



TAMPEREEN TEKNILLINEN YLIOPISTO  
TAMPERE UNIVERSITY OF TECHNOLOGY

**RIKU ITÄPURO**

draft-17.11.2015 Master of Science thesis

Examiner: Prof. Jarmo Harju  
Examiner and topic approved by the  
Faculty Council of the Faculty of  
Computing and Electrical Engineering  
on 4th February 2015

# ABSTRACT

**RIKU ITÄPURO:**

Tampere University of Technology

draft-17.11.2015 Master of Science thesis, 63 pages, 5 Appendix pages

November 2015

Master's Degree Programme in Information Technology

Major: Information Security

Examiner: Prof. Jarmo Harju

Keywords: authentication, authorization, AAA, homenet, home networks, smartphone, SIM, trust-anchor, EAP-SIM, RADIUS

Today, home networks have become more complex and the home owner does not necessarily want to administer all aspects of it. Configuring home network devices does not differ much from configuring enterprise devices. One needs access, credentials to login and knowledge to operate the device. If the configuration is out-sourced to external parties and done remotely, those requirements need adaptation. Access to an end device must be provided from outside, a trustful operator must be hired and login credentials shared. For that, some beforehand set provisioning and distribution of authentication keys is needed.

In this work, an application running on user's smartphone represent this trustful operator. As there already exists an infrastructure within mobile phone subscribers, that is used in the study as a trusted base. To benefit from the mobile identification, it is shown how the authentication and authorization are done using an extendable authentication profile (EAP) and SIM-card. A theory to use EAP-SIM-authentication at home is presented and a simulated test bed to demonstrate that is built, tested and analyzed. The principle has been to reuse existing techniques combining them with such new areas as homenet and delegated management. Authentication claims are transported with WPA2 Enterprise. To further avoid complexity and granularity, we only use a simple model of management network.

As results, the smartphone authentication provides a trust anchor between a configuration agent and the home network. The home network management can be controlled from the smartphone keeping the local phone user still in control. SIM authentication's benefits are strong authentication and existing, large user-base, while its disadvantages include dependency to the mobile operator. Additionally, there remain challenges in keeping SIM's identity private and in disabling unwanted re-authentications.

# TIIVISTELMÄ

**RIKU ITÄPURO:** Älypuhelin kotiverkkojen luottamusankkurina

Tampereen teknillinen yliopisto

Diplomityö, 63 sivua, 5 liitesivua

marraskuu 2015

Tietotekniikan koulutusohjelma

Pääaine: tietoturva

Tarkastaja: Prof. Jarmo Harju

Avainsanat: tunnistaminen, valtuutus, AAA, homenet, kotiverkko, älypuhelin, SIM, luottamusankkuri, EAP-SIM, RADIUS

Kun tietoverkot kodeissa monimutkaistuvat, eivät kotikäyttäjät osaa tai halua enää ylläpitää niitä. Kotiverkkojen ylläpito ei eroa nykyisin paljon yritysympäristöistä. Käyttäjältä vaaditaan läsnäolo, tunnukset ja tietämys laitteiden operointiin. Näitä vaatimuksia täytyy soveltaa, jos ylläpito ulkoistettaisiin ja pääsy kotiverkkoihin sallittaisiin. Luotettava toimija on palkattava ja jaettava tälle tunnistautumiskeino sekä pääsy kohdelaitteelle ulkoa käsin. Tämä edellyttää ennakkotoimia ja tunnistautumisavainten jakelua.

Tutkielma kuvaa toimijaa sovelluksena käyttäjän älypuhelimessa. Koska käyttäjä on matkapuhelinliittymällään jo osa luotettua tilaajarekisteriä, käytetään tätä ominaisuutta hyväksi työssä luottamuksen rakentajana. Matkapuhelintunnistusta hyödynnetään näyttämällä, kuinka laajennettavissa oleva tunnistusmenetelmä EAP yhdistetään SIM-kortilla olevaan tilaajatietoon. Lisäksi näytetään, kuinka pääsynvalvonta hoidetaan RADIUS-protokollalla. EAP-SIM-pohjaisen tunnistuksen teorian tueksi esitetään käytännössä simuloitu käyttöympäristö. Periaatteena on ollut käyttää olemassaolevia tekniikoita yhdistäen niitä uusiin alueisiin, kuten homenet-määritysten kotiverkkoihin ja edustajalle ulkoistettuun hallintaan. Tunnistus- ja valtuutustietojen välittämisen hoitaa WPA2 Enterprise sisältäen RADIUS ympäristön. Välttääksemme monimutkaisuutta ja tarpeetonta hienorakeisuutta, käytämme yksinkertaista hallintaverkkomallia, jonka rajalla on kotiverkosta muuten erollaan oleva älypuhelin.

Tuloksena näytetään, että matkapuhelimella tehty tunnistautuminen luo luottamusankkurin ulkoisen edustajan ja kodin hallintaverkon välille avaten edustajalle hallintayhteyden kotikäyttäjän valvonnassa. SIM-tunnistuksen hyötyjä ovat vahva tunnistus ja laaja käyttäjäkanta. Haittoina ovat riippuvuus teleoperaattorista, käyttäjän identiteetin paljastumisen uhka ja ei-toivottu automaattinen tunnistautuminen.

# PREFACE

[\*TBD\*] PREFACE TEMPLATE! SKIP.

This document template conforms to Guide to Writing a Thesis at Tampere University of Technology (2014) and is based on the previous template. The main purpose is to show how the theses are formatted using LaTeX (or L<sup>A</sup>T<sub>E</sub>X to be extra fancy) .

The thesis text is written into file `d_tyo.tex`, whereas `tutthesis.cls` contains the formatting instructions. Both files include lots of comments (start with `%`) that should help in using LaTeX. TUT specific formatting is done by additional settings on top of the original `report.cls` class file. This example needs few additional files: TUT logo, example figure, example code, as well as example bibliography and its formatting (`.bst`) An example makefile is provided for those preferring command line. You are encouraged to comment your work and to keep the length of lines moderate, e.g. <80 characters. In Emacs, you can use `Alt-Q` to break long lines in a paragraph and `Tab` to indent commands (e.g. inside figure and table environments). Moreover, tex files are well suited for versioning systems, such as Subversion or Git.

Acknowledgements to those who contributed to the thesis are generally presented in the preface. It is not appropriate to criticize anyone in the preface, even though the preface will not affect your grade. The preface must fit on one page. Add the date, after which you have not made any revisions to the text, at the end of the preface.

Tampere, 1.5.2015

Teemu Teekkari

# TABLE OF CONTENTS

1. Introduction . . . . .	1
2. Authentication, Authorization, and Trust . . . . .	5
2.1 802.1X . . . . .	5
2.2 RADIUS . . . . .	6
2.3 WPA2 . . . . .	8
2.4 EAP . . . . .	8
2.5 SIM-based authentication . . . . .	9
2.6 Analysis of EAP-SIM protocol . . . . .	11
2.7 Trust . . . . .	14
3. Home network architecture . . . . .	15
3.1 Home network architecture and IETF . . . . .	15
3.2 Centralization trends in management . . . . .	16
3.3 Methods for introducing trust anchor into the home network . . . . .	17
4. Design of home network trust anchor and separation of change management	20
4.1 Chosen AA design . . . . .	20
4.2 AA component location scenarios . . . . .	22
4.3 Modification of RADIUS messages . . . . .	26
4.4 Similarities with Lock-and-Key method . . . . .	26
4.5 Chosen management model . . . . .	28
4.6 Disconnecting the local controller and offline changes . . . . .	29
4.7 Summary of the chosen solution . . . . .	30
5. Implemented Solution . . . . .	31
5.1 EAP-SIM authentication test bed . . . . .	31
5.2 Detailed description of test runs . . . . .	33
5.3 Simulation on smartphone Jolla . . . . .	37
5.3.1 Cross-compile the programs inside a virtual machine . . . . .	38
5.3.2 Running the new binary on real smartphone. . . . .	39

6. Analysis, Results and Discussion . . . . .	41
6.1 Security considerations . . . . .	41
6.1.1 Confidentiality (privacy) . . . . .	41
6.1.2 Integrity . . . . .	44
6.1.3 Accessibility, DoS and scalability . . . . .	44
6.1.4 RADIUS weaknesses and strengths in limited use cases . . . . .	44
6.1.5 Replay, re-use, re-auth, and brute-force challenges . . . . .	45
6.1.6 Hardware tampering . . . . .	46
6.1.7 Mitigation methods against radio capturing . . . . .	46
6.2 Deployment difficulty and costs . . . . .	47
6.3 Platform specific issues . . . . .	49
6.4 Usability with access points . . . . .	50
6.5 Discussion . . . . .	51
6.6 Alternative management models . . . . .	53
6.7 Bootstrapping with and without PKI . . . . .	54
7. Conclusion . . . . .	57
References . . . . .	63
APPENDIX A: Shell, that starts programs and logs . . . . .	64
APPENDIX B: WPA2-supPLICANT settings . . . . .	66
APPENDIX C: RADIUS server configuration . . . . .	67
APPENDIX D: hlr_auc_gw program's Milenage file . . . . .	68

## LIST OF FIGURES

1.1 Local Controller and Collaborative Management Design . . . . .	2
1.2 Goal of the thesis . . . . .	4
2.1 EAP-logical layering and encapsulation . . . . .	9
2.2 Bird's-eye view to EAP-SIM components . . . . .	12
2.3 Successful EAP-SIM full authentication with RADIUS . . . . .	13
3.1 Trust circles in the beginning . . . . .	18
4.1 Scenario I with 3 separate domains: Cloud, MNO and home net . . .	23
4.2 Scenario II with AuthZ in home network . . . . .	24
4.3 Scenario III with outsourced AA and backup cache for AuthZ . . . .	25
4.4 Scenario IV, offline AA . . . . .	25
4.5 Cisco's view of Lock-and-key access . . . . .	27
4.6 Managed device trusts Authentication Server . . . . .	27
5.1 Plan to benefit from SIM-authentication at home . . . . .	31
5.2 EAP-SIM AuthN messaging in simulation test bed . . . . .	32
6.1 Changing TMSI without the authentication server (or MNO) . . . . .	42
6.2 Future, direct to Cloud connected CPE. AuthN comes from MNO . .	54

## LIST OF TABLES

2.1	Comparison of WPA2-PSK and WPA2-ENTERPRISE modes . . . .	8
2.2	WPA2-Enterprise client setup with EAP-PEAP-MSCHAPv2 and EAP-SIM . . . . .	12
4.1	Location of AA, AuthN and AuthZ in scenarios I-V . . . . .	22
5.1	Versions of components in tests and their architecture. . . . .	38



