Authenticated Wireless Roaming via Tunnels: Making Mobile Guests Feel at Home*

Mark Manulis[1], Damien Leroy[2], Francois Koeune[1], Olivier Bonaventure[2], and Jean-Jacques Quisquater[1]

[1]UCL Crypto Group and [2]IP Networking Lab, Dept CSE
Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium

{mark.manulis|damien.leroy|francois.koeune|olivier.bonaventure|jean-jacques.quisquater}@uclouvain.be

Abstract. In wireless roaming a mobile device obtains a service from some foreign network while being registered for the similar service at its own home network. However, recent proposals try to keep the service provider role behind the home network and let the foreign network create a tunnel connection through which all service requests of the mobile device are sent to and answered directly by the home network. Such *Wireless Roaming via Tunnels (WRT)* offers several (security) benefits but states also new security challenges on authentication and key establishment, as the goal is not only to protect the end-to-end communication between the tunnel peers but also the tunnel itself. In this paper we formally specify mutual authentication and key establishment goals for WRT and propose an efficient and provably secure protocol that can be used to secure such roaming session. Additionally, we describe some modular protocol extensions to address resistance against DoS attacks, anonymity of the mobile device and unlinkability of its roaming sessions, as well as the accounting claims of the foreign network in commercial scenarios.

Key words: Authentication, end-to-end security, key exchange, mobile networks, security model, tunnel, wireless roaming

1 Introduction

In the today's world of mobility one may observe the increasing popularity of wireless networks and devices, both in companies and private households. In many environments wireless technology is the default access technology for a variety of services. Its ubiquity combined to the demands on mobility leads to the interesting research area of *wireless roaming*.

The main goal of wireless roaming is to provide a *mobile device* that is registered at its own *home network* where it can access certain services, e.g. diverse Internet and mobile (phone) applications, with similar services when it roams to another *foreign network*. The problem is that the foreign network is usually not aware whether the mobile device is authorized to request a roaming connection, and the mobile device may not know whether the foreign network is authorized to provide this connection (as a contract partner of the home network).

Current solutions for wireless roaming deployed in wireless local networks (WLANs) and also in mobile phone networks (GSM, CDMA, UMTS, CDMA2000) assume that the requested service is provided directly by the foreign network. The actual involvement of the home network is reduced to the necessary actions related to the authentication of the mobile device and the foreign network as well as the establishment of a security association (e.g., session keys) between these two entities.

The recent proposal for wireless roaming by Sastry et al. [54] is conceptually different. Instead of considering the foreign network as the actual service provider they suggest to keep this role behind the home network by establishing an opaque *tunnel* connection between the networks and using it to provide the mobile device with the requested service.

The main technical contribution of our work is the design of a formal security model for authentication and key establishment in *Wireless Roaming via Tunnels* (WRT, for short) and its realization through a suitable protocol. Section 2 introduces the WRT concept as a solution for various practical problems.

* This work is supported by the Belgian Walloon Region under its RW-WIST Programme, ALAWN Project.

Section 3 describes related work on authentication and key establishment in wireless roaming. In the scope of the security model in Section 4 we describe basic trust assumptions on the protocol participants and specify necessary mutual authentication and key exchange requirements. For the proposed protocol, called `AWRT`, those basic version is described in Section 5 we give detailed security analysis with respect to the specified security goals, provide some efficiency remarks and brief ideas on the practical realization of the tunneled connection. In Section 6 we address several (modular) protocol extensions dealing with forward secrecy for the established keys, stricter resistance against various types of DoS attacks, anonymity and unlinkability issues for the roaming mobile device, and the use of the protocol in commercial scenarios, in which the foreign network is supposed to be reimbursed for the maintenance of the tunnel. We conclude in Section 7.

2 Wireless Roaming via Tunnels

In the following, we introduce the general concept of WRT and illustrate the main differences between our architecture and the original proposal [54]. We then present the advantages and disadvantages of using tunnels as an alternative way to obtain services. Finally, we identify the desired security goals for this new approach.

2.1 The Concept

The main scenario we deal with is that a mobile device, that belongs to a home network (e.g., its company) temporary moves to a foreign network which cannot authenticate it directly. The mobile user would like to access the Internet as a general service from this network under the same conditions and (security and network) policies as within its home network. For this, an opaque tunnel is created from the foreign network to the home network and all the packets sent by the mobile device are sent via this tunnel to its home network that forwards them to the Internet if appropriate.

The only currently available solution for the tunnel-based roaming by Sastry et al. [54] is based on VPN tunnels securing the end-to-end communication between the mobile device and its home network. In their scheme the foreign network accepts every device without any authentication and grants it access to the home network over the Internet. The mobile device can thus initiate a VPN connection (using NAT traversal techniques if necessary). However, this solution has several weaknesses. First, the Internet access granted by the foreign network, even a restricted one, may bear intrusion risks to its infrastructure. Second, the mobile device must comply with the network layer infrastructure of the foreign network (e.g., IPv4/IPv6, IP assignment via DHCP). Third, VPN tunnels do not provide any proof to the foreign network that the mobile device is connecting to its real home network as a VPN connection can be established to any server on the Internet. Fourth, foreign and home networks do not authenticate each other and, as a consequence, neither accounting mechanisms nor quality-of-service contracts can be securely implemented. We fairly remark that Sastry et al. were focusing on the actual architecture for the city-wide WiFi roaming rather than dealing with the related authentication and key establishment goals.

Our approach is different. We increase the involvement of the foreign network into the roaming process by considering it as an inherent part of the security architecture with the obvious goal to eliminate the above mentioned weaknesses. In our setting the foreign network does not simply agree to grant the mobile device a restricted Internet access to the claimed home network but is responsible for the establishment of the tunnel connection between the both after having verified their authenticity. To achieve the desired goals we propose a three-party authentication and key establishment protocol between the mobile device, the foreign and the home network. This protocol permits the authentication of participants and exchange of relevant session keys — not only for the end-to-end secure communication between the device and its home network but also for the secure communication with the foreign network. The successful execution of our protocol means that the foreign network will tunnel all the data sent by the mobile device from its access point to its home network, lowering the risk of intrusion into its own infrastructure.

2.2 Pro and Contra

The concept of WRT appears attractive mostly because of the additional security benefits which it offers and because it shifts the role of the service provider back to the home network, this in contrast to the non-tunnel-based approach.

WRT offers several benefits for the foreign network. First, the automatic forwarding of messages provides better protection of the foreign network's infrastructure, reducing the risk of unauthorized access and intrusion attempts by malicious devices. Second, in WRT the mobile device appears to the outside world (e.g., when accessing a server on the Internet) as part of its home network. Therefore, if a mobile device misbehaves (e.g., up-/downloads illegal content or mounts DoS attacks) causing investigation and IP traceback, the foreign network cannot be blamed or blacklisted for it; Robert et al. [51] argue that WRT would lower the risk for the involvement of the foreign network into the legal investigations resulting from such malicious activities. Additionally, using WRT effectively prevents the mobile device from using any value-added services to which the access is granted based on the IP membership within the domain of the foreign network (e.g., access to digital libraries).

WRT still requires mutual authentication between the mobile device and its home network. This eliminates impersonation attacks in case that the foreign network misbehaves. In contrast to the non-tunnel-based approach, WRT may effectively prevent DNS manipulations by the foreign network; thus, thwarting pharming attacks [58]. In WRT the home network may effectively enforce own security policies regarding the provided service. In particular, the mobile device may access the same set of services while it roams, whereas in the non-tunnel-based approach it can access services offered by the foreign network. For example, considering the connection to the Internet, some networks may impose firewall restrictions on the use of diverse Internet application protocols. A mobile user which roams to a network with stricter policies can then still execute applications he is used to within his own home network.

In commercial roaming scenarios where the foreign network has to be reimbursed for the provided roaming service, WRT can significantly simplify and strengthen the accounting process since both networks can keep track on the duration of the roaming connection and the amount of data being transferred independently of each other.

One may think, that the use of WRT would result in the significant increase of the communication latencies, as every service request has to be sent and answered through the communication link between the foreign and the home network. Obviously, this delay which relates to the *round-trip time* could vary depending on the networks and their geographic location. For example, it has been shown that in country-wide context (on the example of USA) the expected round trip latencies for the TCP/IP traffic remains below 150 ms [38], while in the intercontinental context they remain below 250 ms for over 90% of the residential broadband hosts [26]. These examples show that for many applications such as web surfing and email the additional latencies in WRT would remain almost unnoticeable, and the quality of provided service may still be sufficient even for some real-time applications such as Voice-over-IP for which – according to the ITU-T recommendations [35] – a *one-way* trip latency below 400 ms might still be acceptable.

2.3 Desired Security Goals

Suitable WRT authentication and key establishment protocols should take care of protecting not only the end-to-end communication between the mobile device and the home network but also their communication with the foreign network.

Obviously, the mobile device and its home network must authenticate each other prior to the use of the tunnel and establish a session *end-to-end* key, that is considering the foreign network as a potential man-in-the-middle attacker. To the contrary, in the non-tunnel-based approach the authentication of the home network towards the mobile device is usually not required and the session key for the roaming communication is established between the mobile device and the foreign network.

Further, it is desirable for both networks to mutually authenticate each other prior to the establishment of the tunnel. This would lower the risk of attacks against the infrastructure of both networks, including DoS attacks. In commercial scenarios where the foreign network should be reimbursed for the established

tunnel this mutual authentication would be useful to implement the accounting process, possibly in real-time. Further, since the mobile device and the foreign network may not be aware of each other prior to the protocol execution, the mutual authentication between the networks would indirectly prevent roaming sessions where either the mobile device (hosted by the home network) or the foreign network is not authorized to request or establish the tunnel connection, respectively.

In WRT the actual service requests are sent and answered through the established tunnel and can, therefore, be protected using end-to-end keys. However, the home network and the mobile device may also wish to exchange certain control messages directly with the foreign network; in particular, checks that the mobile device and the home network are still connected, requests to close the established connection, and accounting messages in commercial scenarios. In order to protect such messages and also achieve better robustness of the roaming session (in particular, against session hijacking and data injection attacks) it is desirable to provide the foreign network and the tunnel end-peers with an additional session *tunnel* key; this key is specific to WRT and is not required for the non-tunnel-based roaming.

In addition to the previously mentioned mutual authentication and key establishment goals those formal specification and realization represents the main focus of this work, there are additional goals which we informally address to the end of the paper. Some of them aim to achieve higher robustness of WRT, e.g., protection against DoS attacks. The goal here is to minimize the risk that one party keeps an open connection to another party during the protocol execution without being able to verify whether that party is a valid protocol participant. Another desired goal for WRT in commercial scenarios is to provide the foreign network with some verifiable information to compose its accounting claims. Further security goals address the profiling of the mobile users. In particular, it might be desirable to hide the user's identity from the guest network and achieve unlinkability of its roaming sessions as it is currently the case in GSM/UMTS through the use of TMSI.

3 Related Work on Authentication and Key Establishment in Wireless Roaming

Many authentication and key establishment protocols for wireless roaming have been proposed so far, in both academia and industry (as part of standardization), ranging from mobile phone networks of the 2nd generation (GSM, CDMA) and 3rd generation (UMTS, CDMA2000) over to WLANs (IEEE 802.11 [1]).

3.1 Wireless Roaming in Mobile Phone Networks.

Most of the standard authentication and key establishment protocols for mobile phone networks, e.g., [2, 29, 52], are based on the pre-shared key between the home network and its mobile device. These protocols establish the session key between the mobile device and the foreign network. Several solutions have been further proposed to allow roaming among different mobile phone networks. For example, the authenticated roaming between GSM and UMTS has been specified within the UMTS standard [2] and partially addressed in [3], and the roaming procedure between UMTS and CDMA2000 has been addressed in [37].

3.2 Wireless Roaming in IP Networks.

In wireless IP networks, the access control is usually implemented based on the IEEE 802.11i security architecture, either using pre-shared keys (WPA, WPA2) or using the IEEE 802.1X specification. Although pre-shared keys are widely used for home and small-office networks, these does not actually fit for roaming. In larger networks, preference is given to IEEE 802.1X that is based on EAP [5], a protocol framework for the transmission of the authentication information between clients and networks. The actual protocols implemented within this framework are referred to as EAP methods and can be based for example on usernames and passwords, shared keys, or public-key certificates (supplied by the client, the server, or both). EAP using TTLS [32] is used for roaming purpose within the *Eduroam* infrastructure composed of the European education institutions that have reached an agreement [28]. *Eduroam* uses

EAP-TTLS in association with RADIUS servers hosted in each partner to authenticate and grant an Internet access to students and personnel visiting another institution. Another widely used form of authentication in WLANs, recommended by Wi-Fi alliance as a best practice solution [6], is web-based such that the mobile user provides own credentials as input to the browser form. This technique when based on the username / password is also known as the Universal Access Method (UAM), see also [7, 8, 41] for some variations. These credentials are usually forwarded to the authentication server (e.g., RADIUS). As noticed in [43] such web-based solutions become vulnerable to the access point impersonation, address spoofing, and dictionary attacks. Some improvements to UAM supporting further types of credentials reducing the required user interaction to opening a web-browser have been described by McCann et al. [41].

