be used in predictive mode it is also possible to apply the re-authentication scheme in reactive mode. For this, the MN would have to transmit the *AuthTicket* (1) along with the *ReAuthTicket* (3) in the EAP-Identity-Response. The new AR would transmit the *AuthTicket* to the old AR which could then transmit the reconfiguration context along with the *Ticket* (2) to the new AR. Afterwards, the verification of the *ReAuthTicket* can be performed as described above.

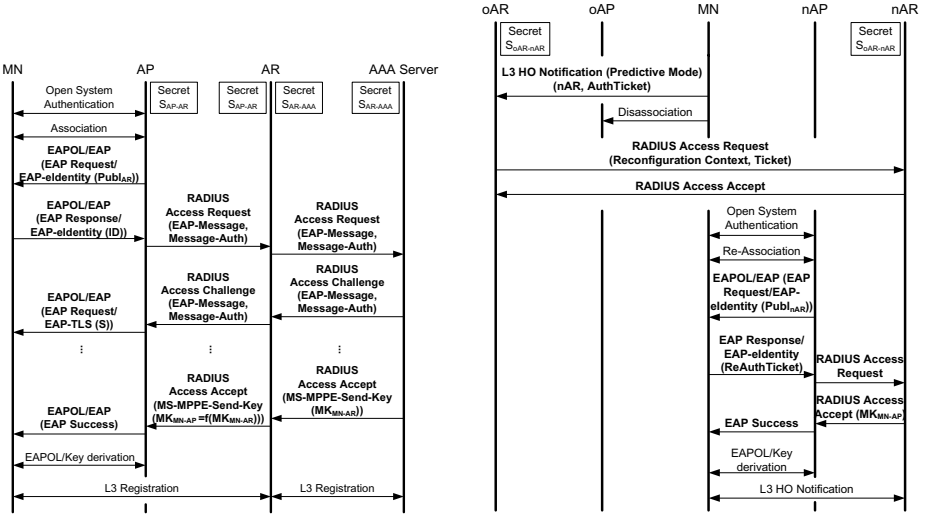
In the case the MN decides to re-attach itself to the old AR instead of the new AR, the fast re-authentication can be applied analogously, with the special case that old AR and new AR are identical.

4 Application to IEEE 802.11i Environments

In IEEE 802.11i environments an AP itself runs authentication procedures. It can be configured to act as an EAP authenticator and to negotiate a master key between a MN and itself by using the mechanisms described in Sect. 2. As the AR is no longer actively involved in this process a key between the MN and the AR used to protect L3 signalling traffic is not established.

As a solution we propose to configure the AP to use its AR as RADIUS server. With exception of the final RADIUS-Access-Accept packet, the AR acts like a RADIUS proxy by relaying the RADIUS packets between AAA server and AP (cf. Fig. 2 (a)). The MN and the AAA server negotiate a master key $MK_{\mathrm{MN-AR}}$ by running EAP-TLS and the AAA server transmits $MK_{\mathrm{MN-AR}}$ in its final RADIUS-Access-Accept packet to the AR. The AR uses $MK_{\mathrm{MN-AR}}$ on the one hand to derive session keys for a secured communication with the MN and on the other hand to derive the master key $MK_{\mathrm{MN-AP}}$ for the AP with a publicly-known one-way function. $MK_{\mathrm{MN-AP}}$ is transmitted to the AP in the corresponding RADIUS-Access-Accept packet. On receipt of the EAP-Success-Message from the AP, the MN also computes $MK_{\mathrm{MN-AP}}$ from $MK_{\mathrm{MN-AR}}$.

Analogously, the fast re-authentication is implemented for this scenario (cf. Fig. 2 (b)). The ticket is created in the same manner as described before and transmitted via the old AR to the new AR. After a successful L2 re-association the re-authentication is slightly different from the one illustrated in Fig. 1 (b). The new AP has to start the authentication procedure by sending the initial EAP-eIdentity-Request containing its AR's public key $Publ_{\mathrm{nAR}}$. The corresponding EAP-Response with the re-authentication ticket inside is addressed to the new AP being from the MN's point of view the EAP authenticator. The new



(a) Startup (b) Handover

Fig. 2. Startup and Handover for IEEE-802.11i-based access networks using EAP-eIdentity and EAP-TLS authentication

AP forwards the response via RADIUS to the new AR which verifies the re-authentication ticket, extracts the new master key nMK, derives the master key $MK_{\mathrm{MN-AP}}$ to be used between the MN and the new AP, and transmits $MK_{\mathrm{MN-AP}}$ encapsulated into a RADIUS-Access-Accept packet to the new AP. The MN derives $MK_{\mathrm{MN-AP}}$ from nMK in the same way.