## LIST OF PROGRAMS

5.1	E90 trying to use its internal EAP-SIM . . . . .	33
5.2	Successful EAP-SIM Authentication . . . . .	34
5.3	First detailed indication of IMSI, captured from WPA2-supPLICANT . .	35
5.4	EAP client's identity transformed, capture from RADIUS server . . .	36
	./apd-tty . . . . .	64
	wpa-simtest.conf . . . . .	66
	hostapd.conf . . . . .	67
	hlr_auc_gw.milenage_db . . . . .	68

## LIST OF ABBREVIATIONS AND SYMBOLS

3GPP	3 <sup>rd</sup> Generation Partnership Project
AAA	Authentication, Authorization, Accounting
AKA	Authentication and Key Agreement
AuC	Authentication Center
CPE	Customer Premise Equipment
EAP	Extensible Authentication Protocol
GAA	Generic Authentication Architecture
GBA	Generic Bootstrapping Architecture, 3GPP standard for user authentication with help of shared key from operator, part of GAA.
GSM	Global System for Mobile Communication (earlier Groupe Spécial Mobile)
HLR	Home Location Registry, ...
IEEE	Institute of Electrical and Electronics Engineers
IMSI	International Mobile Subscriber Identity
ISP	internet service provider
MNO	mobile network operator
MSISDN	Mobile Station Integrated Services Digital Network, user's phone number
RADIUS	Remote Authentication Dial In User Service, protocol and server, AAA service
SIM	Subscriber Identity Module, a smartcard. Also USIM program running in UICC card (UMTS networks)
SSID	Service Set Identifier, identifies Wi-Fi network
TMSI	Temporal Mobile Subscriber Identity
TUT	Tampere University of Technology
Wi-Fi	Wireless local network, implements IEEE 802.11 standards
WPA	Wireless Protected Access version 1
WPA2	Wireless Protected Access version 2

## TERMINOLOGY

**802.1X** port-based access control standard

**access point, AP** connects wireless (Wi-Fi) clients to a wired network, used here to encapsulate EAP messages to RADIUS and to forward them to an authentication server

**authentication server** a server checking identity claims, for example RADIUS server

**authenticator** a local entity that makes the authentication (and authorization) decision for a client based on local and remote claims, part of 802.1X standard

**mobile (network) operator, MNO** a mobile network operator, knows the connection between a SIM-owner and SIM secrets

**proxying RADIUS** RADIUS server standing between a RADIUS client and an authentication server, part of RADIUS server chain

# 1. INTRODUCTION

Managing computer and network devices can be hard. Modern homes have become similar to small offices regarding the equipment present in them. Earlier, it was sufficient to make just minimal settings at home to a modem (cable, phone or radio) and to connect it to the home computer to get a fully working home network with internet connectivity. Today, the home network has expanded with countless devices available. Entertainment centers (AV-amplifiers, media players, game consoles), manageable network devices (switches/routers), and mobile phones represent new devices and network segments beside computers and printers. Sensors and controller devices from the Internet of Things domain bring their own increment to the device count at home. Connecting these devices to the network remains trivial, but managing the network afterwards has become challenging and complex.

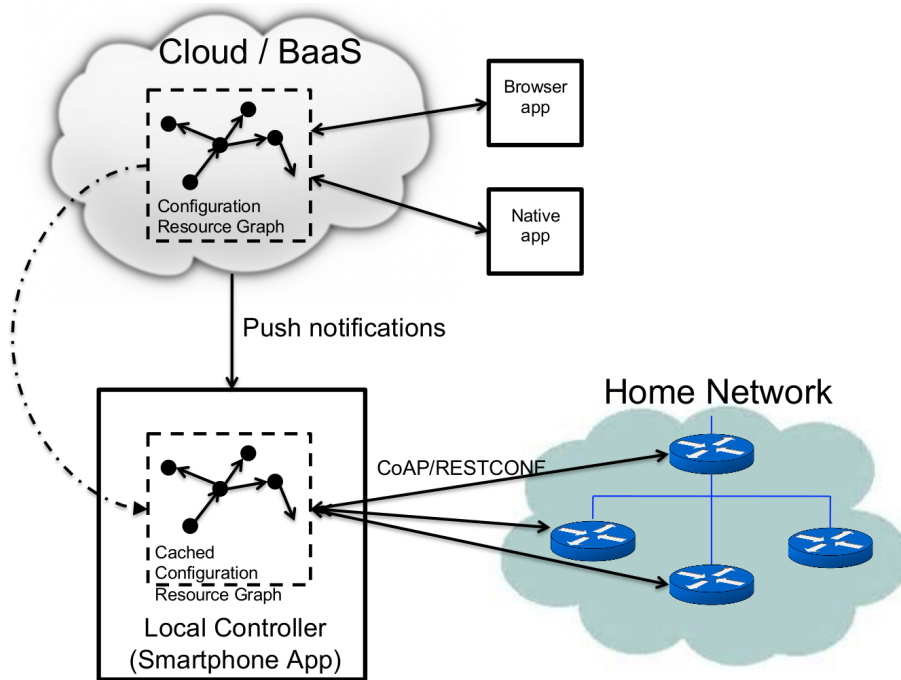
There might be separate areas in homes that have different needs regarding connectivity, resources, and access. In addition, devices in separate segments may not belong to the home owner anymore, hence needing their own administrative parties. For example, an electricity company may have a sensor and controller network, which physically uses the home network, but is logically separated from the other parts of the home network. It is therefore important to keep track of who is allowed to access the different parts of the home network.

The configuration choices in networking devices take some amount of expertise that is not necessarily present at every home. There could exist a market for an external consultant service, which would remotely manage the home network. Remote management eliminates the consultant's physical presence at home and so reduces external actors' costs, but adds many questions regarding security, not only to overcome firewalls, NATs, and the disconnectivity from the Internet. Persons who are allowed to make configuration changes are today often authenticated only by a simple password and physical presence at home. If external help was present, the home owner would need more control allowing only authorized operators in, because the protection used earlier would be missing.

Finally, it is challenging to find a common, trusted entity, upon which every actor could base their trust. Mutual trust is needed to join the formerly unknown parties together safely. Summarized, the problems are the delegated network management, remote provisioning, and trust finding. At its roots, this is an authentication and authorization problem.

One model to solve these problems is to separate the management and control functions from the connectivity and routing issues. Silverajan et al. [40] proposes a model where the management is achieved through a service in a cloud. The cloud service is of type Backend-as-a-Service (BaaS) which is well suited for mobile application usage. The configuration model of devices in a home is mirrored to the cloud as a resource graph. Changes can be planned ahead in the cloud and committed (Pushed) later to the home using configuration tools running over CoAP (Constrained Application Protocol) via a local controller point located at home. CoAP is suitable for IoT devices having constrained resources.

The managers operate in the cloud with the configuration data and the cloud verifies those managers, but the question that remains is how to connect the cloud service to the home network through the local controller securely. The local controller at home would approve the changes, and a smartphone is assumed to function in the local controller role. It will operate as a bridge between the cloud and the home network. See Figure 1.1 for the design of this architecture.



**Figure 1.1** Local Controller and Collaborative Management Design

The delegated service provider therefore does not need to have a direct data access to the home but only to the cloud-based service in order to be able to manage the home network devices. Consultant service is not the only possible delegation for a home network.

One of the security issues is the authentication and authorization from the cloud to the home network. To secure the connection from the cloud service (remote controller) to the home network, there needs to be a mutual trust between the end points, and the research problem here is how to enable the trust between the local controller and the home network as the local controller lies at the edge of the home network. The connection between the remote and local controller is assumed to already be trusted.

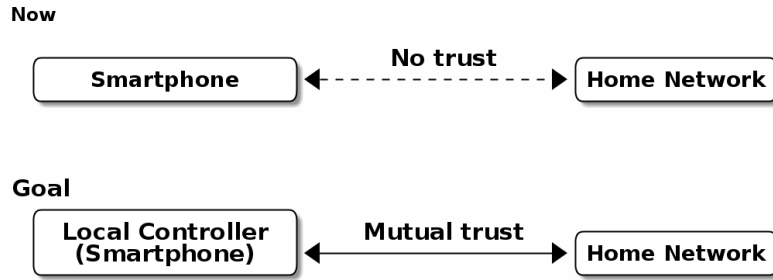
Any encryption between devices needs trusted key exchange beforehand, and finding and establishing trust is needed for that. That is called the key distribution problem. Public and private keys solve the key exchange part, but only partially, because the trust still must be found somewhere. The trust can be derived from the facts that already are known. The ultimate trust can be achieved by verifying the trust chains until the chain reaches a trust anchor. The trust anchor is the fact, state, or place, where the derivation of trust is no longer done, but accepted per se. Combining existing techniques, this thesis presents one possible way to bind the home network's trust to the smartphone's unique, existing secret keys inside the smart card's Subscriber Identify Module (SIM), which then would function as a trust anchor.

The above mentioned cloud solution for delegated home network management currently has a preliminary authentication and access model using pre-defined credentials for accessing the local network in general, and other credentials for secure SSH-connection from the local controller device to configuration targets [40, Chap.4]. That does not yet handle the bootstrap of the infrastructure, i.e., the first trust is taken as given.

The smartphone with its SIM and an existing key infrastructure to the mobile network operator (MNO) would later eliminate the requirement for an additional credential distribution. That issue is studied in this thesis. Although the smartphone provides an alternative authentication method with its SIM key, usual methods to authenticate are still plain username-password combinations. This security issue must be solved before delegation in the cloud can happen.

The goal presented in Figure 1.2 is to make the smartphone a central, trusted controlling point for managing purposes. The normal access between the Internet

and the home network should stay unchanged. The human aspect and usability are also important, but the focus will still be on the authentication and authorization part of the home net management with smartphone as a trust anchor. The proposed model should nevertheless require less effort than the currently used methods on distributing user credentials, finding the right place for them to be inserted, and ensuring that they are written correctly. Besides those, problems such as limited connectivity are studied.



*Figure 1.2 Goal of the thesis*

The thesis is structured as follows: authentication–authorization model is explained in Chapter 2. Chapter 3 describes security in current home network architecture and practices for configuring it. Chapter 4 discusses methods to bring a trust anchor into the home network and explains the chosen method. One specially crafted problem is how the scenarios presented here can be tested without knowing the SIM card’s secret keys and without a real phone operator involved. Those experiments are described in Chapter 5. Results are discussed in Chapter 6, and Chapter 7 concludes the thesis.