Salgarelli et al. [53] suggested a general roaming authentication framework based on the shared keys which can be implemented as an EAP method. Their protocol extends the classical Needham-Schroeder technique [46] to accommodate the authentication servers of the foreign and the home network while minimizing the communication rounds between them. Previously, Molva et al. [45] described another roaming protocol based on shared keys, which was designed for the integration into the IBM's KryptoKnight authentication and key distribution framework. Merino et al. [42] proposed a Single Sign-On authentication architecture based on 802.1X and EAP-TLS [57] relying on the Public Key Infrastructure (PKI). Their method can be combined with any web-based authentication method, e.g., UAM. The drawback of this approach is that the mobile device is assumed to be able to check the validity of the foreign network's certificate while being off-line. Furthermore, the use of public-key operations might be costly for performance-constraint mobile devices. Similar drawbacks appear in the authentication protocols from [9, 33]. Long et al. [40] suggested a roaming protocol based on the modified SSL handshake assuming that mobile device are equipped with public-key certificates, so that the protocol can be executed without active involvement of the home network. Ribeiro et al. [49] described a roaming authentication approach based on IPsec VPNs and a hierarchy of certification authorities. The aforementioned problems with validation of public-key certificates by the mobile device were solved by Meyer et al. [44] via secret sharing technique [55]. In their protocol described as an extension of EAP-TLS [57] each foreign network is assumed to hold a share of the home network's secret key and the respective public-key certificate of the home network is pre-installed at the mobile device. During the execution of the protocol (which is a modified TLS handshake) the foreign and the home network need to cooperate in order to perform the required signature and decryption operations.

Additionally, we mention the commercial system Fon [30], which sells own Wi-Fi routers that mediate the authentication of mobile devices to a Fon server using MAC/IP address filter technique. As noted in [54] the deployed address filter technique allows address spoofing attacks. Another commercial system Wisher [60] requires that foreign networks distribute WPA keys to authorized guests. Obviously, this approach does not protect from the redistribution of the obtained keys by possibly malicious guests.

The aforementioned solutions proposed for wireless non-tunnel-based roaming (in mobile phone and wireless IP networks) have been designed with the main goal to authenticate (and provide a session key to) the mobile and the foreign network, whereby some approaches require the interaction with the home network. The only currently available solution for the tunnel-based roaming by Sastry et al. [54] has already been discussed in Section 2.1.

4 Authentication and Key Establishment Model for WRT

Here we model authentication and key establishment goals of a WRT protocol (denoted as Π within the model). Our definitions extends the classical two-party model from [13].

4.1 Communication Model

Protocol Participants and Long-Lived Keys We consider a *home network* $\mathcal{H}$, a *mobile device* $\mathcal{M}$ registered with $\mathcal{H}$, and a *foreign network* $\mathcal{F}$ as participants of Π. In practice $\mathcal{H}$ and $\mathcal{F}$ can be seen as corresponding authentication servers. We do not distinguish between the participants and their identities,

which are assumed to be unique; the identity of a mobile device $\mathcal{M}$ is assumed to be unique within its own home network $\mathcal{H}$. All protocol participants are modeled as probabilistic polynomial time (PPT) machines. We assume that participating $\mathcal{H}$, $\mathcal{F}$, and $\mathcal{M}$ are in possession of their corresponding *long-lived keys* LL_P, $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$, which are used in different executions of Π. We assume that all long-lived keys as well as further secrets used in Π are polynomially bounded with respect to some *security parameter* κ.

Instances and Protocol Sessions In order to model participation of $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ in distinct sessions of Π we consider an unlimited number of instances: By $[P, s]$ we denote the *s-th instance of* $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$ where $s \in \mathbb{N}$.

An instance $[P, s]$ may be invoked for one session. Throughout the protocol execution $[P, s]$ may learn its unique publicly known *session id* sid_P^s. Instances of $\mathcal{M}$, $\mathcal{F}$ and $\mathcal{H}$ that hold identical session ids $sid_{\mathcal{M}}^s = sid_{\mathcal{F}}^s = sid_{\mathcal{H}}^s$ are *partnered*, i.e. participate in the same session.

Upon the protocol invocation the instance $[P, s]$ is initialized with the long-lived key LL_P and turns into a *processing* state where it proceeds according to the protocol specification, until it collects enough information to decide whether the protocol execution was successful (we say an instance *accepts*) or not (we say an instance *aborts*). Finally, the instance *terminates* meaning that it stops processing any further protocol messages. In a WRT protocol different participants have different acceptance criteria, i.e., an instance of $\mathcal{F}$ accepts when it is ready to create the tunnel, in particular after the computation of the *session tunnel key* $K_t \in \{0,1\}^\kappa$, whereas the instances of $\mathcal{M}$ and $\mathcal{H}$ accept when they are ready to communicate with each over the tunnel, in particular after the computation of the *session end-to-end key* $K_{\mathcal{M},\mathcal{H}} \in \{0,1\}^\kappa$ (in addition to K_t).

4.2 Security Model

Security Associations and Commitments The mobile device $\mathcal{M}$ and its home network $\mathcal{H}$ are assumed to maintain some security association (as a result of the initialization), and to accept the provided tunnel connection if they can successfully authenticate each other upon the tunnel establishment.

To the contrary, there is no security association between $\mathcal{M}$ and $\mathcal{F}$ prior to the execution of Π, i.e. $\mathcal{M}$ and $\mathcal{F}$ need not to be aware of each other, which is fairly natural in the case of roaming. Therefore, in questions related to the authorized WRT participation both, $\mathcal{M}$ and $\mathcal{F}$, rely on $\mathcal{H}$.

On the other hand, as part of their contract we assume that $\mathcal{F}$ creates a tunnel to $\mathcal{H}$ if it successfully authenticates $\mathcal{H}$, whereas $\mathcal{H}$ accepts the provided tunnel after the successful authentication of $\mathcal{F}$ (in addition to the authentication of $\mathcal{M}$). Nevertheless, this does not rule out attacks by malicious $\mathcal{F}$ trying to impersonate $\mathcal{M}$ towards $\mathcal{H}$.

Adversarial Model The adversary $\mathcal{A}$ modeled as a PPT machine is assumed to have complete control over the protocol invocation and the communication channels. Additionally, we allow $\mathcal{A}$ to corrupt parties. However, we will restrict this latter ability of $\mathcal{A}$ in a meaningful way upon defining the actual security goals. We model possible actions of $\mathcal{A}$ through the following set of queries:

- $\mathsf{Invoke}(P, m)$: This is the protocol invocation query that can be asked for some entity $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$. In response, a new instance $[P, s]$ is created and $\mathcal{A}$ is given its first outgoing message. The optional input m indicates the message expected by the instance to start the execution; for the initiator of the protocol m is supposed to be empty.
- $\mathsf{Send}(P, s, m)$: This query models communication control by $\mathcal{A}$ and contains a message m which should be delivered to the s-th instance of $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$. The adversary can decide honestly to forward protocol messages between the instances in order to eavesdrop the protocol execution (that is $\mathcal{A}$ remains *passive*) or to manipulate or inject messages (that is $\mathcal{A}$ becomes *active*). In response, $\mathcal{A}$ receives the outgoing message of $[P, s]$, or an an empty message if $[P, s]$ terminates having processed m.
- $\mathsf{Corrupt}(P)$: This query models corruptions of $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$. In response, $\mathcal{A}$ receives LL_P. As soon as $\mathcal{A}$ corrupts P, all instances of P are also treated as corrupted.

- $\mathsf{RevealKey}(P, s)$: This query models independence of end-to-end keys computed by the instances of $P \in \{\mathcal{M}, \mathcal{H}\}$ in different sessions. In response, $\mathcal{A}$ is given $K_{\mathcal{M},\mathcal{H}}$ held by the instance; the query is answered only if $[P, s]$ has accepted.
- $\mathsf{RevealTunnelKey}(P, s)$: This query models independence of tunnel keys computed by the instances of $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$ in different sessions. In response, $\mathcal{A}$ is given K_t held by the instance; the query is answered only if $[P, s]$ has accepted.

Correctness The following definition of correctness, given from the perspective of one particular session, specifies the purpose of Π with respect to the tunnel creation and key establishment.

Definition 1 (Correctness). *An authentication and key establishment protocol for WRT Π is* correct *if in the presence of a passive adversary $\mathcal{A}$ the invoked instances of $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ terminated having accepted and **all** of the following holds: $\mathcal{M}$ and $\mathcal{H}$ hold the same end-to-end key $K_{\mathcal{M},\mathcal{H}}$; $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ hold the same tunnel key K_t.*

Security Goals We start by defining the requirement of mutual authentication (MA) between the instances of $\mathcal{M}$ and $\mathcal{H}$. In order to reduce the complexity of the model (and proofs) Definition 2 captures additional sub-goals related to the agreement on the session end-to-end and tunnel keys computed by the instances of $\mathcal{M}$ and $\mathcal{H}$, i.e., that at the end of the successful protocol execution both instances hold identical keys (these sub-goals are expressed through conditions 3 and 4). The significant difference is that for the agreement on the tunnel key K_t, the foreign network $\mathcal{F}$ should remain uncorrupted, whereas no such restriction is made for the mutual authentication between $\mathcal{M}$ and $\mathcal{H}$ and the agreement on the end-to-end key $K_{\mathcal{M},\mathcal{H}}$. This models possible attacks of a malicious $\mathcal{F}$ trying to impersonate either $\mathcal{M}$ or $\mathcal{H}$ or to influence a disagreement on the established session end-to-end key $K_{\mathcal{M},\mathcal{H}}$. Since the tunnel key K_t will be used to protect control messages for the maintenance of the tunnel (and not to protect the actual service requests of $\mathcal{M}$ and responses of $\mathcal{H}$) we refrain from consideration of malicious $\mathcal{M}$, $\mathcal{H}$, or $\mathcal{F}$ aiming to compromise the agreement on K_t.

Definition 2 (MA between $\mathcal{M}$ and $\mathcal{H}$). *Given a correct protocol Π by $\mathsf{Game}_{\Pi}^{\mathsf{ma\text{-}m\text{-}h}}(\mathcal{A}, \kappa)$ we denote the interaction between the instances of $\mathcal{M}$, $\mathcal{F}$ and $\mathcal{H}$ with a PPT adversary $\mathcal{A}$ that is allowed to query* Invoke, Send, Corrupt, RevealKey, *and* RevealTunnelKey. *$\mathcal{A}$* wins *if at some point during the interaction:*

*(1) an uncorrupted instance of $\mathcal{M}$ accepts but there is **no** uncorrupted partnered instance of $\mathcal{H}$, **or***
*(2) an uncorrupted instance of $\mathcal{H}$ accepts but there is **no** uncorrupted partnered instance of $\mathcal{M}$, **or***
*(3) uncorrupted partnered instances of $\mathcal{M}$ and $\mathcal{H}$ accept without holding the same session end-to-end key $K_{\mathcal{M},\mathcal{H}}$, **or***
(4) $\mathcal{F}$ is uncorrupted and uncorrupted partnered instances of $\mathcal{M}$ and $\mathcal{H}$ accept without holding the same session tunnel key K_t.

The maximum probability of this event over all adversaries (running in time κ) is denoted

$$\mathsf{Succ}_{\Pi}^{\mathsf{ma\text{-}m\text{-}h}}(\mathcal{A}, \kappa) = \max_{\mathcal{A}} |\Pr[\mathcal{A} \text{ wins in } \mathsf{Game}_{\Pi}^{\mathsf{ma\text{-}m\text{-}h}}(\mathcal{A}, \kappa)]|.$$

Π provides mutual authentication between $\mathcal{M}$ and $\mathcal{H}$ *if this probability is negligible in κ.*

Our next Definition 3 aims to define similar goals with respect to the instances of $\mathcal{F}$ and $\mathcal{H}$: in particular the mutual authentication requirement (conditions 1 and 2), and the requirement related to the agreement on the tunnel key K_t (condition 3). Note that the mutual authentication between $\mathcal{F}$ and $\mathcal{H}$ does not depend on the honesty of $\mathcal{M}$ that can be corrupted by $\mathcal{A}$. Similar to the previous definition $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ are treated as honest with respect to the agreement on K_t.

Definition 3 (MA between $\mathcal{F}$ and $\mathcal{H}$). *Given a correct protocol Π by $\mathsf{Game}_{\Pi}^{\mathsf{ma\text{-}f\text{-}h}}(\mathcal{A}, \kappa)$ we denote the interaction between the instances of $\mathcal{M}$, $\mathcal{F}$ and $\mathcal{H}$ with a PPT adversary $\mathcal{A}$ that is allowed to query* Invoke, Send, Corrupt, RevealKey, *and* RevealTunnelKey. *$\mathcal{A}$* wins *if at some point during the interaction:*

(1) an uncorrupted instance of $\mathcal{F}$ accepts but there is ***no*** *uncorrupted partnered instance of $\mathcal{H}$,* ***or***
(2) an uncorrupted instance of $\mathcal{H}$ accepts but there is ***no*** *uncorrupted partnered instance of $\mathcal{F}$,* ***or***
(3) $\mathcal{M}$ is uncorrupted and uncorrupted partnered instances of $\mathcal{F}$ and $\mathcal{H}$ accept without holding the same session tunnel key K_t.