5 Evaluation

In this section the fast re-authentication scheme is evaluated according to the requirements of Sect. 3.1.

5.1 Security Evaluation

The security evaluation of the proposed re-authentication scheme is based on the assumptions that the underlying cryptographic functions are secure and that random numbers are not predictable. Furthermore, we assume that the EAP TLS authentication with RADIUS as described in Sect. 2 is secure.

On the one hand it has to be shown that authorized nodes – clients as well as servers – are able to successfully perform a fast re-authentication. On the other hand any attacker with capabilities as described in Sect. 3.1 must be prevented from misusing the scheme to get any advantages.

Authorized nodes are able to re-authenticate each other: A node that has been granted access to the old access network has to be granted access to the new one and it can be configured with the same network address as before.

Furthermore, any AR that is supposed to be the new AR for a MN, has to have the ability to verify an authorized request and to prove to the MN that it is the component it claims to be. Obviously, a MN is able to create a matching pair of *AuthTicket* (1) and *ReAuthTicket* (3) and the new AR has the ability to verify *ReAuthTicket* with information from *Ticket* (2) as described in Sect. 3.2. The configuration context transmitted along with the *Ticket* allows the new AR to configure the MN with the same network address as before. By using key material derived from *nMK* to authenticate subsequent signalling messages, the AR can prove its eligibility to the MN. A minor drawback of our scheme is the fact that the new AR is not authenticated until it sends the first signalling message that has to be protected. A challenge/response mechanism could resolve this issue but it would mean an additional round trip between MN and AR or a modification of the final EAP-Success-Message.

Attack prevention: At first we prove a lemma from which the other security properties can easily be derived.

Lemma 1. *If the above-mentioned assumptions are fulfilled an attacker with the capabilities as described in Sect. 3.1 is not able to retrieve nMK from the fast re-authentication process.*

Proof. The value *nMK* is randomly chosen by the MN. As neither the old AR nor the new AR nor the MN are compromised and random numbers are unpredictable, the only chance for an attacker to gain knowledge of *nMK* is to derive it from messages. By passively eavesdropping the air interface, the attacker can obtain *AuthTicket* (1) and *ReAuthTicket* (3). Furthermore, it can get knowledge of *SK* by actively forging an initial EAP-eIdentity-Request with its own public key $Publ_{\mathrm{A}}$. If the public key is not enriched with a certificate – which is, due to performance reasons, assumed to be the case – the MN will issue the re-authentication ticket based on that public key and therefore reveal *SK* to the attacker.

As the underlying cryptographic functions are secure, $(MN\_ID, nMK)_{SK}$ remains unintelligible to the attacker (it is encrypted with $S_{\mathrm{MN-oAR}}$ to which the attacker has no access). Therefore, although knowing *SK* the attacker is not able to obtain *nMK*. The only other occasion *nMK* is transmitted is from the old AR to the new AR and that communication channel is required to be secured (encrypted and authenticated). Furthermore, as both the old AR and the new AR are not compromised, they will not communicate *nMK* over any other channel. The only remaining information sources are $HMAC_{nMK}(Publ_{\mathrm{A}})$ and $HMAC_{nMK}(Publ_{\mathrm{nAR}})$ contained in the respective *ReAuthTickets*. But as the HMAC function is defined to be a one-way function, both values do not reveal any information about *nMK*. □

As signalling messages are authenticated with key material derived from *nMK*, an attacker is unable to send messages in the name of other nodes. The replay of overheard messages is prevented by the introduced sequence numbers.

As ARs only accept tickets from trusted partners, it is impossible for an attacker to deposit a forged ticket at an AR and authenticate itself based on

that ticket. Furthermore, in order to create a re-authentication ticket to a ticket not originated by itself, an attacker would have to know the *nMK* stored in that ticket, which is according to Lemma 1 impossible. For the same reason, it is impossible for an attacker to pose as the AP/AR a MN is supposed to connect to. In order to be able to send messages that are accepted by the MN, the attacker would have to have the ability to produce messages with authentication credentials based on *nMK*.