## 2. AUTHENTICATION, AUTHORIZATION, AND TRUST

Authentication, authorization, and accounting services (AAA) are components for access management. AAA-protocols do not dictate policies, i.e., who is granted an access or what operations a user is allowed to do. They only transport this information between a client who needs them and a server authorized to provide them. Often, the last 'A', which stands for accounting, has been neglected and also here only the first two 'A's are used and later described as AA services. Authentication (AuthN) answers the question of how to identify users and prove that they really are who they claim to be. Authorization (AuthZ) answers the question of what operations the identified users are allowed to do and enforces the usage policy. The rest of the thesis uses short terms AuthN and AuthZ.

In very small environments, AA service is built on a static backend, such as a file on a protected target that an entity wants to access. There, AuthN is checked against a credentials file, and AuthZ is given from a service specific policy file. To be more exact, the identification preceding the authentication is the part where the entity claims and presents its identity to the access controlling system. That can involve sending a username, login name, or other identifier. Authentication in turn is the part where those facts are verified. AuthZ involves checking which rights are available for the authenticated entity.

Before we introduce SIM-based authentication used throughout the thesis, protocols 802.1X, WPA2, EAP and RADIUS are described in the following sections. After this, we expand the term *trust*.

### 2.1 802.1X

802.1X [20] is an IEEE standard protocol for port-based access control. Ports are physical layer ports, not to be mixed with Layer-4 ports such as TCP/UDP ports. Network access through a specific physical port is restricted (controlled) from a client (called Supplicant) until the client has successfully performed AA. An 802.1X



device, where the ports are located, is called authenticator. Third party in 802.1X is an authentication server.

The terms *authenticator* and *authentication server* are easily mixed, but their roles are different: an authenticator works as a gate-keeper to the ports between a supplicant and the network, while an authentication server handles the AA processes. At home the, authenticator usually lies inside the access point (AP), which functions also as a router. However, in large enterprise networks, the authenticator may be a centralized unit, and multiple access points function only as radio stations without routing or authenticator properties.

## 2.2 RADIUS

RADIUS is the most popular provider for the AAA-services [13, p.75]. It was used first with remote terminal and dial-up modem users, hence the name Remote Authentication Dial-In User Service. Later, it was used as a centralized AAA for networking devices such as switches and routers.

RADIUS protocol is a stateless, request-response type client-server protocol. There are four types of RADIUS messages defined in RFC2865 that are used in the AA. ACCESS-REQUEST and ACCESS-CHALLENGE cover both AuthN and AuthZ messaging, while the final RADIUS message is either ACCESS-ACCEPT or ACCESS-REJECT, based on the result given by the final RADIUS server.

Today, RADIUS has some shortcomings and fixing them is no longer reasonable as development has shifted to another AAA protocol called Diameter, which is already in use in 3GPP and 4G networks [43]. Nevertheless, as RADIUS is so wide-spread, it is still used in many places instead of Diameter. Currently, the main environment of RADIUS, besides AA in network managing, are wireless connections (Wi-Fi) in enterprises and nationwide community federations.

When local Wi-Fi groups such as “SparkNet”, “Langaton Tampere”, or “Wippies” started to form around 2005 in Finland, they used 802.1X and RADIUS for AA. Those networks did still have as an alternative AA method a captive portal technique, where the user had to first authenticate on a WWW-page before getting an access. 802.1X and RADIUS brought an external, central RADIUS server for the automatic authentication of requests, without the burden of the captive portal.

The members of the Wi-Fi groups could then use the network in any location where the same uniform SSID (Service Set Identifier) was seen. Roaming became possible

if one found a familiar SSID outside the home area. Later, agreements were made between different local groups to allow roaming, and so federations were born.

As seen from the federated Wi-Fi groups, RADIUS servers can be chained to form a tree. The reasons for the chaining are load balancing and high availability, centralization of distant servers, and a federation of different domains. With a RADIUS hierarchy, the messages can be proxied to the next RADIUS server in the chain, depending on the settings on the proxying RADIUS server.

RADIUS messages are normally not protected from eavesdropping, but they have integrity fields to notice if they have been tampered with. The integrity field is called a Message Authenticator. Notice the use of the term *authenticator* in a different context here, not meaning 802.1X's authenticator. When using RADIUS to AuthN and AuthZ, Requests can only belong to ACCESS-REQUEST messages while Responses can be either an ACCESS-ACCEPT, ACCESS-REJECT, or ACCESS-CHALLENGE message. The Message Authenticator field is sent as last Attribute Value Pair (AVP) of each RADIUS message, and it can belong to either Request or Response. [17, p.20].

The Request Authenticator is a 16-octet long, random number in an ACCESS-REQUEST message, but the Response Authenticator for it is achieved with a one-way MD5 digestion function. The digest is taken from the concatenation of Code, ID, Length, corresponding RequestAuth, Attributes, and a Secret, and can look like *3fef65608...2a79*.

```
Response Authenticator =
    MD5(Code | ID | Length | Request Authenticator | Attributes | Secret)
```

The Secret is the shared secret which has been configured between RADIUS client-server pairs, and it protects some parts of the traffic. Different RADIUS client-server pairs may use different shared secrets, and the RADIUS server must separate them according to the client's IP address to manage proxied RADIUS requests [17].

An exception to the above-mentioned plain-text messaging are the user passwords. If the user password was to be transmitted in RADIUS, it would be sent first through an exclusive OR (XOR) function together with an MD5 digested Secret and Request Authenticator.

```
User-Password = XOR(password, MD5(Secret | Request Authenticator))
```

In the following chapters, it is discussed how the proxying servers take part in the AA decisions. Of main interest there is whether it is possible to inject or modify AuthZ information in those proxying RADIUSes in cases where AuthN and AuthZ are provided from different places [3]. A secondary goal is to universally divide AA regarding the client's domain in the federation.

## 2.3 WPA2

A Wireless Protected Access (WPA or WPA2) protects the traffic in a wireless, shared media, where everyone otherwise can simply listen to all the radio traffic. WPA2 enables both an authenticated access and a message encryption between a client device and a wireless access point (AP) by negotiating session keys. This happens after 802.1X has opened the virtual port in the AP for the client.

The WPA (version 1) was an early subset of then upcoming 802.11i standard, while the WPA2 is the full implementation, also denoted as IEEE 802.11i-2004. The term WPA2 is used throughout the thesis. Client software for 802.11i is called a WPA2-Supplicant, and it is used in wireless clients to communicate with the authenticator.

The WPA2 has two modes of protection: one for groups with a common, pre-shared key (WPA2-PSK, also known as WPA2-Personal) and one for individuals having their own key (WPA2-RADIUS, also known as WPA2-Enterprise). With WPA2-RADIUS, revoking individual access is easier, but client setup is slightly more complicated than on WPA2-PSK, as seen on Table 2.1.

**Table 2.1** *Comparison of WPA2-PSK and WPA2-ENTERPRISE modes*

Property	WPA2-PSK	WPA2-ENTERPRISE
suitable for groups	x	
suitable for individuals		x
individual client revocation		x
client setup	easy	intermediate

## 2.4 EAP

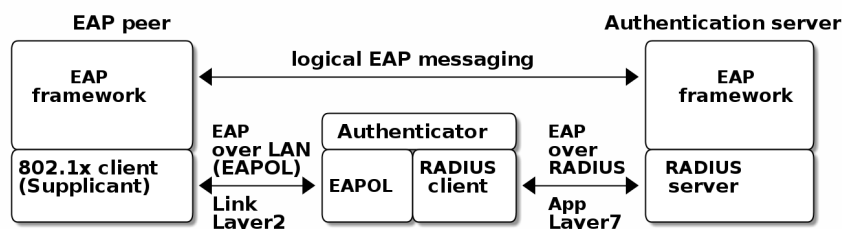
New AuthN methods are invented all the time. Instead of implementing them into 802.1X, it was extended with a modular framework called EAP (Extensible Authentication Protocol) [2]. Researchers justify using EAP, as it provides flexibility independent from underlying technology, whether wireless or wired, and integration with AAA infrastructures, although it adds some overhead to AuthN [35]. Different

authentication methods, such as hashed passwords, TLS certificates, or SIM/AKA using smartphone's SIM card, can be used with EAP. This work uses the EAP-SIM authentication method.

EAP describes only the messaging form, so EAP messages needs to be encapsulated inside another protocol. In Wi-Fi, between a smartphone and an AP, the EAP can be encapsulated into 802.1X protocol (as EAPOL) or into a protected EAP(PEAP) [32] before sending it into air. In a wired network, those EAP messages are translated and encapsulated into RADIUS.

The encapsulation is described in Figure 2.1 where it can be seen that the EAP messaging takes place logically between the EAP peer and the authentication server. On a lower transport layer between them, there is an EAP authenticator, which transfers EAPOL messaging into a RADIUS message.

Furthermore, EAP is used to transfer AuthN messages only. EAP includes neither AuthZ information, which is RADIUS's responsibility, nor session keys, which are negotiated by WPA2. In the end, the authenticator is responsible for opening access for the EAP peer, as 802.1x dictates.



*Figure 2.1 EAP-logical layering and encapsulation*

## 2.5 SIM-based authentication

SIM associates a physical card used in smartphones to a subscriber of the Mobile Network Operator (MNO). SIM here means the secret keys and the application in mobile phone's SIM or USIM inside UICC (Universal Integrated Circuit Card). The secret keys are hardware-protected and only usable for applications in a SIM card. The SIM's storage also includes a unique serial number ICCID (Integrated Circuit Card Identifier) which identifies the SIM globally, and a unique IMSI (International Mobile Subscriber Identity). The IMSI is a composition of digits belonging to the Mobile Country Code (MCC, 2 digits), Mobile Network Code (MNC, 2-3 digits), and Mobile Subscriber Identification Number (MSIN, 10 digits at maximum). It is not

to be mixed with MSISDN (Mobile Station Integrated Services Digital Network), which is the user's full international phone number.

SIM card usage can be controlled with two passwords: PIN and PUK. PUK is used as a remedy if PIN has been entered incorrectly too many times. If the card has other applications, such as a mobile electrical signature application *Mobiilivarmenne* (see Section 3.3), they may have different keys and codes.

The passwords, keys, and cards are distributed by the MNO. They provide the mobile network connectivity to the customers of the MNO. The secret keys are used for authenticating an IMSI to an MNO, and this enables MNOs to identify their customer in the network and to charge them accordingly. The client's identity is verified when the SIM is delivered. It is assumed that the SIM card represents its owner, but in reality nothing prevents an identity thief from stealing someone's SIM card. Although the 4-digit PIN tries to prevent the usage of the stolen SIM, this is considered a weak safe [28, p.31]. The most important outcome of this distribution is the achieved trust between the client and the MNO.