The maximum probability of this event over all adversaries (running in time κ) is denoted

$$\mathsf{Succ}^{\mathsf{ma\text{-}f\text{-}h}}_{\Pi}(\mathcal{A},\kappa) = \max_{\mathcal{A}} |\Pr[\mathcal{A} \text{ wins in } \mathsf{Game}^{\mathsf{ma\text{-}f\text{-}h}}_{\Pi}(\mathcal{A},\kappa)]|.$$

Π provides mutual authentication between $\mathcal{F}$ and $\mathcal{H}$ *if this probability is negligible in κ.*

Here, we provide some observations concerning Definitions 2 and 3 with respect to the authorization issues. Recall, that according to our model instances of protocol participants are seen as partnered if they hold the same session ids. This implies that any protocol Π which satisfies both of the above defined mutual authentication requirements ensures that if an uncorrupted instance of $\mathcal{H}$ accepts then there are uncorrupted instances of $\mathcal{M}$ and $\mathcal{F}$ that are also partnered. That is the decision of $\mathcal{H}$ to accept in some session of Π implies that $\mathcal{H}$ treats $\mathcal{M}$ and $\mathcal{F}$ as authorized participants of a WRT session. Since our model does not consider malicious $\mathcal{M}$, $\mathcal{F}$, or $\mathcal{H}$ aiming to disrupt the agreement on K_t the above mentioned "transitive" partnering between the instances of $\mathcal{M}$ and $\mathcal{F}$ ensures that all partnered instances that accept in some protocol session hold the same session tunnel key K_t.

In the following we focus on the secrecy of $K_{\mathcal{M},\mathcal{H}}$ and K_t. For this, we make use of the classical notion of authenticated key exchange (AKE) security (cf. [13, 21, 22]), adopted to the setting of our model. The basic idea of AKE-security in WRT is to model the indistinguishability of $K_{\mathcal{M},\mathcal{H}}$ and K_t computed in some *test* session from some randomly chosen values by any *outsider* adversary. The significant difference is that for $K_{\mathcal{M},\mathcal{H}}$ a possibly malicious foreign network $\mathcal{F}$ should be also treated as such adversary.

In order to model the AKE-security of the end-to-end key $K_{\mathcal{M},\mathcal{H}}$ we first specify the auxiliary notion of *e2e-freshness* for the instances of $\mathcal{M}$ and $\mathcal{H}$, which defines the conditions under which $\mathcal{A}$ can be treated as an outsider with respect to the test session for which it has to distinguish $K_{\mathcal{M},\mathcal{H}}$. In particular, these conditions prevent active participation of $\mathcal{A}$ on behalf of either $\mathcal{M}$ or $\mathcal{H}$ by restricting $\mathcal{A}$ from respective corruptions, and capture known-key attacks allowing $\mathcal{A}$ to reveal end-to-end keys computed in sessions that are different from the test session. Moreover, $\mathcal{A}$ is not restricted from revealing K_t. That is the knowledge of K_t (e.g., by $\mathcal{F}$) should not compromise the end-to-end communication security between $\mathcal{M}$ and $\mathcal{H}$.

Definition 4 (e2e-Freshness). *In the execution of Π an instance $[P,s]$ with $P \in \{\mathcal{M},\mathcal{H}\}$ is* e2e-fresh *if* ***none*** *of the following holds:*

- *$\mathcal{A}$ asks* $\mathsf{Corrupt}(P)$*;*
- *Case $P = \mathcal{M}$: $\mathcal{A}$ asks* $\mathsf{RevealKey}(\mathcal{M},s)$ *after $[\mathcal{M},s]$ has accepted or* $\mathsf{RevealKey}(\mathcal{H},t)$ *after $[\mathcal{H},t]$ has accepted and $[\mathcal{M},s]$ and $[\mathcal{H},t]$ are partnered;*
- *Case $P = \mathcal{H}$: $\mathcal{A}$ asks* $\mathsf{RevealKey}(\mathcal{H},s)$ *after $[\mathcal{H},s]$ has accepted or* $\mathsf{RevealKey}(\mathcal{M},t)$ *after $[\mathcal{M},t]$ has accepted and $[\mathcal{H},s]$ and $[\mathcal{M},t]$ are partnered.*

In order to model the AKE-security of the tunnel key K_t we specify the auxiliary notion of *t-freshness*, this time for the instances of $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$, which defines the conditions under which $\mathcal{A}$ can be treated as an outsider with respect to the test session for which it has to distinguish K_t. These conditions are widely similar to those defined for the e2e-freshness except that $\mathcal{A}$ is now allowed to reveal tunnel keys computed in sessions that are different from the test session, and is not restricted from revealing $K_{\mathcal{M},\mathcal{H}}$.

Definition 5 (t-Freshness). *In the execution of Π an instance $[P,s]$ with $P \in \{\mathcal{M},\mathcal{F},\mathcal{H}\}$ is* t-fresh *if* ***none*** *of the following holds:*

- *$\mathcal{A}$ asks* $\mathsf{Corrupt}(P)$*;*
- *Case $P = \mathcal{M}$: $\mathcal{A}$ asks* $\mathsf{RevealTunnelKey}(\mathcal{M},s)$ *after $[\mathcal{M},s]$ has accepted or* $\mathsf{RevealTunnelKey}(P',t)$ *for $P' \in \{\mathcal{H},\mathcal{F}\}$ after $[P',t]$ has accepted and $[\mathcal{M},s]$ and $[P',t]$ are partnered;*

- *Case $P = \mathcal{F}$: $\mathcal{A}$ asks* $\mathsf{RevealTunnelKey}(\mathcal{F}, s)$ *after* $[\mathcal{F}, s]$ *has accepted or* $\mathsf{RevealTunnelKey}(P', t)$ *for* $P' \in \{\mathcal{M}, \mathcal{H}\}$ *after* $[P', t]$ *has accepted and* $[\mathcal{F}, s]$ *and* $[P', t]$ *are partnered;*
- *Case $P = \mathcal{H}$: $\mathcal{A}$ asks* $\mathsf{RevealTunnelKey}(\mathcal{H}, s)$ *after* $[\mathcal{H}, s]$ *has accepted or* $\mathsf{RevealTunnelKey}(P', t)$ *for* $P' \in \{\mathcal{M}, \mathcal{F}\}$ *after* $[P', t]$ *has accepted and* $[\mathcal{H}, s]$ *and* $[P', t]$ *are partnered.*

Further, we introduce two additional queries $\mathsf{TestKey}(P, s)$ and $\mathsf{TestTunnelKey}(P, s)$. The query $\mathsf{TestKey}(P, s)$ can be asked to an instance of $P \in \{\mathcal{M}, \mathcal{H}\}$ and is answered only if the instance $[P, s]$ has already accepted. The answer of this query is based on some secret bit $b \in \{0, 1\}$ chosen in advance. In response to $\mathsf{TestKey}(P, s)$ $\mathcal{A}$ is given either $K_{\mathcal{M},\mathcal{H}}$ (if $b = 1$) or a randomly chosen value from $\{0, 1\}^\kappa$ (if $b = 0$). The only difference between $\mathsf{TestKey}(P, s)$ and $\mathsf{TestTunnelKey}(P, s)$ is that for the latter we assume $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$ and the key which is returned in the case that bit $b = 1$ is the tunnel key K_t.

Now we are ready to formally define AKE-security for the end-to-end and tunnel keys computed in Π. Using the auxiliary definitions of e2e- and t-freshness we can provide one definition for both goals.

Definition 6 (E2E / Tunnel AKE). *Given a correct protocol Π, a uniformly chosen bit b, a type of AKE-security $\alpha \in \{\mathsf{ake\text{-}e2e}, \mathsf{ake\text{-}t}\}$, and a PPT adversary $\mathcal{A}$ with access to the queries* Invoke, Send, Corrupt, RevealKey, RevealTunnelKey, *and* TestKey *(if $\alpha = \mathsf{ake\text{-}e2e}$) or* TestTunnelKey *(if $\alpha = \mathsf{ake\text{-}t}$), by $\mathsf{Game}_{\Pi}^{\alpha,b}(\mathcal{A}, \kappa)$ we denote the following interaction between the instances of $\mathcal{M}$, $\mathcal{F}$ and $\mathcal{H}$ with $\mathcal{A}$:*

- *$\mathcal{A}$ interacts with instances via queries;*
- *Case $\alpha = \mathsf{ake\text{-}e2e}$: at some point $\mathcal{A}$ asks a* TestKey *query to an instance $[P, s]$ which has accepted and is **e2e-fresh** (and remains such by the end of the interaction);*
  Case $\alpha = \mathsf{ake\text{-}t}$: at some point $\mathcal{A}$ asks a TestTunnelKey *query to an instance $[P, s]$ which has accepted and is **t-fresh** (and remains such by the end of the interaction);*
- *$\mathcal{A}$ continues interacting with instances and when $\mathcal{A}$ terminates, it outputs a bit, which is then set as the output of the interaction.*

$\mathcal{A}$ wins *if the output of $\mathsf{Game}_{\Pi}^{\alpha,b}(\mathcal{A}, \kappa)$ is identical to b. The maximum probability of the adversarial advantage over the random guess of b, over all adversaries (running in time κ) is denoted*

$$\mathsf{Adv}_{\Pi}^{\alpha}(\mathcal{A}, \kappa) = \max_{\mathcal{A}} |2 \Pr[\mathsf{Game}_{\Pi}^{\alpha,b}(\mathcal{A}, \kappa) = b] - 1|.$$

If this advantage is negligible in κ and $\alpha = \mathsf{ake\text{-}e2e}$ (or $\alpha = \mathsf{ake\text{-}t}$) then Π provides end-to-end *(or* tunnel*)* AKE-security.

Remark 1. Our notions of freshness restrict $\mathcal{A}$ from corruptions of $\mathcal{M}$ and $\mathcal{H}$ (in e2e-freshness) and from corruptions of $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ (in t-freshness). This implies that our definitions of AKE-security do not consider *forward secrecy*. This is done on purpose, since the basic version of our protocol specified and formally analyzed in Section 5 does not provide this property, mainly for the reasons of efficiency, resulting in a possible use of the protocol for the performance-constrained mobile devices. Nevertheless, in Section 6.1 we show, how forward secrecy for $K_{\mathcal{M},\mathcal{H}}$ and K_t can be easily achieved using the classical Diffie-Hellman technique [25].

5 Authentication and Key Establishment Protocol for WRT

In the following we introduce `AWRT` — our protocol for authentication and key establishment for WRT between $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$. This section describes its basic version, analyzes security and evaluates performance. Optional security extensions and the discussion on the use of `AWRT` in commercial networks are postponed to Section 6.

5.1 Building Blocks

`AWRT` uses several (well-known) cryptographic primitives:

- A *pseudo-random function* $\texttt{PRF} : \{0,1\}^\kappa \times \{0,1\}^* \rightarrow \{0,1\}^*$ which is used for the purpose of key derivation and can be realized using block-ciphers or keyed one-way hash functions. By $\mathsf{Adv}^{\texttt{prf}}_{\texttt{PRF}}(\kappa)$ we denote the maximum advantage over all PPT adversaries (running within time κ) in distinguishing the outputs of PRF from those of a random function better than by a random guess.
- An *asymmetric encryption scheme* satisfying the property of indistinguishability under (adaptive) chosen-ciphertext attacks (IND-CCA2) [48] whose encryption and decryption operations are denoted $\mathcal{E}$ and $\mathcal{D}$, respectively. By $\mathsf{Adv}^{\texttt{ind-cca2}}_{(\mathcal{E},\mathcal{D})}(\kappa)$ we denote the maximum advantage over all PPT adversaries (running within time κ) in breaking the IND-CCA2 property of $(\mathcal{E},\mathcal{D})$ better than by a random guess; The property of IND-CCA2 security is for example preserved in the several encryption schemes including RSA-OAEP [14,31], Cramer-Shoup [24], and DHIES [4].
- A *digital signature scheme* which provides existential unforgeability under chosen message attacks (EUF-CMA) whose signing and verification operations are denoted Sig and Ver, respectively. By $\mathsf{Succ}^{\texttt{euf-cma}}_{(Sig,Ver)}(\kappa)$ we denote the maximum success probability over all PPT adversaries (running within time κ) given access to the signing oracle in finding a forgery; Examples of such schemes include DSS [47] and PSS [15], though schemes that provide a stronger version of EUF-CMA (cf. [17,34]) can be applied as well.
- A *message authentication code* function MAC that satisfies *weak unforgeability against chosen message attacks* (WUF-CMA) [12], e.g., the popular function HMAC [10,11] can be used for this purpose. By $\mathsf{Succ}^{\texttt{wuf-cma}}_{\texttt{MAC}}(\kappa)$ we denote the maximum success probability over all PPT adversaries (running within time κ) given access to the MAC oracle in finding a MAC forgery.

5.2 Initialization

We assume that prior to the execution of AWRT the involved parties are in possession of the following long lived keys: $LL_\mathcal{F}$ consists of a private/public signature/verification key pair $(sk_\mathcal{F}, vk_\mathcal{F})$ and a decryption/encryption key pair $(dk_\mathcal{F}, ek_\mathcal{F})$; $LL_\mathcal{H}$ consists of a private/public signature/verification key pair $(sk_\mathcal{H}, vk_\mathcal{H})$ and a pair $(\mathcal{M}, (k_\mathcal{M}, \alpha_\mathcal{M}))$ where $(k_\mathcal{M}, \alpha_\mathcal{M})$ is a high-entropy secret key consisting of a PRF key $k_\mathcal{M}$ and a MAC key $\alpha_\mathcal{M}$ shared with the hosted $\mathcal{M}$; consequently $LL_\mathcal{M}$ consists of $(k_\mathcal{M}, \alpha_\mathcal{M})$. Note that in practice it is sufficient for $\mathcal{H}$ and $\mathcal{M}$ to share $k_\mathcal{M}$ and derive the corresponding MAC key $\alpha_\mathcal{M}$ as $\texttt{PRF}_{k_\mathcal{M}}(l)$ for some publicly fixed label l.

Further we assume that the public keys of networks are known amongst them in advance (implied by their contract). They can also be handled via self-signed or classical PKI certificates. $\mathcal{F}$ and $\mathcal{H}$ can choose their long-lived keys independently and $\mathcal{H}$ can choose $k_\mathcal{M}$ for each hosted $\mathcal{M}$ according to their assumed trust relationship.