A last security requirement is that neither the new AR can determine the old key material $MK_{\mathrm{MN-oAR}}$ nor can the old AR determine the new key material $MK_{\mathrm{MN-nAR}}$. As the new AR is not at all involved in the key establishment between the MN and the old AR, it is at most as powerful as a conventional MN and therefore not able to gather any information about $MK_{\mathrm{MN-oAR}}$. The old AR is able to extract $(MN\_ID, nMK)_{SK}$ out of *AuthTicket*. All it needs is *SK* to decrypt *nMK*. But *SK* is only transmitted in *ReAuthTicket* and is encrypted by $Publ_{\mathrm{nAR}}$. As the old AR does neither launch active attacks nor collude with other mobile nodes, *SK* is not revealed to it.

5.2 Performance Considerations

The fast re-authentication protocol consists of only three messages exchanged between the MN and the authenticator (AP or AR). In the IEEE 802.11i scenario two additional messages between the new AP and the new AR accrue. Unlike during the startup with an AAA server involved, the whole communication is held local as the AR generally belongs to the same layer 2 network as the AP.

The computational overhead to compile the fast re-authentication messages is also low. In order to create *AuthTicket*, the mobile node has to draw two random numbers and to perform two symmetric encryption operations. The first message of the re-authentication procedure sent by the new AR is independent from the client and requires no computational resources. A more complex task is the creation of the *ReAuthTicket* as the MN has to use an asymmetric encryption algorithm to encrypt the secret key *SK* with the new AR's public key $Publ_{\mathrm{nAR}}$. The computational overhead can be limited e.g. by using encryption friendly public keys like small RSA exponents. In case the MN already knows $Publ_{\mathrm{nAR}}$ in advance, the re-authentication ticket can also be prepared in advance. The MN may have re-authenticated with that particular AR before and have cached its public key, or the old AR may provide a service that mobile nodes can use to request public keys of certain ARs before initiating HOs. The other operations necessary to compile the *ReAuthTicket* like HMAC-computation are fast operations. With the decryption of *SK* the new AR has to apply only one asymmetric operation. The ticket's decryption and verification require only one symmetric operation and a HMAC computation.

6 Conclusion

In this paper we have addressed the issue of providing a fast mechanism for the re-authentication of mobile nodes performing handovers across access network

boundaries. A ticket-based predictive mechanism has been defined that modifies well-established protocol stacks only slightly. In essence, a ticket generated by the MN is forwarded by the current L3 PoA to the destination L3 PoA and the MN re-authenticates itself by proving knowledge of the ticket's contents. The (re-)authentication with an access network is initiated by a modified EAP-Identity-Request called EAP-eIdentity which can be answered by a conventional EAP-Identity-Response or with an EAP-Identity-Response that contains a re-authentication ticket. In the first case, the normal authentication procedure is run whereas in the latter case the MN is immediately authenticated.

Furthermore, we have shown the application of our scheme to IEEE 802.11i like environments and how RADIUS can be used to distribute key material derived from the authentication process. The scheme has been evaluated with respect to security and performance properties. We have shown that our scheme is secure when dealing with single attackers located as mobile nodes in the area covered by the access network. We have further shown that the scheme is efficient in terms of the number of necessary round trips, the time complexity of the underlying operations, and the computational overhead for mobile nodes.

Future work will include hardening the scheme against compromised core network entities and against collusions of mobile nodes with infrastructural devices.

References

1. R. Droms, *Dynamic host configuration protocol*, IETF RFC 1541, March 1997.
2. P. Jayaraman, R. Lopez, Y. Ohba, M. Parthasarathy, and A. Yegin, *PANA Framework*, IETF, Internet-Draft, July 2005. [Online]. Available: www.ietf.org/internet-drafts/draft-ietf-pana-framework-05.txt
3. L. Mamakos, K. Lidl, J. Evarts, D. Carrel, D. Simone, and R. Wheeler, *A Method for Transmitting PPP Over Ethernet (PPPoE)*, IETF RFC 2516, February 1999.
4. B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz, *Extensible Authentication Protocol (EAP)*, IETF RFC 3748, June 2004.
5. C. Rigney, S. Willens, A. Rubens, and W. Simpson, *Remote Authentication Dial In User Service (RADIUS)*, IETF RFC 2865, June 2000.
6. T. Dierks and C. Allen, *The TLS Protocol version 1.0*, IETF RFC 2246, January 1999.
7. B. Aboba and D. Simon, *PPP EAP TLS Authentication Protocol*, IETF RFC 2716, October 1999.
8. B. Aboba and P. Calhoun, *RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)*, IETF RFC 3579, September 2003.
9. G. Zorn, *Microsoft Vendor-specific RADIUS Attributes*, IETF RFC 2548, March 1999.