AA services need to trust some entity endpoint. In case of the MNO and the SIM, they already mutually trust each other, and the SIM can be used to open access to the mobile networks. Access to Wi-Fi networks still needs a separate access credential, and that was the reason for developing EAP-SIM and later the derivatives EAP-AKA and EAP-AKA'. The goal was to combine the existing keys used in GSM (Global system for Mobile communication) in a secure way to Wi-Fi access. The general purpose EAP-methods existing in 2004 were not compatible with GSM protocols for this purpose. [19, p.93] The results of that development gave us EAP-types EAP-SIM, EAP-AKA, or EAP-AKA'(AKA-PRIME).

EAP-SIM is the original type created for GSM networks and defined in RFC4186 [18]. It is a challenge-response method and similar to AuthN used in GSM, but it adds mutual AuthN; i.e., also the network is authenticated. Beginning from 3GPP networks, the new types EAP-AKA and AKA' can be used. EAP-AKA is defined in RFC4187 [6] and uses the 3GPP's AKA (Authentication and Key Agreement) protocol. It adds to EAP-SIM additional parameters [7], such as sequence numbering from the MNO to protect replay attacks and more advanced digestion functions instead of SHA-1. Otherwise the protocol messaging is same as in EAP-SIM. Finally, there exists EAP-AKA' that enhances AKA by including a Service Set Identifier (SSID) in the key derivation function, which limits the possibility of using possibly compromised network's nodes and keys.

Using EAP-SIM means using the secret key inside a SIM card with A3/A8 algorithms to generate valid responses for the challenges coming from an MNO and to derive session keys. The algorithms A3/A8 and their possible implementations (COMP128, COMP128v2, COM Pv3) are not of interest in this work. They can be MNO specific or known reference algorithms.

EAP-SIM variants provide strong AuthN, which here means two-factor AuthN. One factor is something you own (the physical SIM card) while another is something you know (the SIM card's PIN). Biometric factor, i.e., what you are, is not used here, but that would be a third possible factor. Software-based certificates, while stronger than regular passwords, do not, on the other hand, possess the properties *non-copiable* or *unique*, so they can only be considered as strong passwords and do not fulfill the requirements for two-factor AuthN. If we nonetheless were using software certificates with a method such as EAP-TLS, then the certificates (for CA and client) and the private key should still be provisioned first, which would defeat what we want to achieve in easy user experience.

Disadvantages with SIM are the dependency on the mobile operator and internet connection, although disconnectivity issues are later addressed partly in Section 4.6. Using a smartphone may cost money, either to the client or to the service provider, but the costs could be lower than using an SMS, because the network used is an IP network instead of a cellular phone network.

In many parts, SIM variants of EAP are simpler than other EAP variants to the mobile client. Table 2.2 compares the setup of Wi-Fi in clients of one existing organization to EAP-SIM. The example is taken from setting up Nokia Communicator model E90, but in general, the same options are also needed for other clients, also with laptops. It is noteworthy that plain EAP-SIM will not support identity hiding. This will be discussed further later on. If we also added PEAP to EAP-SIM (in last column of Table 2.2), the comparison would be more fair. As can be seen from the table, leaving certificates out from the environment makes the client setup easier with the price of revealing the smartphone user's identity.

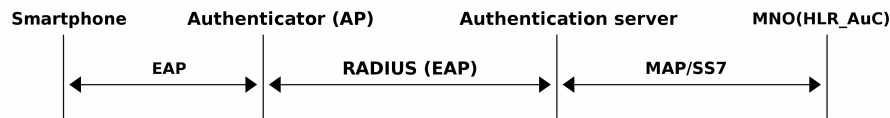
## 2.6 Analysis of EAP-SIM protocol

A bird's-eye view to the EAP-SIM protocol messaging between the smartphone, an AP, an authentication server, and an MNO with its Home Location Registry Authentication Center (HLR\_AuC) is described in Figure 2.2. The traffic is EAP on the left, RADIUS in the middle, and MAP/SS7, which is a mobile connection

**Table 2.2** WPA2-Enterprise client setup with EAP-PEAP-MSCHAPv2 and EAP-SIM

Task: (x) = “needed”, (N/A) = “not available”	EAP-PEAP with MSCHAPv2	EAP-SIM	EAP-PEAP with EAP-SIM
CA settings: - choose CA for the RADIUS - if CA-key not known, fetch <i>securely</i>	x x		x x
Other settings: - used EAP-method - validation of RADIUS server’s name - encapsulation (WPA2/802.1X) - password	x x x x	x  x(PIN)	x x
Identity hiding: - enable PEAP - outer identity - inner identity	x x x	N/A N/A N/A	x x

application running over a signaling system (SS7) used in cellular networks, on the right.

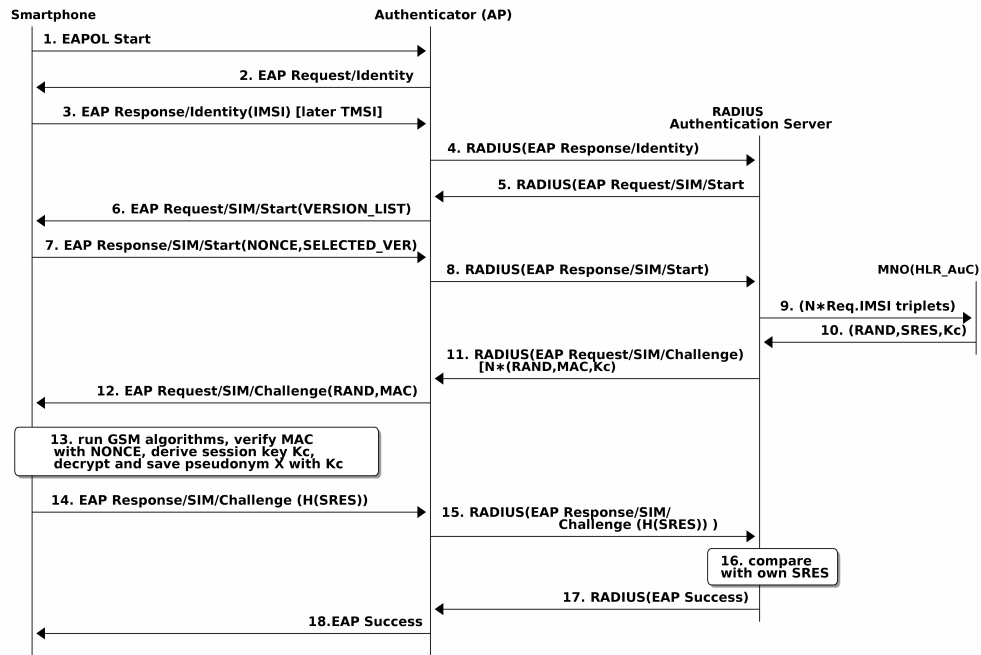
**Figure 2.2** Bird’s-eye view to EAP-SIM components

Protocol analysis of full EAP-SIM authentication is described in Figure 2.3. Important parameters for this work are IMSI, NONCE, and triplet values RAND, SRES, and Kc. From the traffic between the Supplicant (here smartphone) and the authenticator (in AP) we can see that IMSI is first used in message 3. IMSI is the identity which the authentication server would next try to challenge as part of the AuthN and for which the AuthZ would be checked.

All EAP-SIM derivatives provide mutual authentication. An operator (network) is authenticated with the help of a nonce, which is by definition “a number used only once” and can be thought of as a client’s challenge to the network. The nonce is transmitted in the message 7 in Figure 2.3. The client later checks in the process 13 whether the RAND values from the operator were digested with the correct nonce and so authenticates the operator.

The client in turn is authenticated when the authentication server generates a challenge with an aid of a triplet from the MNO and the client responds to the challenge

correctly after processing it with its own  $K_i$ . The correct answer would be SRES which the Authentication server received in message 10.



**Figure 2.3** Successful EAP-SIM full authentication with RADIUS

After mutual authentication, the AuthN phase has been completed. The authentication server completes the AuthZ by sending the authenticator either an Access-Accept or Access-Deny RADIUS message. An accept message triggers 802.1x protocol to open a virtual port in the AP and lets the WPA2 process continue exchanging WPA2 session keys.

Both parties have now retrieved the same trusted key  $K_c$ . The authenticator has received it directly from the RADIUS message 10, and the smartphone has generated it using its own secret  $K_i$  key in the process 13. Therefore the derivation of a secret session key for WPA2 is possible.

After the session has been set, the IMSI may be left out and a temporal IMSI (TMSI) can be used instead to hide the client's identity, for example, in a fast re-authentication case to reduce the risk of exposing the client's IMSI unnecessarily. Unfortunately, at that point, the IMSI has already been exposed at least once in plain text, namely in message 3.

TMSI is composed of a pseudonym and a realm part and can be a string. So, one can send `my-string-which-can-change@...operator.domain` instead of the IMSI



number as an identity. It must be noted that the TMSI used here differs from the TMSI used in 3GPP networks. Those context must not be mixed, otherwise the security that they bring may decrease, i.e. one must not use the TMSI received from 3GPP as a TMSI in EAP-SIM.

## 2.7 Trust

Secure communication has many layers, and on its base lies trust. Only after completing the trust setting phase, it is meaningful to complete the other security layers. For example, secret keys enable encrypted communication, but the keys need to be delivered through a trusted channel first. The same applies to public key infrastructure solutions when verifying the public keys. Thus, we can see that trust really is the first layer to be fixed.

Even without trust, some form of secure asymmetric key-exchange is achievable with Diffie-Hellman key-exchange [12]. Unfortunately, it is vulnerable to Man-in-the-middle (MitM) attacks, where the protocol does not notice if messaging has gone through a third party impersonating as the corresponding messaging partner to both ends. MitM can read and decrypt encrypted messages and forward the possibly changed message with a correct-looking signature. With trust set between two devices, i.e., if they can securely authenticate each other, secret communication is achieved. Secure network configuration and credential exchange is then possible.

This trust can be used to include other components under the same trust circle in the home network. As mentioned earlier, SIM and the MNO trust each other; hence mutual authentication between them is possible. That is later shown to be an important factor. Also the key distribution problem mentioned in Chapter 1 is solved already at a SIM card distribution phase. As AuthN-AuthZ at home proceeds through the authenticator, the authenticator must use it as a derivation function to extend trust and deliver this information further.