Remark 2. Since the networks $\mathcal{F}$ and $\mathcal{H}$ can usually swap their roles (e.g., bidirectional roaming contracts between $\mathcal{H}$ and $\mathcal{F}$, or a roaming contract among some set of networks from which $\mathcal{F}$ and $\mathcal{H}$ can be seen as any two chosen networks), $LL_\mathcal{H}$ may include some $(dk_\mathcal{H}, ek_\mathcal{H})$ and $LL_\mathcal{F}$ may also include $(\mathcal{M}, (k_\mathcal{M}, \alpha_\mathcal{M}))$ for each mobile device $\mathcal{M}$ hosted by $\mathcal{F}$.

Remark 3. For the purpose of efficiency and scalability it might be desirable for the networks to use a single private/public key pair to decrypt and to sign. There exist several secure schemes that support both operations with the same public key pair (and could be deployed in our protocol), e.g., [23]. Nevertheless, the use of the same key pair in two different operations contradicts to the general principles of a secure protocol design and is, therefore, not recommended.

5.3 Protocol Execution

In the following we provide some explanations on the techniques used in our AWRT protocol and their relationship to the security goals. The actual specification is illustrated in Figure 1.

First, we mention an optional *time-stamp* T which can be chosen by $\mathcal{F}$ and sent to $\mathcal{H}$ in order to address possible accounting issues in commercial roaming scenarios where $\mathcal{F}$ should be reimbursed for the provided tunnel connection; more discussion on this can be found in Section 6.4.

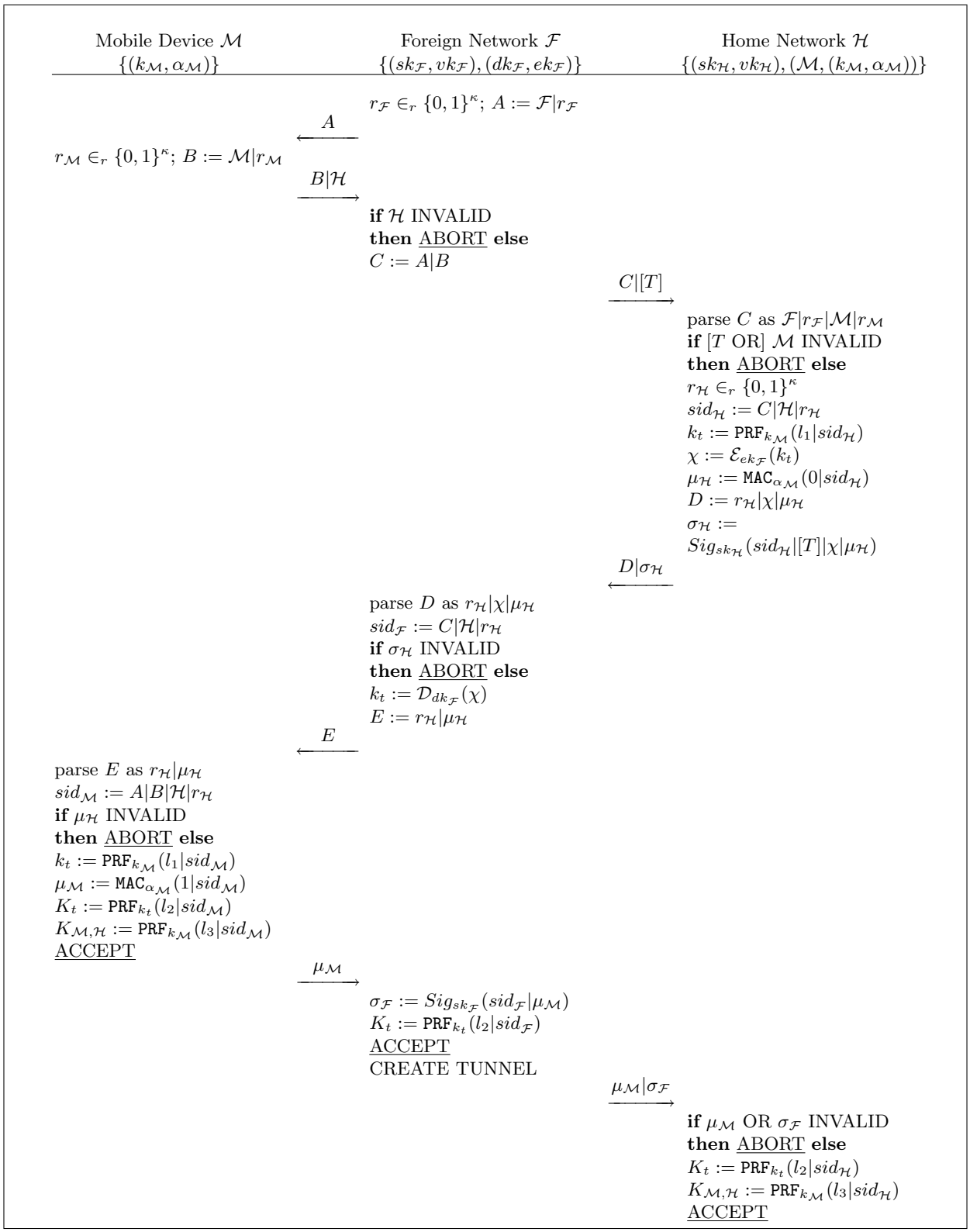

Fig. 1. Illustration of the basic version of AWRT between the participating mobile device $\mathcal{M}$, foreign network $\mathcal{F}$, and home network $\mathcal{H}$. At the end of the protocol: $\mathcal{M}$ and $\mathcal{H}$ hold the end-to-end key $K_{\mathcal{M},\mathcal{H}}$; $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ hold the tunnel key K_t, so that the tunnel between $\mathcal{M}$ and $\mathcal{H}$ can be established by $\mathcal{F}$.

AWRT uses publicly known distinct *labels* l_i, $i = 1,\ldots,3$, which are fixed in advance and used as input to PRF to derive various secret keys at different protocol stages.

In AWRT each party $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$ computes own session id sid_P as a concatenated bit string $\mathcal{F}|r_{\mathcal{F}}|\mathcal{M}|r_{\mathcal{M}}|\mathcal{H}|r_{\mathcal{H}}$ where r_P denotes a *random nonce* chosen by P.

The end-to-end key $K_{\mathcal{M},\mathcal{H}}$ is derived by $\mathcal{M}$ and $\mathcal{H}$ as the output of PRF (with label l_3 and the session id) using the shared secret key $k_{\mathcal{M}}$. Obviously, the equality $sid_{\mathcal{M}} = sid_{\mathcal{H}}$ is necessary for $\mathcal{M}$ and $\mathcal{H}$ to compute the same value of $K_{\mathcal{M},\mathcal{H}}$. The assurance of this equality is given to $\mathcal{M}$ through the MAC value $\mu_{\mathcal{H}}$ since the corresponding MAC key $\alpha_{\mathcal{M}}$ is known only to $\mathcal{M}$ and $\mathcal{H}$. Similarly, $\mathcal{H}$ gains this assurance from the MAC value $\mu_{\mathcal{M}}$. Observe that bits 0 and 1 are used as additional inputs for the computation of $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$, respectively, to break the "symmetry" and guarantee that $\mu_{\mathcal{H}} \neq \mu_{\mathcal{M}}$. Due to the construction of session ids from fresh nonces (seen as challenges) the successful verification of these MAC values provides also the mutual authentication between $\mathcal{M}$ and $\mathcal{H}$.

The mutual authentication between $\mathcal{F}$ and $\mathcal{H}$ is achieved via digital signatures $\sigma_{\mathcal{H}}$ and $\sigma_{\mathcal{F}}$ as the signed messages include $sid_{\mathcal{H}}$ and $sid_{\mathcal{F}}$, respectively.

The tunnel key K_t is derived by $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ as the output of PRF (with label l_2 and the session id) using the *pre-tunnel key* k_t which is computed by the parties in two different ways: $\mathcal{M}$ and $\mathcal{H}$ derive k_t from PRF (with label l_1 and the session id) using the shared secret $k_{\mathcal{M}}$, whereas $\mathcal{F}$ obtains k_t via decryption from the cipher-text χ received from $\mathcal{H}$. The protection of χ by $\sigma_{\mathcal{H}}$ ensures $\mathcal{F}$ that $\mathcal{H}$ holds the same value for k_t. Since $\mathcal{F}$ produces $\sigma_{\mathcal{F}}$ subsequently to the verification of $\sigma_{\mathcal{H}}$ the validity of $\sigma_{\mathcal{F}}$ ensures $\mathcal{H}$ that the cipher-text χ was delivered to $\mathcal{F}$ without modification allowing $\mathcal{F}$ to decrypt the same value for k_t. Note that the mutual authentication between $\mathcal{M}$ and $\mathcal{H}$ also implies that the partnered instances of these two parties derive the same value for k_t too.

5.4 Security Analysis

In this section we prove that AWRT satisfies the previously defined security goals. In all our theorems by q we denote the total number of the invoked protocol sessions in the corresponding interactions between the adversary $\mathcal{A}$ and the protocol participants. We start by proving the defined mutual authentication goals.

Theorem 1 (MA between $\mathcal{M}$ and $\mathcal{H}$). *Given a WUF-CMA secure* MAC *the basic version of* AWRT *described in Figure 1 provides mutual authentication between the participating mobile device and its home network in the sense of Definition 2, and*

$$\mathsf{Succ}^{\mathsf{ma\text{-}m\text{-}h}}_{\mathtt{AWRT}}(\kappa) \leq \frac{2q^2}{2^\kappa} + 2\mathsf{Succ}^{\mathtt{wuf\text{-}cma}}_{\mathtt{MAC}}(\kappa).$$

Proof. (Sketch) In our proofs we apply the meanwhile classical proving technique from [56]. Here we construct a *sequence of games* $\mathbf{G}_i$, $i = 0,\ldots,2$ and denote by $\mathsf{Win}^{\mathsf{ma\text{-}m\text{-}h}}_i$ the event that an adversary $\mathcal{A}$ breaks the mutual authentication between $\mathcal{M}$ and $\mathcal{H}$ in game $\mathbf{G}_i$, i.e., wins in the corresponding interaction as described in Definition 2. Note that $\mathcal{A}$ is allowed to corrupt $\mathcal{F}$ for the winning conditions (1) – (3), but not for (4).

Game $\mathbf{G}_0$. [*Real protocol*] This is the real $\mathsf{Game}^{\mathsf{ma\text{-}m\text{-}h}}_{\mathtt{AWRT}}(\kappa)$ played between a simulator Δ and a PPT adversary $\mathcal{A}$. Δ simulates the actions of the participating $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ according to the protocol specification and answers all queries of $\mathcal{A}$.

Game $\mathbf{G}_1$. [*Collisions for chosen nonces $r_{\mathcal{M}}$ and $r_{\mathcal{H}}$*] In this game the simulation aborts if during the interaction the simulator chooses the same random nonce $r_{\mathcal{M}}$ resp. $r_{\mathcal{H}}$ on behalf of $\mathcal{M}$ resp. $\mathcal{H}$ in two different protocol sessions. Considering the collision probability for the same nonce to be chosen twice we obtain

$$|\Pr[\mathsf{Win}^{\mathsf{ma\text{-}m\text{-}h}}_1] - \Pr[\mathsf{Win}^{\mathsf{ma\text{-}m\text{-}h}}_0]| \leq \frac{2q^2}{2^\kappa}. \quad (1)$$

Note that since in AWRT a session id of $P \in \{\mathcal{M}, \mathcal{F}, \mathcal{H}\}$ is computed as concatenated string $\mathcal{F}|r_{\mathcal{F}}|\mathcal{M}|r_{\mathcal{M}}|\mathcal{H}|r_{\mathcal{H}}$ this game rules out the occurrence of the same session ids computed by the instances of $\mathcal{M}$ and $\mathcal{H}$ in

two different sessions, regardless of the chosen $r_{\mathcal{F}}$. In particular, this game implies that $sid_{\mathcal{M}}$ and $sid_{\mathcal{H}}$ remain unique for each invoked session.

Game $\mathbf{G}_2$. [*MAC forgeries for $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$*] This game is identical to Game $\mathbf{G}_1$ with the only exception that Δ fails if $\mathcal{A}$ asks a Send query to an instance of $\mathcal{M}$ containing a valid $\mathtt{MAC}$ value $\mu_{\mathcal{H}}$ not previously output by an instance of $\mathcal{H}$ or a Send query to an instance of $\mathcal{H}$ containing a valid $\mu_{\mathcal{M}}$ not previously output by an instance of $\mathcal{M}$.

The probability that the simulation aborts can be upper-bounded through the probability of forging any of the both $\mathtt{MAC}$ values. To see this, consider Δ given access to the $\mathtt{MAC}$ oracle. Δ simulates the execution of $\mathtt{AWRT}$ according to the specification except that it computes $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$ through the corresponding oracle calls. In case that the simulation aborts Δ is in possession of a valid $\mathtt{MAC}$ value (representing either $\mu_{\mathcal{H}}$ or $\mu_{\mathcal{M}}$) which was not obtained through any previous oracle call. Hence, Δ can easily output it as a forgery. This implies

$$|\Pr[\mathsf{Win}_2^{\mathsf{ma\text{-}m\text{-}h}}] - \Pr[\mathsf{Win}_1^{\mathsf{ma\text{-}m\text{-}h}}]| \leq 2\mathsf{Succ}_{\mathtt{MAC}}^{\mathtt{wuf\text{-}cma}}(\kappa). \tag{2}$$

Having eliminated possible forgeries for $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$ we observe that since these $\mathtt{MAC}$ values are computed over the session ids $sid_{\mathcal{H}}$ and $sid_{\mathcal{M}}$, respectively, that according to the previous game are unique for each new session, this game rules out any successful replay attacks using $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$. Therefore, any successful verification of $\mu_{\mathcal{H}}$ by an instance of $\mathcal{M}$ and of $\mu_{\mathcal{M}}$ by an instance of $\mathcal{H}$ implies that there are two instances of $\mathcal{M}$ and $\mathcal{H}$ that hold the same session ids, and are, therefore partnered. Since verification of $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$ is the necessary requirement for the acceptance of the instances of $\mathcal{M}$ and $\mathcal{H}$ in $\mathtt{AWRT}$ we follow that this game ensures mutual authentication between $\mathcal{M}$ and $\mathcal{H}$ and excludes attacks by which $\mathcal{A}$ can win based on conditions (1) and (2).

Further, we focus on the attacks based on conditions (3) and (4). Since the session end-to-end key $K_{\mathcal{M},\mathcal{H}}$ is derived by the instances of $\mathcal{M}$ and $\mathcal{H}$ in a deterministic way as $\mathtt{PRF}_{k_{\mathcal{M}}}(l_3, sid_{\mathcal{M}})$ and $\mathtt{PRF}_{k_{\mathcal{M}}}(l_3, sid_{\mathcal{H}})$, respectively, it follows that if any two partnered instances of $\mathcal{M}$ and $\mathcal{H}$ accept then they hold identical values for $K_{\mathcal{M},\mathcal{H}}$, i.e. the probability of $\mathcal{A}$ to win in this game through condition 3 is 0. Finally, we observe that if any two partnered instances of $\mathcal{M}$ and $\mathcal{H}$ accept then they hold the tunnel key K_t computed in a deterministic way as $\mathtt{PRF}_{k_t}(l_2, sid_{\mathcal{M}})$ and $\mathtt{PRF}_{k_t}(l_2, sid_{\mathcal{H}})$, respectively. Hence, the probability that $\mathcal{A}$ wins in this game through condition 4, is upper-bounded by the probability that these partnered instances have computed different values for the pre-tunnel key k_t. Since k_t is derived by the partnered instances of $\mathcal{M}$ and $\mathcal{H}$ in a deterministic way as $\mathtt{PRF}_{k_{\mathcal{M}}}(l_1, sid_{\mathcal{M}})$ and $\mathtt{PRF}_{k_{\mathcal{M}}}(l_1, sid_{\mathcal{H}})$, respectively, thus using the same values for $k_{\mathcal{M}}$ and the session ids, it follows that the probability of $\mathcal{A}$ winning in this game through condition 4 is also 0. Summing up the discussed probabilities of $\mathcal{A}$ to win in this game based on conditions 3 and 4 we obtain

$$\Pr[\mathsf{Win}_2^{\mathsf{ma\text{-}m\text{-}h}}] = 0. \tag{3}$$

Combining the previous equations, we conclude the proof. □

Theorem 2 (MA between $\mathcal{F}$ and $\mathcal{H}$). *Given a EUF-CMA secure (Sig, Ver) the basic version of* $\mathtt{AWRT}$ *described in Figure 1 provides mutual authentication between the participating foreign and home networks in the sense of Definition 3, and*

$$\mathsf{Succ}_{\mathtt{AWRT}}^{\mathsf{ma\text{-}f\text{-}h}}(\kappa) \leq \frac{2q^2}{2^\kappa} + 2\mathsf{Succ}_{(Sig,Ver)}^{\mathtt{euf\text{-}cma}}(\kappa).$$

Proof. (Sketch) Similar to the previous proof we construct a sequence of games $\mathbf{G}_i$, $i = 0, \ldots, 2$ and denote by $\mathsf{Win}_i^{\mathsf{ma\text{-}f\text{-}h}}$ the event that $\mathcal{A}$ breaks the mutual authentication between $\mathcal{F}$ and $\mathcal{H}$ in game $\mathbf{G}_i$, i.e., wins in the corresponding interaction as described in Definition 3. Note that $\mathcal{A}$ is allowed to corrupt $\mathcal{F}$ for the winning conditions (1) – (2), but not for (3).

Game $\mathbf{G}_0$. [*Real protocol*] This is the real $\mathsf{Game}_{\mathtt{AWRT}}^{\mathsf{ma\text{-}f\text{-}h}}(\kappa)$ played between a simulator Δ and a PPT adversary $\mathcal{A}$. Δ simulates the actions of the participating $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ according to the protocol specification and answers all queries of $\mathcal{A}$.

Game $\mathbf{G}_1$. [*Collisions for nonces $r_{\mathcal{F}}$ and $r_{\mathcal{H}}$*] Similar to the proof of Theorem 1 we abort the simulation in this game if during the interaction Δ chooses the same random nonce $r_{\mathcal{F}}$ resp. $r_{\mathcal{H}}$ on behalf of

$\mathcal{F}$ resp. $\mathcal{H}$ in two different protocol sessions. Thus,

$$|\Pr[\mathsf{Win}_1^{\mathsf{ma\text{-}f\text{-}h}}] - \Pr[\mathsf{Win}_0^{\mathsf{ma\text{-}f\text{-}h}}]| \leq \frac{2q^2}{2^\kappa}. \tag{4}$$

Obviously, this game implies that $sid_{\mathcal{F}}$ and $sid_{\mathcal{H}}$ remain unique for each invoked session, regardless of the chosen $r_{\mathcal{M}}$.

Game $\mathbf{G}_2$. [*Signature forgeries for $\sigma_{\mathcal{H}}$ and $\sigma_{\mathcal{F}}$*] This game is identical to Game $\mathbf{G}_1$ with the only exception that Δ fails if $\mathcal{A}$ asks a Send query to an instance of $\mathcal{F}$ containing a valid signature $\sigma_{\mathcal{H}}$ not previously output by an instance of $\mathcal{H}$ or a Send query to an instance of $\mathcal{H}$ containing a valid signature $\sigma_{\mathcal{F}}$ not previously output by an instance of $\mathcal{F}$.

Assume that Δ simulates the protocol execution according to the specification except that it is given access to the signing oracles which it queries in order to obtain the corresponding signatures $\sigma_{\mathcal{H}}$ and $\sigma_{\mathcal{F}}$ on behalf of $\mathcal{H}$ and $\mathcal{F}$, respectively. In case that the simulation aborts Δ is in possession of a valid signature (representing either $\sigma_{\mathcal{H}}$ or $\sigma_{\mathcal{F}}$) which was not obtained through any previous oracle call, and can, therefore, be returned by Δ as a corresponding forgery. Hence,

$$|\Pr[\mathsf{Win}_2^{\mathsf{ma\text{-}f\text{-}h}}] - \Pr[\mathsf{Win}_1^{\mathsf{ma\text{-}f\text{-}h}}]| \leq 2\mathsf{Succ}^{\mathsf{euf\text{-}cma}}_{(Sig,Ver)}(\kappa). \tag{5}$$

Since each of these signatures is computed over the corresponding session id $sid_{\mathcal{F}}$ or $sid_{\mathcal{H}}$ (amongst other inputs), respectively, that according to the previous game are unique for each new session, this game rules out any successful replay attacks. Since verification of $\sigma_{\mathcal{H}}$ by an instance of $\mathcal{F}$ and verification of $\sigma_{\mathcal{F}}$ by an instance of $\mathcal{H}$ is the necessary requirement for the acceptance in `AWRT` we follow that upon acceptance of any instance of $\mathcal{F}$ (or $\mathcal{H}$) there is a corresponding partnered instance of $\mathcal{H}$ (or $\mathcal{F}$). This implies the mutual authentication between $\mathcal{F}$ and $\mathcal{H}$ and excludes attacks by which $\mathcal{A}$ can win based on conditions (1) and (2).

Further, we focus on the attacks based on condition (3). Observe, that if two partnered instances of $\mathcal{F}$ and $\mathcal{H}$ accept then the exchanged signatures $sid_{\mathcal{F}}$ or $sid_{\mathcal{H}}$ between these instances were valid (and also not replayed). This implies that the integrity of the cipher-text χ transmitted to the instance of $\mathcal{F}$ was preserved. Since decryption $\mathcal{D}$ is a deterministic operation we follow that upon acceptance both partnered instances of $\mathcal{F}$ and $\mathcal{H}$ hold the same value for the pre-tunnel key k_t, which is used by the instances to derive the tunnel key K_t prior to acceptance. Hence, the probability that partnered instances of $\mathcal{F}$ and $\mathcal{H}$ accept with two different tunnel keys in this game is 0, i.e.

$$\Pr[\mathsf{Win}_2^{\mathsf{ma\text{-}f\text{-}h}}] = 0. \tag{6}$$

Combining the previous equations, we conclude the proof. □

Now we focus on the AKE-security of the established session end-to-end and tunnel keys.

Theorem 3 (End-to-End AKE). *Given a WUF-CMA secure* `MAC` *and a pseudo-random* `PRF` *the basic version of* `AWRT` *described in Figure 1 provides end-to-end AKE-security in the sense of Definition 6, and*

$$\mathsf{Adv}^{\mathsf{ake\text{-}e2e}}_{\mathtt{AWRT}}(\kappa) \leq \frac{4q^2}{2^\kappa} + 4\mathsf{Succ}^{\mathsf{wuf\text{-}cma}}_{\mathtt{MAC}}(\kappa) + 4q\mathsf{Adv}^{\mathsf{prf}}_{\mathtt{PRF}}(\kappa).$$

Proof. (Sketch) As in the previous proofs we construct a sequence of games $\mathbf{G}_i$, $i = 0, \ldots, 4$ and denote by $\mathsf{Win}_i^{\mathsf{ake\text{-}e2e}}$ the event that $\mathcal{A}$ breaks the end-to-end AKE-security of `AWRT` in game $\mathbf{G}_i$, i.e., wins in the corresponding interaction as described in Definition 6 (for the case $\alpha = \mathsf{ake\text{-}e2e}$).

Game $\mathbf{G}_0$. [*Real protocol*] This is the real $\mathsf{Game}^{\mathsf{ake\text{-}e2e}}_{\mathtt{AWRT}}(\kappa)$ played between a simulator Δ and a PPT adversary $\mathcal{A}$. Δ simulates the actions of the participating $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ according to the protocol specification and answers all queries of $\mathcal{A}$. Recall, that the $\mathsf{TestKey}$ query is asked by $\mathcal{A}$ to an e2e-fresh instance of either $\mathcal{M}$ or $\mathcal{H}$ which has previously accepted. In order to prevent $\mathcal{A}$ from active participation on behalf of either $\mathcal{M}$ or $\mathcal{H}$ we first exclude possible impersonation attacks against any of these parties. For this we utilize games $\mathbf{G}_1$ and $\mathbf{G}_2$ from the proof of Theorem 1.

Game $\mathbf{G}_1$. [*Collisions for nonces $r_{\mathcal{M}}$ and $r_{\mathcal{H}}$*] The simulation in this game aborts (and the output bit of the interaction is set at random) if the same random nonce $r_{\mathcal{M}}$ (or $r_{\mathcal{H}}$) is chosen by Δ on behalf of $\mathcal{M}$ (or $\mathcal{H}$) in two different protocol sessions, implying

$$|\Pr[\mathsf{Win}_1^{\mathsf{ake\text{-}e2e}}] - \Pr[\mathsf{Win}_0^{\mathsf{ake\text{-}e2e}}]| \leq \frac{2q^2}{2^\kappa}. \tag{7}$$

Game $\mathbf{G}_2$. [*MAC forgeries for $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$*] The simulation in this game aborts (and the output bit of the interaction is set at random) if $\mathcal{A}$ asks as part of its Send query to $\mathcal{M}$ (or to $\mathcal{H}$) a valid $\mathtt{MAC}$ value $\mu_{\mathcal{H}}$ (or $\mu_{\mathcal{M}}$) which was not previously output by an instance of $\mathcal{H}$ (or $\mathcal{M}$), so that

$$|\Pr[\mathsf{Win}_2^{\mathsf{ake\text{-}e2e}}] - \Pr[\mathsf{Win}_1^{\mathsf{ake\text{-}e2e}}]| \leq 2\mathsf{Succ}_{\mathtt{MAC}}^{\mathtt{wuf\text{-}cma}}(\kappa). \tag{8}$$

Having eliminated possible forgeries and replay attacks with respect to $\mu_{\mathcal{H}}$ and $\mu_{\mathcal{M}}$ we observe that any e2e-fresh instance of $\mathcal{M}$ (or $\mathcal{H}$) which accepts has a partnered instance of $\mathcal{H}$ (or $\mathcal{M}$) which is also e2e-fresh.

Game $\mathbf{G}_3$. [*Pseudo-randomness of k_t*] This game is identical to Game $\mathbf{G}_2$ except that Δ in each session chooses the pre-tunnel k_t at random on behalf of an instance of $\mathcal{H}$ instead of deriving it via $\mathtt{PRF}$ and uses the same value in the simulation of the corresponding partnered instance of $\mathcal{M}$ (to preserve consistency). Following the classical reductionist argument we obtain

$$|\Pr[\mathsf{Win}_3^{\mathsf{ake\text{-}e2e}}] - \Pr[\mathsf{Win}_2^{\mathsf{ake\text{-}e2e}}]| \leq q\mathsf{Adv}_{\mathtt{PRF}}^{\mathtt{prf}}(\kappa). \tag{9}$$

Note that this game ensures independence between k_t (used to derive the tunnel key K_t) and the shared key $k_{\mathcal{M}}$ (used to derive the end-to-end key $K_{\mathcal{M},\mathcal{H}}$). We remark that the randomly chosen k_t is also sent encrypted in χ to $\mathcal{F}$ ensuring the consistency between the tunnel key computed by the possibly corrupted instances of $\mathcal{F}$ and uncorrupted instances of $\mathcal{H}$ and $\mathcal{M}$.