## 3. HOME NETWORK ARCHITECTURE

### 3.1 Home network architecture and IETF

While a home network is any network located at a person's home consisting of devices and their connections, either wired or wireless, this thesis avoids using the term *homenet* in that context because *homenet* is reserved to Internet Engineering Task Force Working Group's (IETF WG) *homenet*. IETF is responsible for most internet technology standards, and WG *homenet* was started in year 2011. The current drive in *homenet* management is towards the IPv6 environment as it fulfills the future routing and addressing needs. Regarding home networks, *homenet* has five issues to solve: service discovery, network security, prefix configuration for routers, routing management, and name resolution [1]. As old technologies cannot be forgotten, home networks will be heterogeneous, having both old and new technology, and their interoperability is an important issue when planning future home networks. Segmenting a home into multiple subnets will also be part of planning *homenets*; for example, the home network can include areas for home members, guests, and management. It will not be uncommon to have an inexpensive second network operator subscription for backup purposes at home. Those issues are discussed in the multihoming parts of *homenet* WG. Lastly, end-to-end access, i.e., restriction-free access is in IETF's agenda. It was the key element for the Internet's success and enabled many new applications in the past, but has since then faced difficulties because of firewalls and NATs.

Securing a home network and its router's configuration can be done, for example by first limiting access to their administrative ports with static or dynamic extended access control lists (ACL) in routers. To get through administrative ports, i.e., to login and make configuration changes, there exists either an AAA or local authentication. Authorized agents can then make changes, either directly in the device or through some management protocol, such as SNMP or NETCONF (RFC6241 [14]). SNMP has been in use for over 30 years and is well supported in routers. Yet there are multiple versions for this protocol. While earlier versions (v1, v2) did not provide any encryption of messages, version 3 knows, for example, about public

keys and is secure enough when used correctly. NETCONF is a modern protocol and runs over SSH or CoAP protocols, for example.

Customer Premises Equipment (CPE), such as ADSL broadband routers or set-top boxes, connect the customer's network to the operator's network. There are existing protocols for managing CPEs on the border of home network and operator. For example, TR-069 standard [48] for CPEs has been used to implement self-configuration architecture in home networks [37].

RFC7368 [5] from Arkko about IPv6 Home Networking Architecture Principles defines the borders of the home network and states that internal borders in a home network should possibly be automatically discovered. Limiting those borders to a specific interface type would make it difficult to connect different realms locally. The same document continues stating that while a home network should self-configure and self-organize itself as far as possible, self-configuring unintended devices should be avoided, and the home network user should be let to decide whether a device becomes trusted. So, these statements show that the home network environment still needs external configuration even with the proposed automation aids.

## 3.2 Centralization trends in management

Traditionally, configuration management of individual network devices has been done using each device's console or web interface. As the number of devices has increased, it would have been reasonable to rationalize the process by utilizing a central management, not least to prevent human errors for repetitive tasks. The reason, why this has not happened at home, is because network devices there often are too heterogeneous, bought at different times from different vendors and therefore incompatible with each other.

To help in moving the management to a more centrally model, the home network will see the smartphone as a central managing local controller. Usually, home users already have a phone, which can be considered 'smart'. Most smartphones have Wi-Fi capabilities and writing programs for them is possible even with only little knowledge. When we choose a smartphone to be the management point, the other benefits are numerous: a management software can be delivered and updated from the cloud to diverse smartphone types, existing user base having smartphones is orders of magnitude more than in any single organization, and as the most important fact, the trust anchor can be set to the smartphone.

The users already are centrally located in operators' user databases in HLR-AuC. To achieve the management paradigm change to a central configured one, we still need to bridge the home network to that model with a trusted local controller and then resolve the work-flow of change management.

Home network change management itself is mostly excluded from this work. For example, it is desirable, that changes in home network are done only through local controller, not at local device because of synchronization issues, even if synchronizing algorithms such as Trickle [22] were used in home network for configuration propagation. As another example, configuration also includes power level settings of devices to save electricity based on usage profile. For example at nights or when there is nobody home, some devices do not need to be working at their maximum capacity, and details of how this scheduled is out of scope.

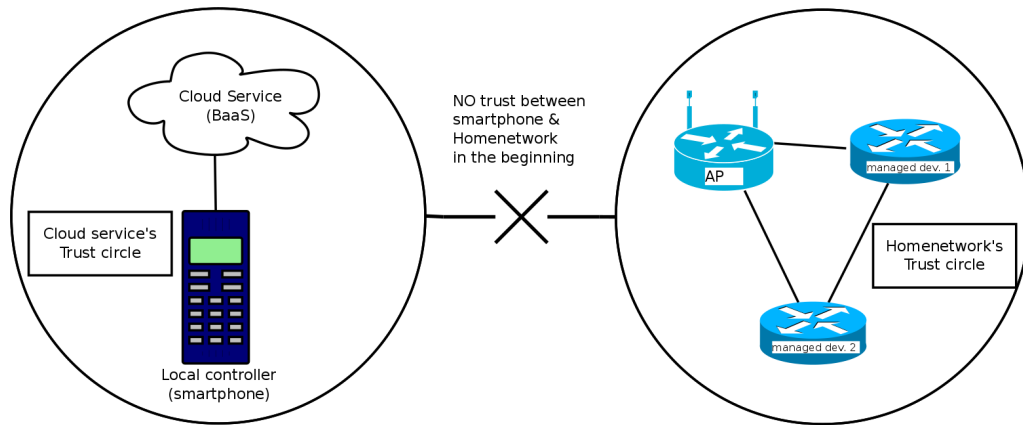
Instead, we study interfaces of AA. Main points here are an existing infrastructure (phones, internet access, Wi-Fi access points), a strong authentication (two-factor), and authentication methods (EAP-SIM, EAP-AKA, EAP-AKA').

### 3.3 Methods for introducing trust anchor into the home network

Trust anchor information, may it then be a secret or some other evidence, can be delivered to a trust device via physical transport channel separate from the actual communicating channel. Traditional way to do that is with a password inside a sealed envelope or a one-time password list that for example online banks use today. The secret can also be sent as an SMS. The trust anchor is part of bootstrapping, which is needed because although the smartphone and the MNO already trust each other, the trust between the smartphone and AP, and thus the management network at home, is non-existing in the beginning as can be seen from Figure 3.1.

In this thesis, the phone brings trust to the home network by completing a full EAP-SIM AA through the local authenticator. SIM's identity is verified by HLR AuC at the phone operator's end and AuthZ added to it later. The verification leaves a trail on the local authenticator and opens a trust channel for a limited period of time for changes from the phone. The disconnection, i.e., the revocation of trust has only been theoretically presented, but not tested in simulated environment.

This chosen model will be fully explained in the next chapter, but before that, we introduce 4 alternative approaches to use SIM's unique properties besides EAP-SIM. Other techniques are Bluetooth SIM Access Profile(Bluetooth SAP), direct



**Figure 3.1** Trust circles in the beginning

connection through PC/SC (Personal Computer/Smart Card), CallerID service from phone network, and Mobile signing service. Bluetooth SIM and PC/SC would need patching of smartphone's software to work. On the other hand, the smartphone would any way need to download a controlling application in the beginning for advanced use, so these techniques could be studied further in another work.

Caller ID as an authentication method uses cellular network's controlling channels. When a phone makes a call, the receiving end gets to know caller's phone number (MSISDN) before it answers the call. That information is called Caller ID and it has been in use successfully for some door locking implementations. It does not cost anything for caller or responder, because after receiving the CallerID information, responder can hang up upcoming call and no call expenses are created. It can also be made safe at least in Finland by limiting which teleoperators are allowed to connect.

SIM card can also benefit from electronic signatures. European Telecommunications Standards Institute (ETSI) has defined a standard for mobile signature services (MSS) in ETSI TS 102 204. MNOs in Finland have diverse implementations for this. The universal service is called "Mobiilivarmenne", but MNO Sonera's brand for it is "Sonera ID" while MNO Elisa calls it "Elisa Mobiilivarmenne".

When AuthN and AuthZ comes from outside, one possibility is to use a federated Mobile AuthN Service, which then is connected to MSSP (Mobile Signature Service Provider) with ETSI-204. Benefits for ETSI-204 federation are similar to those with federation of Wi-Fi groups mentioned in Section 2.2. No home device needs to implement it at home, but also MNO benefits as it sees the service as just one client instead of all possible clients. Without the federation, the mobile AuthN services would need to be multiplied with the number of the separate home networks needing authentication service.

Project Moonshot [27], is in its early phases. Its goal is to enable federated access universally to applications and services. If it worked and was used together with MSSP, it may offer SIM-based SSH access to authenticator. Modifications are then needed both in SSH server and client. Additionally, EAP must be used through tunneling, for example as an inner protocol of EAP-TTLS.

At this point a question might rise, why these external service providers are needed. Is it not easier and simpler to just send an SMS with password code to the smartphone, when access confirmation is needed? Mobile SIM provides two-way AuthN part as discussed earlier. Without need for strong AuthN, that model would indeed be simpler, but using SIM also solves initial key distribution problem. Additionally, mutual AuthN problem would still need to be solved: Who sent that password and where that password should be inserted?

Mutual AuthN is important, because if fake access point were involved, the dishonest admin could lure users to take part in Man in the Middle scenarios. For example, the smartphone user can start the authentication process to the AP which is located at neighbor and think that he is using his own home AP. The malicious neighbor AP (MitM) allows access to the user and starts listening the traffic on AP's network interface. At that time, MitM may capture from the smartphone traffic, which may include sensitive data such as username, passwords, or even configuration data. It is not very difficult to even fake an SSH-server, if the client (smartphone here) does not check the server's changed fingerprint.

## 4. DESIGN OF HOME NETWORK TRUST ANCHOR AND SEPARATION OF CHANGE MANAGEMENT

This chapter describes, how the smartphone becomes a trust anchor for the home network and how the change management happens after that. On its simplest, the smartphone connects with a Wi-Fi link to an AP in the home network and authenticates with SIM-card. The resulting authorized connection brings a trust relationship between the smartphone (a local controller) and the home network (managed devices) anchoring the trust to the smartphone so that the management can happen. In essence, the presence of the smartphone at home opens the gate for the management, though it needs a little interaction on behalf the user.

### 4.1 Chosen AA design

Network can be divided into separate segments based on user's role and needs, such as guest or home members segment. The segments provide base connectivity layer and simple separation. Different services, like disk storage, can force their own policy on application level. It is not defined here, if the segmentation is made physical or virtual (VLAN, Virtual LAN). There is also a segment for devices management. An analogy to the real world would be a hotel, where customers use public access corridors and doors, and service personnel privileged corridors and doors.

Router devices know, how to connect different segments and how the path goes from segment A to segment B. Normally, they can also control access to the segments with the aid of access control lists (ACL), where decision is made based on current configuration or user's role. Decision can happen at the border of network or specific segment.