Game $\mathbf{G}_4$. [*Pseudo-randomness of $K_{\mathcal{M},\mathcal{H}}$*] This final game is identical to Game $\mathbf{G}_3$ except that Δ in each session chooses $K_{\mathcal{M},\mathcal{H}}$ at random on behalf of an instance of $\mathcal{M}$ instead of deriving it via $\mathtt{PRF}$ and uses the same value in the simulation of the corresponding partnered instance of $\mathcal{H}$ (to preserve consistency), so that

$$|\Pr[\mathsf{Win}_4^{\mathsf{ake\text{-}e2e}}] - \Pr[\mathsf{Win}_3^{\mathsf{ake\text{-}e2e}}]| \leq q\mathsf{Adv}_{\mathtt{PRF}}^{\mathtt{prf}}(\kappa). \tag{10}$$

As a result of this game the answer given to $\mathcal{A}$ in response to its $\mathsf{TestKey}$ query to some e2e-fresh instance of $\mathcal{M}$ or $\mathcal{H}$ is a completely random value, regardless of the chosen bit b. Obviously, the probability of $\mathcal{A}$ to win in this game is given by the probability of a random guess, i.e.

$$\Pr[\mathsf{Win}_4^{\mathsf{ake\text{-}e2e}}] = \frac{1}{2}. \tag{11}$$

The combination of the above equations concludes the proof. □

Theorem 4 (Tunnel AKE). *Given a EUF-CMA secure (Sig, Ver), a IND-CCA2 secure $(\mathcal{E}, \mathcal{D})$ and a pseudo-random* $\mathtt{PRF}$ *the basic version of* $\mathtt{AWRT}$ *described in Figure 1 provides tunnel AKE-security in the sense of Definition 6, and*

$$\mathsf{Adv}_{\mathtt{AWRT}}^{\mathsf{ake\text{-}e2e}}(\kappa) \leq \frac{6q^2}{2^\kappa} + 4\mathsf{Succ}_{\mathtt{MAC}}^{\mathtt{wuf\text{-}cma}}(\kappa) + 4\mathsf{Succ}_{(Sig,Ver)}^{\mathtt{euf\text{-}cma}}(\kappa) + 2q\mathsf{Adv}_{(\mathcal{E},\mathcal{D})}^{\mathtt{ind\text{-}cca2}}(\kappa) + 6q\mathsf{Adv}_{\mathtt{PRF}}^{\mathtt{prf}}(\kappa).$$

Proof. (Sketch) In the following we construct a sequence of games $\mathbf{G}_i$, $i = 0, \ldots, 7$ and denote by $\mathsf{Win}_i^{\mathsf{ake\text{-}t}}$ the event that $\mathcal{A}$ breaks the tunnel AKE-security of $\mathtt{AWRT}$ in game $\mathbf{G}_i$, i.e., wins in the corresponding interaction as described in Definition 6 (for the case $\alpha = \mathsf{ake\text{-}t}$).

Game $\mathbf{G}_0$. [*Real protocol*] This is the real $\mathsf{Game}_{\mathtt{AWRT}}^{\mathsf{ake\text{-}t}}(\kappa)$ played between a simulator Δ and a PPT adversary $\mathcal{A}$. Δ simulates the actions of the participating $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ according to the protocol specification and answers all queries of $\mathcal{A}$. Recall, that the $\mathsf{TestTunnelKey}$ query is asked by $\mathcal{A}$ to a t-fresh instance of either $\mathcal{M}$, $\mathcal{F}$, or $\mathcal{H}$ which has previously accepted. Note in particular, that the notion of

t-freshness excludes any corruptions of $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$. In order to prevent $\mathcal{A}$ from active participation on behalf of either of these parties we first exclude possible impersonation attacks.

Game $\mathbf{G}_1$. [*Collisions for nonces $r_\mathcal{F}$, $r_\mathcal{M}$, and $r_\mathcal{H}$*] The simulation in this game aborts (and the output bit of the interaction is set at random) if the same random nonce $r_\mathcal{F}$, $r_\mathcal{M}$, or $r_\mathcal{H}$ is chosen by Δ on behalf of $\mathcal{F}$, $\mathcal{M}$, or $\mathcal{H}$, respectively, in two different protocol sessions. Thus,

$$|\Pr[\mathsf{Win}_1^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_0^{\mathsf{ake\text{-}t}}]| \leq \frac{3q^2}{2^\kappa}. \tag{12}$$

Game $\mathbf{G}_2$. [*MAC forgeries for $\mu_\mathcal{H}$ and $\mu_\mathcal{M}$*] The simulation in this game aborts (and the output bit of the interaction is set at random) if $\mathcal{A}$ asks as part of its Send query to $\mathcal{M}$ (or to $\mathcal{H}$) a valid $\mathtt{MAC}$ value $\mu_\mathcal{H}$ (or $\mu_\mathcal{M}$) which was not previously output by an instance of $\mathcal{H}$ (or $\mathcal{M}$). Obviously,

$$|\Pr[\mathsf{Win}_2^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_1^{\mathsf{ake\text{-}t}}]| \leq 2\mathsf{Succ}_{\mathtt{MAC}}^{\mathtt{wuf\text{-}cma}}(\kappa). \tag{13}$$

Having eliminated possible forgeries and replay attacks with respect to $\mu_\mathcal{H}$ and $\mu_\mathcal{M}$ we observe that any t-fresh instance of $\mathcal{M}$ (or $\mathcal{H}$) which accepts has a partnered instance of $\mathcal{H}$ (or $\mathcal{M}$) which is also t-fresh.

Game $\mathbf{G}_3$. [*Signature forgeries for $\sigma_\mathcal{H}$ and $\sigma_\mathcal{M}$*] The simulation in this game aborts (and the output bit of the interaction is set at random) if $\mathcal{A}$ asks as part of its Send query to $\mathcal{F}$ (or to $\mathcal{H}$) a valid signature $\sigma_\mathcal{H}$ (or $\sigma_\mathcal{F}$) which was not previously output by an instance of $\mathcal{H}$ (or $\mathcal{F}$), so that

$$|\Pr[\mathsf{Win}_3^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_2^{\mathsf{ake\text{-}t}}]| \leq 2\mathsf{Succ}_{(Sig,Ver)}^{\mathtt{euf\text{-}cma}}(\kappa). \tag{14}$$

Having eliminated possible forgeries and replay attacks with respect to $\sigma_\mathcal{H}$ and $\sigma_\mathcal{F}$ we observe that any t-fresh instance of $\mathcal{F}$ (or $\mathcal{H}$) which accepts has a partnered instance of $\mathcal{H}$ (or $\mathcal{F}$) which is also t-fresh. In particular, the combination of this and the previous game ensures that if at least one t-fresh instance of some protocol party accepts then there exist partnered instances of the other two parties, which are also t-fresh.

Game $\mathbf{G}_4$. [*Pseudo-randomness of $K_{\mathcal{M},\mathcal{H}}$*] This game is identical to Game $\mathbf{G}_3$ except that Δ in each session on behalf of any partnered instances of $\mathcal{M}$ and $\mathcal{H}$ chooses the end-to-end key $K_{\mathcal{M},\mathcal{H}}$ as a random value and not as an output of $\mathtt{PRF}$, s.t.

$$|\Pr[\mathsf{Win}_4^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_3^{\mathsf{ake\text{-}t}}]| \leq q\mathsf{Adv}_{\mathtt{PRF}}^{\mathtt{prf}}(\kappa). \tag{15}$$

This game ensures independence between the pre-tunnel key k_t and $K_{\mathcal{M},\mathcal{H}}$ which may be revealed by $\mathcal{A}$ without compromising the t-freshness of the instance.

Game $\mathbf{G}_5$. [*Pseudo-randomness of k_t*] This game is identical to Game $\mathbf{G}_4$ except that Δ in each session on behalf of the partnered instances of $\mathcal{M}$ and $\mathcal{H}$ chooses the pre-tunnel key k_t as a random value and not as an output of $\mathtt{PRF}$, s.t.

$$|\Pr[\mathsf{Win}_5^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_4^{\mathsf{ake\text{-}t}}]| \leq q\mathsf{Adv}_{\mathtt{PRF}}^{\mathtt{prf}}(\kappa). \tag{16}$$

Note that in this game Δ computes χ as an encryption of this random value.

Game $\mathbf{G}_6$. [*Security of χ*] In order to exclude any information leakage about k_t upon its transmission to $\mathcal{F}$ we consider the following game, in which Δ in each session computes χ as an encryption $\mathcal{E}_{ek_\mathcal{F}}(\beta)$ where β is some randomly chosen value, independent of k_t. Note that Δ derives K_t on behalf of the partnered instances of $\mathcal{F}$, $\mathcal{M}$, and $\mathcal{H}$ still using k_t. It is possible to construct a distinguisher with given access to the *real-or-random* encryption oracle (and the decryption oracle) that is able to use $\mathcal{A}$ in this and the previous game to break the IND-CCA2 security of $(\mathcal{E},\mathcal{D})$, s.t.

$$|\Pr[\mathsf{Win}_6^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_5^{\mathsf{ake\text{-}t}}]| \leq q\mathsf{Adv}_{(\mathcal{E},\mathcal{D})}^{\mathtt{ind\text{-}cca2}}(\kappa). \tag{17}$$

Game $\mathbf{G}_7$. [*Pseudo-randomness of k_t*] In this final game Δ proceeds as before except that in each session on behalf of the partnered instances of $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{H}$ it chooses the tunnel key K_t as a random value and not as an output of $\mathtt{PRF}$. Obviously,

$$|\Pr[\mathsf{Win}_7^{\mathsf{ake\text{-}t}}] - \Pr[\mathsf{Win}_6^{\mathsf{ake\text{-}t}}]| \leq q\mathsf{Adv}_{\mathtt{PRF}}^{\mathtt{prf}}(\kappa). \tag{18}$$

As a result of this game the answer given to $\mathcal{A}$ in response to its $\mathsf{TestTunnelKey}$ query to some t-fresh instance of $\mathcal{M}$, $\mathcal{F}$, or $\mathcal{H}$ is a completely random value, regardless of the chosen bit b. Obviously, the probability of $\mathcal{A}$ to win in this game is given by the probability of a random guess, i.e.

$$\Pr[\mathsf{Win}_{7}^{\mathsf{ake\text{-}e2e}}] = \frac{1}{2}. \tag{19}$$

The combination of the above equations concludes the proof. □

5.5 Remarks on Efficiency

In case of roaming protocols the most significant impact on the communication complexity has the number of communication rounds on the path with the highest round trip time, i.e. the link between $\mathcal{F}$ and $\mathcal{H}$. Therefore, several solutions for the non-tunnel-based roaming mentioned in Section 3 aim to minimize this number. However, in a WRT session each request of $\mathcal{M}$ would imply one full communication round between the networks. Nevertheless, it is still desirable to minimize this communication before the tunnel is created in order to reduce the impact of possible network faults and DoS attacks. With this in mind we observe that our `AWRT` protocol requires one full communication round between the networks prior to the creation of the tunnel by $\mathcal{F}$. The additional half a round needed to deliver the authentication information to $\mathcal{H}$ can be easily interleaved with the first service request of $\mathcal{M}$. In this sense our `AWRT` protocol is comparable to the authentication protocols from [45, 53].

Further, the specification given in Figure 1 separates messages according to the two simultaneous mutual authentication processes that take place between $\mathcal{M}$ and $\mathcal{H}$, and between $\mathcal{F}$ and $\mathcal{H}$. This significantly simplifies the proof. However, some communication bandwidth can be saved by removing the `MAC` value $\mu_{\mathcal{M}}$ from the last message of $\mathcal{F}$ to $\mathcal{H}$, still allowing $\mathcal{H}$ to verify $\mu_{\mathcal{M}}$ indirectly, via the verification of $\sigma_{\mathcal{F}}$. This optimization relies on the assumption that $\mathcal{F}$ could not have created valid $\sigma_{\mathcal{F}}$ without knowing the required $\mu_{\mathcal{M}}$, thus, implying the necessary communication between $\mathcal{F}$ and $\mathcal{M}$ due to the unforgeability of `MAC` and the secrecy of $\alpha_{\mathcal{M}}$.

Finally, we observe that in `AWRT` the mobile device $\mathcal{M}$ does *not* need to perform any costly public-key operations, unlike the non-tunnel-based protocols in [9, 33, 40, 42, 44] and the WRT approach in [54]. Hence, `AWRT` can also be used with performance-constraint mobile devices such as PDAs and smart phones provided they have a wireless IP interface. Note that `AWRT` uses public-key operations for the transport of k_t and for the mutual authentication between $\mathcal{F}$ and $\mathcal{H}$.

Remark 4. If desired, the modularity of `AWRT` allows to completely remove public-key operations (and the corresponding long-lived keys) resulting in a more efficient protocol that would nevertheless still ensure end-to-end security between $\mathcal{M}$ and $\mathcal{H}$ in the presence of $\mathcal{F}$ (yet more efficient than [54]) without providing the tunnel key and the mutual authentication between the networks.

5.6 Ideas on Practical Realization and Tunnel Establishment

The description of `AWRT` in Figure 1 is kept general. Therefore, in the following we highlight some practical ideas on the realization of the protocol across the Internet based on the available standards.

The foreign network $\mathcal{F}$ will typically participate in `AWRT` using its own access point to which $\mathcal{M}$ connects on the data-link layer (layer 2) prior to the execution of `AWRT`. Since the communication between $\mathcal{M}$ and $\mathcal{F}$ takes part on layer 2 there is no need for $\mathcal{F}$ to assign an IP address to $\mathcal{M}$. Moreover, on the link between $\mathcal{M}$ and $\mathcal{F}$ the protocol can be implemented as a new EAP method within IEEE 802.1X. The home network $\mathcal{H}$ can participate in `AWRT` as a gateway or a protocol-specific server with some known DNS name or IP address, i.e. protocol messages exchanged between $\mathcal{F}$ and $\mathcal{H}$ can be transmitted over Internet through a classical UDP connection.

Once `AWRT` is successfully executed, $\mathcal{F}$ can continue acting as a layer 2 bridge to $\mathcal{H}$ for the whole roaming session. In this way $\mathcal{M}$ can be seen as "local" from the perspective of $\mathcal{H}$, i.e. $\mathcal{H}$ can allocate own IP address for $\mathcal{M}$ either as a parameter within `AWRT` or via DHCP. Messages between $\mathcal{M}$ and $\mathcal{H}$ can be tunneled using a simple IP-over-IP or upper layer protocol (such as L2TP [59]). The end-to-end traffic

between $\mathcal{M}$ and $\mathcal{H}$ can be protected using the Authentication Header (AH) or Encapsulating Security Payload (ESP) mechanisms in the *tunnel mode*, as defined within IPsec [36], whereby deriving the session key from $K_{\mathcal{M},\mathcal{H}}$, i.e., without executing IKE [20], the native key exchange protocol of IPsec.

Further, in order to avoid session hijacking attacks (incl. traffic injection), the tunnel path between $\mathcal{F}$ and $\mathcal{H}$ can also be secured using AH or ESP mechanisms, this time in the *transport mode*, whereby the corresponding session key should be derived from K_t.

6 Optional Protocol Extensions

AWRT can be extended in a modular way to deal with the issues of forward secrecy, resistance to certain types of DoS attacks, anonymity and unlinkability of roaming sessions, and accounting in commercial scenarios with the reimbursement of $\mathcal{F}$ upon the provided roaming service.

6.1 Forward Secrecy

Roughly speaking the requirement of forward secrecy for some session key means that an adversary should not be able to break the AKE-security of the protocol even if it gains access to the long-lived keys of participants after their instances have accepted in the test session. The common way to achieve forward secrecy is to derive the key from some independent ephemeral secret information which is valid only for one particular session.

Forward Secrecy of End-to-End Keys In order to achieve forward secrecy for $K_{\mathcal{M},\mathcal{H}}$ we can modify the basic version of AWRT using the classical Diffie-Hellman technique [25] and assuming that the necessary computations are performed in some cyclic group $\mathbb{G}$ of prime order q (polynomial in κ) generated by g. The idea is to derive $K_{\mathcal{M},\mathcal{H}}$ from an ephemeral secret $g^{x_\mathcal{M} x_\mathcal{H}}$ (with $x_\mathcal{M}, x_\mathcal{H} \in \mathbb{Z}_q$) as an output of $\mathtt{PRF}_{f(g^{x_\mathcal{M} x_\mathcal{H}})}(l_3|sid)$ where f is a randomness extractor (cf. [27]) and sid the corresponding session id. For this, $\mathcal{M}$ must choose its own secret exponent $x_\mathcal{M}$ and include $g^{x_\mathcal{M}}$ into its first protocol message, thus sending $B|g^{x_\mathcal{M}}|\mathcal{H}$ to $\mathcal{F}$ which then forwards $C|T|g^{x_\mathcal{M}}$ to $\mathcal{H}$. In turn, $\mathcal{H}$ chooses own secret exponent $x_\mathcal{H}$ and includes $g^{x_\mathcal{H}}$ into its protocol message $D|g^{x_\mathcal{H}}|\sigma_\mathcal{H}$ while computing $\mu_\mathcal{H}$ over $0|sid_\mathcal{H}|g^{x_\mathcal{M}}|g^{x_\mathcal{H}}$ allowing $\mathcal{M}$ to check the integrity of delivered $g^{x_\mathcal{M}}$ and sent $g^{x_\mathcal{H}}$. For the same reason, $\mathcal{M}$ must compute own $\mu_\mathcal{M}$ over $1|sid_\mathcal{M}|g^{x_\mathcal{M}}|g^{x_\mathcal{H}}$. Note that the AKE-security of such $K_{\mathcal{M},\mathcal{H}}$ would further rely on the hardness of the Decisional Diffie-Hellman Problem [16].

Forward Secrecy of Tunnel Keys In order to achieve forward secrecy for K_t we can apply a Generalized Diffie-Hellman technique, used e.g. in [19] for the purpose of group key exchange. Let $\mathbb{G}$ be a cyclic group and g its generator as described in the previous paragraph. The goal is to derive K_t from an ephemeral secret $g^{x_\mathcal{M} x_\mathcal{F} x_\mathcal{H}}$ (with $x_\mathcal{M}, x_\mathcal{F}, x_\mathcal{H} \in \mathbb{Z}_q$) as an output of $\mathtt{PRF}_{f(g^{x_\mathcal{M} x_\mathcal{F} x_\mathcal{H}})}(l_2|sid)$. In this case all computations involving the pre-tunnel key k_t become obsolete. Therefore, the modified protocol does not require to keep the decryption/encryption key pair $(dk_\mathcal{F}, ek_\mathcal{F})$ as part of the foreign network's long-lived key. The protocol proceeds as follows. $\mathcal{M}$ includes $g^{x_\mathcal{M}}$ into its first protocol message $B|g^{x_\mathcal{M}}|\mathcal{H}$ to $\mathcal{F}$ which in turn chooses own exponent $x_\mathcal{F}$ and sends $C|T|g^{x_\mathcal{M}}|g^{x_\mathcal{F}}|g^{x_\mathcal{M} x_\mathcal{F}}$ over to $\mathcal{H}$. Then, $\mathcal{H}$ chooses own $x_\mathcal{H}$ and replies with $D'|\sigma_\mathcal{H}$ where $D' := r_\mathcal{H}|g^{x_\mathcal{M} x_\mathcal{H}}|g^{x_\mathcal{F} x_\mathcal{H}}|\mu_\mathcal{H}$ with $\mu_\mathcal{H}$ computed over $0|sid_\mathcal{H}|g^{x_\mathcal{M}}|g^{x_\mathcal{F} x_\mathcal{H}}$ (allowing $\mathcal{M}$ to check the integrity of delivered $g^{x_\mathcal{M}}$ and sent $g^{x_\mathcal{F} x_\mathcal{H}}$), and $\sigma_\mathcal{H}$ computed over $D'|g^{x_\mathcal{M}}|g^{x_\mathcal{F}}|g^{x_\mathcal{M} x_\mathcal{F}}$ (allowing $\mathcal{F}$ also to check the integrity of $g^{x_\mathcal{M}}$, $g^{x_\mathcal{F}}$, and $g^{x_\mathcal{M} x_\mathcal{F}}$, delivered to $\mathcal{H}$ in the previous message of $\mathcal{F}$). $\mathcal{F}$ includes $g^{x_\mathcal{F} x_\mathcal{H}}$ into its message $E|g^{x_\mathcal{F} x_\mathcal{H}}$ to $\mathcal{M}$. Finally, $\mathcal{M}$ computes $\mu_\mathcal{M}$ over $1|sid_\mathcal{M}|g^{x_\mathcal{M}}|g^{x_\mathcal{F} x_\mathcal{H}}$, and $\mathcal{F}$ computes $\sigma_\mathcal{F}$ over $sid_\mathcal{F}|\mu_\mathcal{M}|g^{x_\mathcal{M} x_\mathcal{H}}$. It is easy to check that at the end of the successful protocol execution all parties are able to compute identical

$$g^{x_\mathcal{M} x_\mathcal{F} x_\mathcal{H}} = (g^{x_\mathcal{M} x_\mathcal{F}})^{x_\mathcal{H}} = (g^{x_\mathcal{M} x_\mathcal{H}})^{x_\mathcal{F}} = (g^{x_\mathcal{F} x_\mathcal{H}})^{x_\mathcal{M}}$$

and derive the same K_t. Note that in this case the AKE-security of such K_t would further rely on the hardness of the Group Decisional Diffie-Hellman Problem [18].

We stress that if forward secrecy should be simultaneously achieved for $K_{\mathcal{M},\mathcal{H}}$ and K_t then $\mathcal{M}$ and $\mathcal{H}$ must use different independently chosen exponents for the computation of each of these keys; otherwise $K_{\mathcal{M},\mathcal{H}}$ can be easily recovered using message D'. Note that achieving forward secrecy increases the protocol costs by additional modular exponentiations.

Remark 5. Some recent work on AKE-security, additionally, strengthens $\mathcal{A}$ by allowing it to corrupt participants *or* reveal ephemeral secrets used by (uncorrupted) participants during the test session. It might be possible to apply techniques used in [39] to achieve such *strong AKE-security* for the session keys computed in `AWRT` (at least for the end-to-end key $K_{\mathcal{M},\mathcal{H}}$). However, we leave such discussion out of scope of the current work.

6.2 Denial-of-Service and Hijacking

Here we present some ideas on how to enhance `AWRT` towards resistance against some types of DoS attacks. Due to the higher communication delays on the path between $\mathcal{F}$ and $\mathcal{H}$ it might be desirable to decrease the risk that $\mathcal{F}$ opens a connection to $\mathcal{H}$ for the third protocol message without gaining stronger confidence that a party which requested the tunnel is a valid mobile device hosted by $\mathcal{H}$; otherwise a DoS-attacker can compose the second protocol message $B|\mathcal{H}$ and then simply close own connection. A possible solution to minimize this risk is to equip $\mathcal{M}$ with a corresponding public key certificate issued by $\mathcal{H}$ and demand a signature on this second message (which also includes a fresh nonce of $\mathcal{F}$). Observe, that this solution, although computationally expensive, minimizes the risk since the attacker must either forge the signature or be a holder of some valid certificate. Nevertheless, it is not completely satisfactory since $\mathcal{F}$ is not able to judge (without further interaction) whether the certificate has not been revoked. One could further reduce the risk by requiring that device certificates are issued for some short validity period.

A similar threat is given for $\mathcal{H}$ which could be forced to keep the connection to $\mathcal{F}$ after the fourth protocol message without gaining stronger confidence that both parties $\mathcal{M}$ and $\mathcal{F}$ are legitimate; otherwise a DoS attacker may flood $\mathcal{H}$ with messages of the form $C|T$ and close own connection thereafter. The risk here can be minimized by requiring that $C|T$ is signed by $\mathcal{F}$ whereby the time-stamp T would also serve as a protection against replay attacks. Note that this signature cannot replace $\sigma_{\mathcal{F}}$ from the last message as this is required for the mutual authentication between $\mathcal{F}$ and $\mathcal{H}$. Additional confidence for $\mathcal{H}$ that $C|T$ has been sent by $\mathcal{F}$ after the communication with a valid $\mathcal{M}$ can be achieved by letting $\mathcal{M}$ compute a `MAC` value $\mu'_{\mathcal{H}}$ on the message $B|\mathcal{H}$ using $\alpha_{\mathcal{M}}$ and $\mathcal{F}$ to forward $C|T|\mu'_{\mathcal{H}}$ (in addition to its signature) to $\mathcal{H}$. Note that $\mu'_{\mathcal{H}}$ cannot replace $\mu_{\mathcal{H}}$ as the latter is needed for the mutual authentication between $\mathcal{M}$ and $\mathcal{H}$.

Finally, we consider protection against hijacking of a roaming session, mentioned in [53] for the non-tunnel-based case. Similarly, in WRT an attacker can wait until the protocol completes and the tunnel is established and then try to hijack the session from $\mathcal{M}$ or $\mathcal{H}$ (or both of them — tunnel hijacking). On the wireless link between $\mathcal{M}$ and $\mathcal{F}$ the session can be hijacked by using a more powerful transmitting device than $\mathcal{M}$, and on the path between $\mathcal{F}$ and $\mathcal{H}$ a hijacker can be any intermediate communication node. In WRT this attack may be used to waste the bandwidth resources assigned to the tunnel by $\mathcal{F}$. Our basic protocol allows for an elegant solution through the use of the tunnel key K_t requiring that each message which is supposed to be sent through the tunnel during the roaming session is authenticated such that $\mathcal{F}$ is able to verify its validity prior to the delivery. Note that a hijacker cannot bias the end-to-end communication between $\mathcal{M}$ and $\mathcal{H}$ which is protected by $K_{\mathcal{M},\mathcal{H}}$.

6.3 Confidentiality of $\mathcal{M}$'s Identity

Whenever a mobile device roams across several foreign networks (contract partners of $\mathcal{H}$) it might be desirable to keep its identity $\mathcal{M}$ undisclosed and even to prevent (possibly colluding) foreign networks from being able to link its roaming activities, thus eliminating the risk of profiling (examples for this can also be found in roaming for mobile phone networks, e.g., TMSI in GSM and UMTS). A straightforward solution is to let $\mathcal{M}$ choose some set of "one-time" roaming aliases and reveal them to its home network

before it roams. Obviously, this solution requires additional storage resources and its most significant disadvantage is that $\mathcal{M}$ must choose (and update) its aliases while being in the area covered by $\mathcal{H}$ (see also the discussion in [45]).