An example of a deeper access controlling solution would be a traditional firewall and packet inspection in the interconnects of segments, but even more complex and modern model would be the de-perimeterization trend set originally by Open Group's Jericho Work Group [31] in 2004, that will not leave trust verification to perimeters

of network (firewalls and application proxies), but always handles traffic as coming from unreliable source. One implementation of de-perimeterization is Google's BeyondCorp [47], where traffic always travels through Access Control Engine and is suspected as being external, even when it originates from inside networks.

On our chosen model, on the other hand, once a user has been authorized into a management network, access will stay open for him, at least for a (predefined) limited time. So, instead of checking user's credentials each time data is received, this model only checks, from where data is received. Data received from the management network is allowed for changes. It is arguably a lighter method than to always make full AuthN and AuthZ but may suffice here, at first.

When the home network needs a secure link to the smartphone, earlier mentioned trust is the first one needed. The trust is achieved by checking, whether the smartphone can access the home management network using only its trusted SIM-card providing AuthN. AuthZ in turn is compared to existing roles of IMSI in the authenticator.

When AP forwards authentication request to the next RADIUS server, it can ask or receive, beside AuthN and AuthZ, other service parameters, such as provisioning. RADIUS can carry those extra attributes in its ACCESS-ACCEPT message. In essence, AuthZ part itself can be thought as a one type of service provisioning. That would then allow the smartphone to connect to a specific management network and access the devices either via command line interfaces, SNMP, or similar [29, p.4]. There exists RADIUS attribute types for directing user into specific VLAN. If those do not suffice, there are also special Vendor Specified Attributes (VSA). VSAs allow vendors to define up to 255 own attributes that can be used in provisioning in homogeneous environment.

That way (3rd party) authentication server can decide which network segment the device would be put in. In our case, admin users are put in to the management network. Yet, usually RADIUS's ACCESS-ACCEPT message, which means AuthN and AuthZ were successful, puts the user in default network, i.e., it just gives basic access, because not all end devices support those provisioning parameters.

In the first prototype it is enough to identify an authorized smartphone's SIM. The smartphone holding the authorized SIM is granted access to the parts of the management network and it is authenticated strongly. User management is outsourced to the MNO, which already has provided SIM cards to users. What remains, is the adding of the user's IMSI to the authorized users' list. That list can be located on diverse places, as can be seen in Section 4.2.



After authentication and authorization have succeeded, WPA2 session key creation occurs between AP and the smartphone. The authenticator has opened a port to the smartphone for access. It specifically has opened an access to the management network for the configuration changes. The local RADIUS (if existing) and AP have trail of a successful authentication and they know which IMSI has successfully authenticated in the home network. They also know the mapping between IMSI and temporal TMSI for cases when the smartphone later would need re-authentication.

Even though the AP now has authenticated the smartphone user, the managed devices still need to have their own access control. They may consult a local RADIUS server, which tells whether there currently is an authenticated smartphone present and then the changes going to the management network would be allowed. Smartphone could also have received the login credentials to devices through earlier configuration information and use them after getting into management network. This kind of AA would happen on upper, application layer (layer 7), while AA discussed here (802.1X) happens already on lower, media layer (layer 2).

## 4.2 AA component location scenarios

The AA components AuthN and AuthZ can be found in diverse location combinations depending, who provides those services and are there any caching available. Table 4.1 represents the non-inclusive list of locations of AuthN and AuthZ in 5 scenarios. The locations in the table are marked as (i) for internal or (e) for external and the scenarios are described in detail after the table. Authenticator is the entity which gives the final decision about access regardless of the location of AA and it is always internal, located at home network.

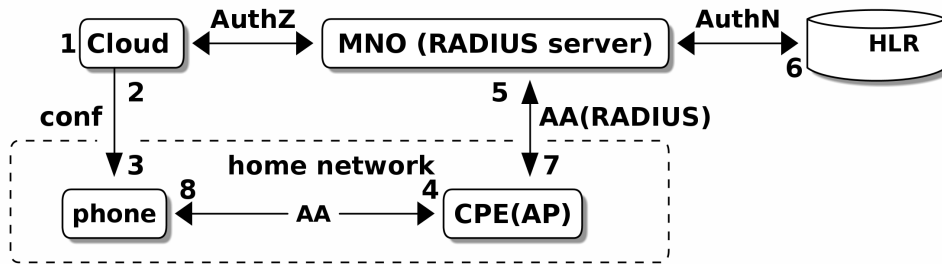
**Table 4.1** Location of AA, AuthN and AuthZ in scenarios I-V

scene no:	AuthN	AuthZ
I	e	e
II	e	i
III	e	i/e (proxied)
IV	i	i
V	-	-

Here, the AuthN component is usually located outside the home, as the MNO provides AuthN, unless we are considering offline or recurring AuthN (in Scenario IV). The AuthZ component may be placed more freely. It can be at the MNO (I), at home (II-IV), or at the external provider (III). Authenticator on the other hand must stay at home, and it always gives the final decision about the access. If the AuthZ

decision is made on remote, 3rd party AuthZ server (I,III), then that server needs to have either local AuthZ data or access to cloud service's AuthZ data. Further it seems inevitable, that delegating AuthZ function would simplify home network management. Because the cloud already has AuthZ data of eligible IMSI accounts, then instead of putting logic on CPE for AuthZ, CPE could just trust the 3rd party service's AuthZ message, which in case of RADIUS is either *ACCESS-ACCEPT* or *ACCESS-REJECT*. The last scenario (V) represents the case, when the environment has not yet been bootstrapped so neither AuthN nor AuthZ is ready yet.

The first AA-scenario is presented here thoroughly as an example. The goal is to make the smartphone trusted to the home network and later proceed to trusted configuration change. The steps are numbered and explained in detail in Figure 4.1. The configuration change is allowed, if CPE gets ACCEPT message from MNO. Allowed users have been inserted earlier to the MNO from the Cloud.



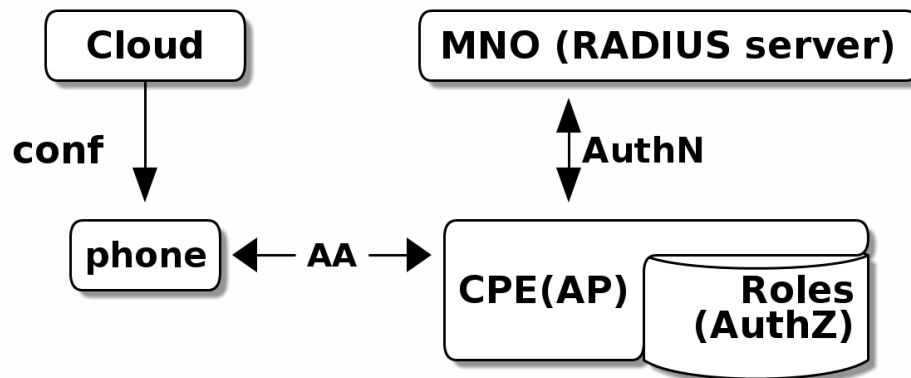
**Figure 4.1** Scenario I with 3 separate domains: Cloud, MNO and home net

1. The model has been changed in the Cloud.
2. The Cloud sends the changes to the Local Controller (phone).
3. If the changes are privileged, they need to be approved by the phone user. The phone user must authenticate to the management network.
4. The phone user starts the authentication process to management network using EAP-SIM and reveals its IMSI.
5. A CPE (AP) forwards the authentication to an MNO's RADIUS server using RADIUS protocol.
6. The MNO has RADIUS server which uses a HLR-AuC for authentication triplets. This RADIUS continues the authentication process until to the end. The MNO also asks (AuthZ phase) from the Cloud, whether IMSI user has

an admin role. The MNO returns in a RADIUS message either *ACCESS-ACCEPT*, if user is both known AND has admin role, or *ACCESS-REJECT*, if either property fails.

7. The CPE receives this ACCEPT or REJECT. If there were other RADIUSes between the CPE and the MNO, they would have acted as proxy RADIUS servers.
8. IF ACCEPTed, then the smartphone is both authenticated and authorized and it now can send configuration change messages to the CPE, which recognizes them coming from an authorized network.

In the second scenario (Figure 4.2), AuthN is asked from an MNO but AuthZ is checked from a local database. Local data comes from a data model, i.e., from the configuration data and it will be saved in CPE, or some other place within the home network. Benefit of local roles is that administrative users can be held at local base. Even, if some intruder would get positive AuthN result, the AuthZ still would deny him the access. When we think, that local role cache only involves few IMSI numbers, this is administratively tolerable.

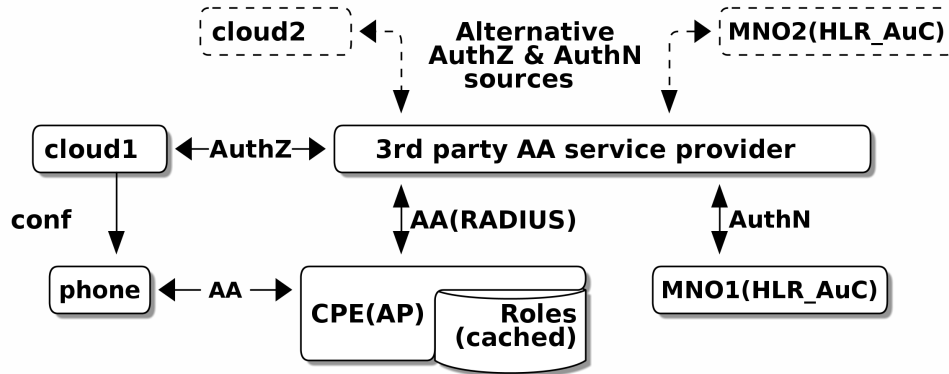


**Figure 4.2** Scenario II with AuthZ in home network

Similar to the first scenario is scenario III (Figure 4.3), but this time there is a service provider between the CPE and the MNO, so AA is fully outsourced: the local AP communicates with RADIUS protocol to the external authentication server. That in turn gets AuthN from the MNO via its own HLR-AuC gateway and AuthZ from the cloud. It can also use alternative sources for AuthN.

Locally there is a cache for roles in case of network disconnectivity.

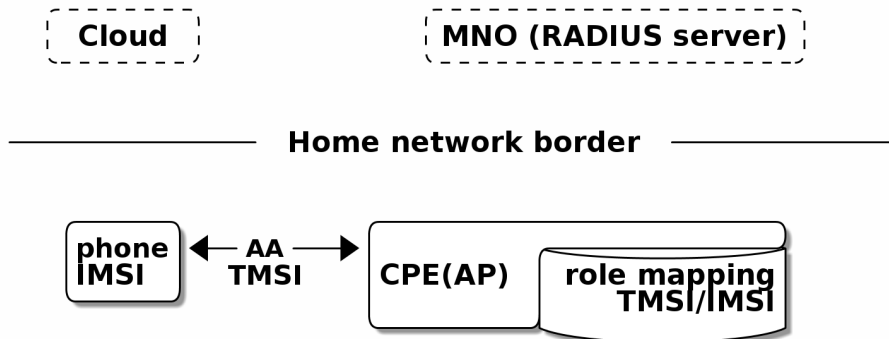
Here benefit is, that 3rd party authentication server may have direct contracts to many alternative MNOs, so the user is free to choose its MNO operator. As a bonus, the MNOs already delegate requests to right operator, if they happen to get AuthN request which does not belong to them. This is similar to federated service.