Our AWRT protocol allows for a more elegant approach if we rely on the fact that each home network usually serves as a foreign network from the perspective of its contract partners. This implies that $LL_{\mathcal{H}}$ used in AWRT also includes a key pair $(dk_{\mathcal{H}}, ek_{\mathcal{H}})$. Assuming that $\mathcal{M}$ knows $ek_{\mathcal{H}}$ we can slightly modify AWRT by replacing the identity $\mathcal{M}$ in the second protocol message with its encrypted equivalent $\mathcal{E}_{ek_{\mathcal{H}}}(\mathcal{M})$ which will also be used in the construction of session ids. Obviously, upon receiving $C|T$ the home network would be able to recover $\mathcal{M}$ through the corresponding decryption. Due to the IND-CCA2 property of $(\mathcal{E}, \mathcal{D})$ this solution would ensure not only anonymity of $\mathcal{M}$ towards $\mathcal{F}$ but also the unlinkability among different AWRT executions with $\mathcal{M}$; at the cost of one additional public key operation by $\mathcal{M}$ and $\mathcal{H}$.

6.4 Reimbursement of $\mathcal{F}$'s Roaming Costs

In commercial scenarios the foreign network $\mathcal{F}$ has to be reimbursed for the provided roaming service. Since $\mathcal{F}$ is not aware of $\mathcal{M}$ it usually presents its claims to $\mathcal{H}$. Note that the reimbursement process (as any payment process) cannot be realized solely using cryptographic techniques. Nevertheless, it might be desirable to allow $\mathcal{F}$ to support its claims with some cryptographic proof, which can also be shown to some judge. Observe, that in AWRT the foreign network obtains $\sigma_{\mathcal{H}}$ computed amongst other parameters on the identity $\mathcal{F}$ and the time-stamp T chosen by $\mathcal{F}$. However, the existence of $\sigma_{\mathcal{H}}$ alone does not guarantee that $\mathcal{F}$ created a tunnel to $\mathcal{H}$ and actually provided the roaming service. Therefore, upon the maintenance of the tunnel $\mathcal{F}$ may request further signatures $\sigma'_{\mathcal{H}}$ from $\mathcal{H}$ computed on some fresh time-stamp $T' > T$ and the session id, and close the tunnel connection if the expected signature is not delivered. Special control messages exchanged between $\mathcal{M}$ and $\mathcal{H}$ and protected using keys derived from $K_{\mathcal{M},\mathcal{H}}$ can be specified in order to ensure $\mathcal{H}$ (and $\mathcal{M}$) that the tunnel has been created and kept open during the interval $[T, T']$; otherwise $\mathcal{H}$ may refuse to send $\sigma'_{\mathcal{H}}$. A valid pair $(\sigma_{\mathcal{H}}, \sigma'_{\mathcal{H}})$ would then serve as a cryptographically protected acknowledgement of $\mathcal{H}$ that the tunnel connection was kept open by $\mathcal{F}$ within the time interval $[T, T']$.

7 Conclusion

The wireless roaming approach via tunnels preserves the home network's role as the actual service provider and has a number of additional security benefits. However, it opens new challenges on the authentication and key establishment protocol that precedes the establishment of the tunnel connection. Having formally specified the necessary requirements we designed an appropriate provably secure protocol AWRT which requires one communication round between the foreign and the home network prior to the creation of the tunnel. Additionally, we provide some ideas on possible extensions of the basic version of AWRT towards forward secrecy, DoS resistance, anonymity of roaming mobile devices and unlinkability of roaming sessions, and argued on the possible use of AWRT in commercial scenarios. The most appealing future work is the detailed specification of the protocol within the EAP framework and its implementation based on current standards within a suitable WRT architecture (the architecture can be based on the ideas given in [54]), whereby special attention should be paid to the realization of the tunnel connection, including the necessary control messages and their protection.

References

1. *802.1X-2004 IEEE Standard for Local and Metropolitan Area Networks – Port-Based Network Access Control.* IEEE, 2004.
2. 3GPP. Group Core Network and Terminals; Handover Procedures (Release 7). Technical report, 2007.
3. 3GPP. Group Services and System Aspects; 3G Security; Security Architecture. Technical report, 2008.
4. M. Abdalla, M. Bellare, and P. Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *CT-RSA'01*, *LNCS* 2020, pp. 143–158. Springer, 2001.

5. B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, and H. Levkowetz. Extensible Authentication Protocol (EAP). RFC 3748, IETF, 2004.
6. B. Anton, B. Bullock, and J. Short. Best Current Practices for Wireless Internet Service Provider (WISP) Roaming. Wi-Fi Alliance - Wireless ISP Roaming (WISPr), Feb. 2003.
7. G. Appenzeller, M. Roussopoulos, and M. Baker. User-Friendly Access Control for Public Network Ports. In *INFOCOM'99*, pp. 699–707. IEEE, 1999.
8. P. Bahl, S. Venkatachary, and A. Balachandran. Secure Wireless Internet Access in Public Places. In *ICC'01*, vol. 10, pp. 3271–3275. IEEE, 2001.
9. K. Bayarou, M. Enzmann, E. Giessler, M. Haisch, B. Hunter, M. Ilyas, S. Rohr, and M. Schneider. Towards Certificate-Based Authentication for Future Mobile Communications. *Wireless Personal Communications*, 29(3-4):283–301, 2004.
10. M. Bellare. New Proofs for NMAC and HMAC: Security without Collision-Resistance. In *CRYPTO'06*, *LNCS* 4117, pp. 602–619. Springer, 2006.
11. M. Bellare, R. Canetti, and H. Krawczyk. Keying Hash Functions for Message Authentication. In *CRYPTO '96*, *LNCS* 1109, pp. 1–15. Springer, 1996.
12. M. Bellare and C. Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In *ASIACRYPT'00*, *LNCS* 1976, pp. 531–545. Springer, 2000.
13. M. Bellare and P. Rogaway. Entity Authentication and Key Distribution. In *CRYPTO '93*, *LNCS* 773, pp. 232–249. Springer, 1993.
14. M. Bellare and P. Rogaway. Optimal Asymmetric Encryption — How to Encrypt with RSA. In *EUROCRYPT'94*, *LNCS* 950, pp. 92–111. Springer, 1994.
15. M. Bellare and P. Rogaway. The Exact Security of Digital Signatures - How to Sign with RSA and Rabin. In *EUROCRYPT'96*, *LNCS* 1070, pp. 399–416. Springer, 1996.
16. D. Boneh. The Decision Diffie-Hellman Problem. In *ANTS-III*, *LNCS* 1423, pp. 48–63. Springer, 1998.
17. D. Boneh, E. Shen, and B. Waters. Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. In *PKC'06*, *LNCS* 3958, pp. 229–240. Springer, 2006.
18. E. Bresson, O. Chevassut, and D. Pointcheval. The Group Diffie-Hellman Problems. In *SAC'02*, *LNCS* 2595, pp. 325–338. Springer, 2002.
19. E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably Authenticated Group Diffie-Hellman Key Exchange. In *CCS'01*, pp. 255–264. ACM, 2001.
20. C. Kaufman, Ed. Internet Key Exchange (IKEv2) Protocol. RFC 4306, IETF, 2005.
21. R. Canetti and H. Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In *EUROCRYPT'01*, *LNCS* 2045, pp. 453–474. Springer, 2001.
22. K.-K. R. Choo, C. Boyd, and Y. Hitchcock. Examining Indistinguishability-Based Proof Models for Key Establishment Protocols. In *ASIACRYPT'05*, *LNCS* 3788, pp. 585–604. Springer, 2005.
23. J.-S. Coron, M. Joye, D. Naccache, and P. Paillier. Universal Padding Schemes for RSA. In *CRYPTO'02*, *LNCS* 2442, pp. 226–241. Springer, 2002.
24. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *CRYPTO'98*, *LNCS* 1462, pp. 13–25. Springer, 1998.
25. W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Trans. on Information Theory*, IT-22(6):644–654, 1976.
26. M. Dischinger, A. Haeberlen, P. K. Gummadi, and S. Saroiu. Characterizing Residential Broadband Networks. In *IMC'07*, pp. 43–56. ACM, 2007.
27. Y. Dodis, R. Gennaro, J. Håstad, H. Krawczyk, and T. Rabin. Randomness Extraction and Key Derivation Using the CBC, Cascade and HMAC Modes. In *CRYPTO'04*, *LNCS* 3152, pp. 494–510. Springer, 2004.
28. eduroam. `http://www.eduroam.org`.
29. ETSI Technical Specification. Digital Cellular Telecommunications System (Phase 2+); Security Related Network Function. TS 100 929, ETSI, 2008.
30. Fon. `http://www.fon.com`
31. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP Is Secure under the RSA Assumption. *Journal of Cryptology*, 17(2):81–104, 2004.
32. P. Funk and S. Blake-Wilson. Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0). RFC 5281, IETF, 2008.
33. J. Gu, S. Park, O. Song, J. Lee, J. Nah, and S. W. Sohn. Mobile PKI: A PKI-Based Authentication Framework for the Next Generation Mobile Communications. In *ACISP'03*, *LNCS* 2727, pp. 180–191. Springer, 2003.
34. Q. Huang, D. S. Wong, and Y. Zhao. Generic Transformation to Strongly Unforgeable Signatures. In *ACNS'07*, *LNCS* 4521, pp. 1–17. Springer, 2007.
35. ITU-T. One-Way Transmission Time. G.114, 2003.

36. S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301, IETF, 2005.
37. S. Kim, H. Cho, H. Hahm, S. Lee, and M. S. Lee. Interoperability between UMTS and CDMA 2000 Networks. *IEEE Wireless Communications*, 10(1):22–28, 2003.
38. K. Lakshminarayanan and V. N. Padmanabhan. Some Findings on the Network Performance of Broadband Hosts. In *IMC'03*, pp. 45–50. ACM, 2003.
39. B. A. LaMacchia, K. Lauter, and A. Mityagin. Stronger Security of Authenticated Key Exchange. In *ProvSec'07*, *LNCS* 4789, pp. 1–16. Springer, 2007.
40. M. Long, C.-H. Wu, and J. D. Irwin. Localised Authentication for Inter-Network Roaming across Wireless LANs. *IEE Proceedings Communications*, 151(5):496–500, 2004.
41. S. McCann, R. Hancock, and E. Hepworth. Novel WLAN Hotspot Authentication. In *3G Mobile Communication Technologies Conf.*, pp. 59–63. IEE, 2004.
42. A. S. Merino, Y. Matsunaga, M. Shah, T. Suzuki, and R. H. Katz. Secure Authentication System for Public WLAN Roaming. *Mobile Networks and Applications*, 10(3):355–370, 2005.
43. U. Meyer. *Secure Roaming and Handover Procedures in Wireless Access Networks*. PhD thesis, TU Darmstadt, 2005.
44. U. Meyer, J. Cordasco, and S. Wetzel. An Approach to Enhance Inter-Provider Roaming Through Secret Sharing and its Application to WLANs. In *WMASH'05*, pages 1–13. ACM, 2005.
45. R. Molva, D. Samfat, and G. Tsudik. Authentication of Mobile Users. *IEEE Network*, 8:26–34, 1994.
46. R. M. Needham and M. D. Schroeder. Using Encryption for Authentication in Large Networks of Computers. *CACM*, 21(12):993–999, 1978.
47. NIST. Digital Signature Standard (DSS). FIPS PUB 186-2, 2000.
48. C. Rackoff and D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *CRYPTO'91*, *LNCS* 576, pp. 433–444. Springer, 1991.
49. C. Ribeiro, F. Silva, and A. Zúquete. A Roaming Authentication Solution for Wifi using IPsec VPNs with Client Certificates. In *TERENA Networking Conf.*, 2004.
50. C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). RFC 2865, IETF, 2000.
51. R. Robert, M. Manulis, F. De Villenfagne, D. Leroy, J. Jost, F. Koeune, C. Ker, J.-M. Dinant, Y. Poullet, O. Bonaventure, and J.-J. Quisquater. WiFi Roaming: Legal Implications and Security Constraints. *Int. J. of Law and Inf. Technology*, 16(3):205–241. Oxford University Press, 2008.
52. G. Rose and G. Koien. Access Security in CDMA2000, including a Comparison with UMTS Access Security. *IEEE Wireless Commmunications*, 11(1):19–25, 2004.
53. L. Salgarelli, M. Buddhikot, J. Garay, S. Patel, and S. Miller. Efficient Authentication and Key Distribution in Wireless IP Networks. *IEEE Wireless Communications*, 10(6):52–61, 2003.
54. N. Sastry, K. Sollins, and J. Crowcroft. Architecting Citywide Ubiquitous Wi-Fi Access. In *HotNets-VI*, 2007. available at `http://conferences.sigcomm.org/hotnets/2007/papers/hotnets6-final88.pdf`.
55. A. Shamir. How to Share a Secret. *CACM*, 22(11):612–613, 1979.
56. V. Shoup. Sequences of Games: A Tool for Taming Complexity in Security Proofs. Cryptology ePrint Archive, Report 2004/332, 2006.
57. D. Simon, B. Aboba, and R. Hurst. The EAP-TLS Authentication Protocol. RFC 5216, IETF, 2008.
58. S. Stamm, Z. Ramzan, and M. Jakobsson. Drive-By Pharming. In *ICICS'07*, *LNCS* 4861, pp. 495–506. Springer, 2007.
59. W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter. Layer Two Tunneling Protocol L2TP. RFC 2661, IETF, 1999.
60. Wisher. `http://www.wisher.com`