*Figure 4.3 Scenario III with outsourced AA and backup cache for AuthZ*

Allowed users are verified from the cloud's registries and specific IMSI is authenticated at the MNO. It may need some preparation, if SIM identities are temporary, i.e., TMSI is used. Still, IMSI is carried out at the first message of full authentication. Later, the server would need to have mapping between IMSI and TMSI, but because only full authentication is used, there should be no problem.

If internet connection is down, local AA is still possible, if there has been at least one full AA round. Full authentication uses IMSI, which is the identity of the phone's SIM. Fast re-authentication on the other hand would use temporal identity TMSI, which is lighter in operation than full authentication, and is shown in Figure 4.4.



*Figure 4.4 Scenario IV, offline AA*

TMSI can change each time the AuthN request had been sent. Mapping of TMSI is cached on the authenticator and the round-trip and handling at the HLR can so be eliminated. When the smartphone makes a re-authentication request with its temporal TMSI value, the CPE would still know the the right mapping to IMSI authorization so this works even when there is no internet connectivity, but full authentication would not work here then because of the missing MNO.

Scenario V represents the case when nothing has been configured in network, there are no admin users, and APs are set to factory default. CPE has first neither trust nor roles. Scenario therefore needs pre-configuring and bootstrapping after which it can become any of scenarios I-IV. Bootstrapping is discussed later in Section 6.7.

### 4.3 Modification of RADIUS messages

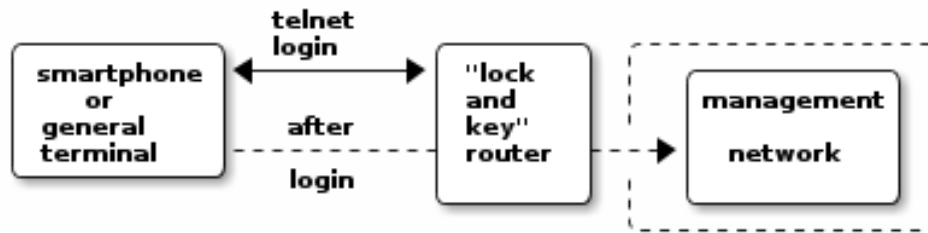
Our model would greatly benefit from the modification of RADIUS messages in proxying RADIUS, if that is possible as was mentioned in Section 2.2(RADIUS). The modification is needed, when the proxying RADIUS wants to combine AuthN message from the MNO to AuthZ decision received from elsewhere.

RADIUS servers may implement different vocabulary in their AVP set. RFC6929 [10] reminds, that even when the RADIUS proxies do not understand all AVPs inside RADIUS message, they must deliver those attributes and that allows us to use larger set of AVPs than is in any (proxying) RADIUS server's vocabulary. By adding AVPs inside the authorization packet, we achieve extra information about validity of the access request. That information may include a VLAN parameter or a time stamp for a forced logout. RFC2865 [39] says, that the forwarding RADIUS proxy may alter the packet as it passes it, but because an alteration would invalidate the packet's signature, the proxy has to re-sign the packet. In the end, proxying RADIUS can technically insert some data into bypassing messages.

### 4.4 Similarities with Lock-and-Key method

The method is very similar to the concept used on routers to dynamically enable access to certain parts of network by first letting the user to log in to the router. If the access succeeds, the router dynamically adds route to the management (or other restricted) part from the users network.

Device provider Cisco calls this "Lock-and-Key" access and uses dynamic access list to implement it [33, p.117]. Lock-and-key is presented on Figure 4.5. Smartphone

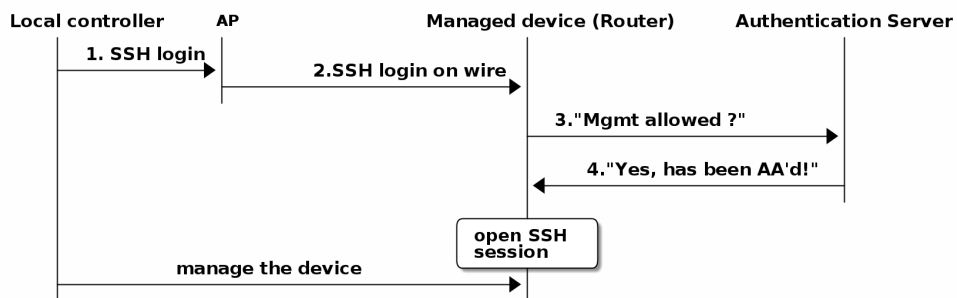


*Figure 4.5 Cisco's view of Lock-and-key access*

has only limited access to the network before AA has completed, while in the Lock-and-key the other parts of network are already open and successful login to the router opens access to even more segments through it. In other words, Lock-and-Key protects IP-access in layer-3 and though needs IP addressing, while 802.1x's protection starts already at layer-2 between the smartphone and AP. Captive portals are similar to Lock-And-Key.

Both methods, 802.1X and Lock-and-Key (and captive portals) can have RADIUS as an authentication server. When RADIUS is not available, for example because internet is down, there almost always exist as a failover a local password method in the configurable router.

This thesis suggests a mix of these methods: EAP-SIM 802.1x WPA2 for authenticating and encrypting in the local network with SIM and Lock-and-key type modification in the AP to further access the management network. Finally, RADIUS protocol is used to transfer parameters, that the smartphone would need in communicating with devices in need of changes.



*Figure 4.6 Managed device trusts Authentication Server*

1. Smartphone connects a Router via wireless AP and needs to login

2. Smartphone uses telnet (or ssh) to login to the ROUTER. with credentials.
3. ROUTER(as RADIUS client) checks AA from Authentication Server(or proxy)
4. AA-server answers based on earlier SIM-authentication that this request is correct

## 4.5 Chosen management model

The simple way to propagate changes from the cloud (or any outside source) is to make them come from the trusted place, here the smartphone, where an application takes care of sending them right to the end devices. This simplification has some pitfalls. If the smartphone continuously stays in the management network, then the changes coming later may not be separated from the currently approved. If we understood that the change approval belongs to the AA-process, then the later approvals would also need an AA.

The smartphone should be dropped out forcibly from the management network either right after the changes have been sent or after a predefined timeout period during which more changes can be send. That period can be called a management session. Timeout method was the original idea and should be kept in mind when the final implementation is made.

The session time and the logout can be handled in AP directly with an timer. After a specific time, AP simple drops the connection (WPA2-session) to the phone. This needs modification to the AP software, if there is not already such method.

Other solution would be for AP to listen for a command from an external AuthZ server, where similar timer would trigger a notification event. That also needed modifications; a listening process to the AP and the timer into the AuthZ server.

Later it was learned [39], that terminating a session is not included in the original RADIUS protocol. The root cause is, that messages originating from the RADIUS server are not defined in the RADIUS protocol and so AP as RADIUS client cannot receive RADIUS server initiated disconnection messages. Additional extensions such as Disconnect and Change-of-Authorization (CoA) packets, also known as RADIUS Dynamic Authorization or RADIUS Disconnection Message(DM) [9], have later been brought in to the protocol by diverse vendors, but they may not all be implemented on every device. Disconnect-Request is sent to UDP port 3799, so authenticator should listen also that in addition to RADIUS UDP port 1812. As a side note, Diameter protocol would provide server initiated messaging.

To demonstrate how the model works, we present the case of adding a new admin user. Let's first suppose, for case of simplicity, that the home network has been already configured(bootstrapped) and it is functioning properly. To further make the case simpler, let's think, that the Local Controller is the sole administrative source for the configuration changes, and leave the cloud outside, because we know that there already exists a configuration framework in the cloud.

Now, let's think what happens, when the cloud operator (or owner of home network) tries to modify attributes, which give access to a new actor, such as a new operator, who would want to have access to separate segments of home network. First, we need to have that segment separation change approved and after that we want to allow the newcomer account to have access to that segment and only to that. For the first part, which is normal operation, approving would perhaps yet not be necessary, but for the second part we need some checking unless our trust to cloud operator is ultimate.

When CPE of home network is about to input configuration changes which would change the balance of authors or roles, it needs to check that the change is permitted. Permission needs to be asked from a trusted point, here mobile SIM. Instead of that, the CPE checks from its own state database, whether mobile SIM has been given access to management network. This thesis justifies the management and therefore the changes to be allowed, if the smartphone user is currently logged in to the management network. After the AA is ready, the smartphone has all means to connect the configured targets

It must be noted, that the smartphone can already have an association to a non-management network with Wi-Fi. If that is the case, it first must disconnect from there and then connect, i.e., do the AA in the correct management network. This implies disconnection from other services using Wi-Fi link, because smartphones currently have only one Wi-Fi radio available and routing would still prefer Wi-Fi as a default gateway, although possible non-Wi-Fi data link still may stay up and operational.

## **4.6 Disconnecting the local controller and offline changes**

After the phone has been successfully connected to the management network, changes coming from the phone can reach configurable targets, for example routers. To defeat the missing internet connection problem, a simple solution would be sending a one-time password to a predefined phone via an SMS, but what entity would then check that and who would be authorized to send that message? Authenticating



server, which has no internet connection should have a predefined way to check that one-time password received via SMS is correct.

Solution for this could be still using a co-existing captive portal for emergency access. As AP is programmable, it could provide this portal. Alternatively, existing programs such as *ChilliSpot* or *NoCatAuth* could be used as WWW-portals. For that to success, the WWW-portal would also need an open access without 802.1X port based access control.

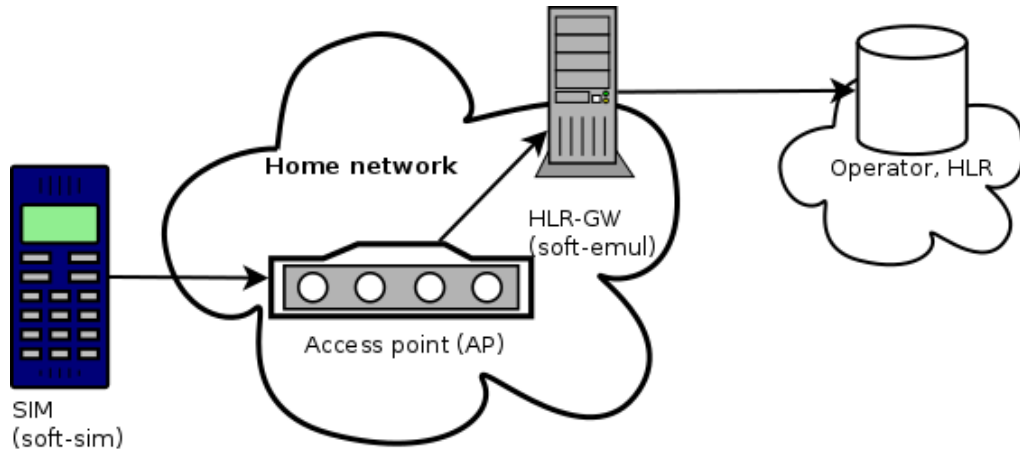
## 4.7 Summary of the chosen solution

The chosen solution to benefit from SIM is via EAP-profiles, as EAP is well known when using WPA2-Enterprise protection in Wi-Fi. Design is based on 802.1X and mobile network operator aided authentication and it is a variation of lock-and-key design.

It is assumed, that the local controller (smartphone) delivers all changes to target devices. The distribution of changes [40] is not further described here. The AP functions both as the authenticator and RADIUS client in scenarios I-IV. Authenticator receives RADIUS messages from authentication server, even when there would be a separate local RADIUS server running as a proxy.

## 5. IMPLEMENTED SOLUTION

To prove that the proposed model works, empirical tests were done. A preliminary plan to benefit from SIM-authentication at home is presented in Figure 5.1. The real operator (MNO) and its HLR were planned to be replaced with a gateway at home network and the real phone with its simulated counterpart. HSS would replace HLR in 3G/UMTS networks. Only the Local Controller's interaction with the home network is tested and Cloud part set away.



*Figure 5.1 Plan to benefit from SIM-authentication at home*

First it is shown how EAP-SIM authentication works in a simulated environment with different clients. This is further analyzed with aid of network traces. In the end, it is shown, that the changes in the management network are possible from the local controller with command line interface (CLI).

### 5.1 EAP-SIM authentication test bed

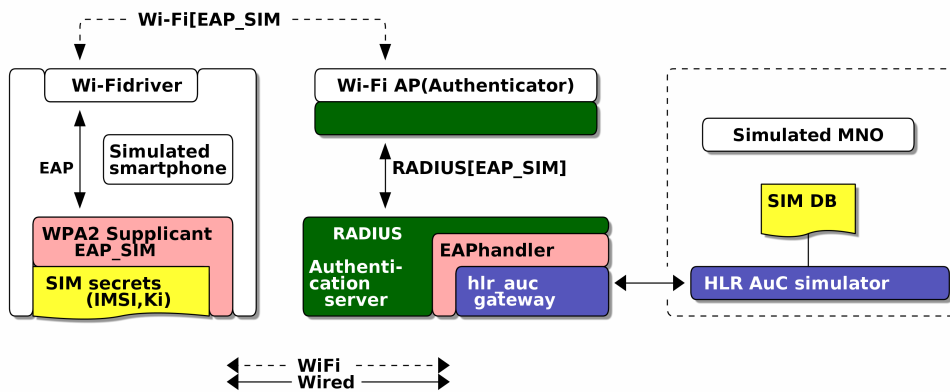
Physical devices used in the tests were three smartphones, a Wi-Fi access point, and a laptop. The smartphones were Nokia E70-1, Nokia E90, and Jolla. The Nokia phones featured EAP-SIM by factory, but the Jolla phone missed a crucial software library and right compilation time configuration set on its WPA-suplicant for SIM.

Jouni Malinen’s software package *HostAP* [26] provides components for WPA2-Supplicant, Wireless Access point (AP), HLR-gateway (for GSM networks), and EAP-endpoint with or without RADIUS-server. From those, executable binary programs `wpa_supplicant`, `hostapd` (for wired RADIUS server), and `hlr_auc_gw` were used. The versions of *HostAP* used in the tests were 2.2 and later 2.3, while version 2.4 was published on March 2015 and 2.5 on September 2015. The software may be distributed, used, and modified under the terms of BSD license.

For a more realistic test, additional hardware AP running OpenWRT firmware was used instead of `hostapd`’s AP. OpenWRT AP worked as a RADIUS client connecting to the RADIUS server still provided by the `hostapd`. OpenWRT AP did not try to open EAP-messages, but just encapsulated them into RADIUS packets. RADIUS server’s configuration file can be seen in Appendix 7.

Laptop’s role was therefore physically split-brain; it asked for AA in the end from itself. Figure 5.2 shows how EAP-SIM AuthN messages (dashed and solid arrowed lines) flow when using simulated WPA2-Supplicant and HLR-AuC as simulation environment.

To make thing more interesting, laptop’s WPA2-suppliant with known software SIM part was later implemented inside Jolla. This was possible thanks to open software and similarity of Jolla’s environment to the one used in laptop. Modification involved compiling PCSC-libraries and recompiling the WPA-suppliant used in Jolla with simulator options for ARMv7 architecture.



**Figure 5.2** EAP-SIM AuthN messaging in simulation test bed

The algorithm used in the demo was an internal GSM-Milenage, which handles EAP-SIM beside EAP-AKA. Milenage is a reference implementation and as such suitable for operators, who do not want to invent their own security algorithms.

OPc and Seq parameters from Milenage were not used, because they are not needed in EAP-SIM.

## 5.2 Detailed description of test runs

The first tests were done with hostapd as a wireless AP. Test run with Nokia E70-1 with Symbian 60 Series OS (2006) had a non-registered SIM card. In testing the authentication, there was no indication of EAP method present in captures, the only indication of security was a message “Open System” in application logs, which only means that no pre-shared key is used. The reason for that was not found, but it may have been problem on hostapd’s settings.

Nokia E90, with a registered SIM had better results. This time the capture was made in monitor mode, i.e., the capture interface was mon.wlan0. Traffic captures revealed some EAP traffic involving some EAP-SIM Request–Response pairs.

*Capture 5.1 E90 trying to use its internal EAP-SIM*

#	time	from->	to	Proto	len	Message
1	1.5588	e90	-> AP	EAPOL	18	Start
2	1.5589	AP	-> e90	EAP	118	Request, Identity
3	1.5607	AP	-> e90	EAP	117	Request, Identity
4	1.8055	AP	-> e90	EAP	72	Request, GSM Subscriber Identity Modules EAP (EAP-SIM)
5	1.8070	AP	-> e90	EAP	71	Request, EAP-SIM
6	1.8053	e90	-> AP	EAP	74	Response, Identity
7	1.8095	e90	-> AP	EAP	106	Response, EAP-SIM
8	1.8055	AP	-> e90	EAP	72	Request, EAP-SIM
9	1.8070	AP	-> e90	EAP	71	Request, EAP-SIM

When opening Responses number 6 and 7, we can see that we got to the part, where a nonce, selected version, and an identity were sent from the phone (message 7 in Figure 2.3, but RADIUS server did not know how to handle them. Even when authentication conversation would not complete fully, authenticator still received a claim of identification from the smartphone. Yet, as there is no full AuthN, no proof of identity existed in that case. Both the unregistered SIM and the registered SIM could not have been verified with the operator, because it was locally simulated and they both therefore correctly ended the conversation as should regarding RFC4186 [18].

At this point, physical phones were put aside and a simulated SIM-card environment also at client’s side was used. After WPA2-Supplicant run on laptop with simulated SIM-card access with SIM/USIM protocols, respective EAP-SIM, logging

revealed that “Hostapd will send SIM/AKA authentication queries over a UNIX domain socket to an external hlr\_auc\_gw. program.” This notified, that Hostapd successfully had received SIM card’s AuthN request and was going to forward that to simulated HLR authentication program. Now it was time to build an environment to help automating the tests. Tests were run from a shell program (Appendix 7), which started the needed programs. It also recorded used configurations, logs, and traffic captures for later analysis.

Wireless capture of traffic between WPA2-Supplicant and AP was made at WPA2-Supplicant on the wireless card. Wired capture between AP and RADIUS server was made at wired card. Wireless capture was not made in monitoring mode, so not all 802.11 details in data packets were captured [51]. That was not a problem, because the focus was in the EAP messaging instead of radio channel details. Additionally, the messages between RADIUS server and HLR were recorded. Whole conversation is given first here together and afterwards analyzed more thoroughly.

Packet capture of successful SIM-authentication with corresponding parts of logs at WPA2-Supplicant, RADIUS server and packet captures 802.1X, RADIUS and HLR is shown below. Communicating partners are denoted as AP and phone for Wi-Fi traffic, AP-wired and RADIUS-srv for RADIUS on wire, and finally RADIUS-hlr and HLR-AuC gw for simulated MAPS/SS7 traffic. The capture has been done at the WPA2-suppliant.

*Capture 5.2 Successful EAP-SIM Authentication*

No.	Time	Src	Dest	Proto	Len	Info
129	7.9830	AP-wifi	phone	EAP	23	Request, Identity
130	7.9832	phone	AP-wifi	EAP	39	Response, Identity
131	7.9887	AP-lan	RADIUS	RADIUS	235	Access-Request(1) (id=162, l=193)
132	7.9889	RADIUS	AP-lan	RADIUS	108	Access-Challenge(11) (id=162, l=66)
133	7.9908	AP-wifi	phone	EAP	38	Request, GSM Subscriber Identity Modules EAP (EAP-SIM)
134	7.9924	phone	AP-wifi	EAP	70	Response, (EAP-SIM)
135	7.9945	AP-lan	RADIUS	RADIUS	272	Access-Request(1) (id=163, l=230)
-		RAD-hlr	HLR-AuC	socket		SIM-REQ-AUTH <IMSI> 3
-		HLR-AuC	RAD-hlr	socket		SIM-RESP-AUTH <IMSI>, 3 triplets
136	8.0024	RADIUS	AP-lan	RADIUS	256	Access-Challenge(11) (id=163, l=214)
137	8.0040	AP-wifi	phone	EAP	186	Request, (EAP-SIM)
138	8.0043	phone	AP-wifi	EAP	46	Response, (EAP-SIM)

139	8.0063	AP-lan	RADIUS	RADIUS	248	Access-Request (1) (id=164, l=206)
140	8.0065	RADIUS	AP-lan	RADIUS	202	Access-Accept (2) (id=164, l=160)
141	8.0110	AP-wifi	phone	EAP	22	Success
142	8.0112	AP-wifi	phone	EAPOL	135	Key (Message 1 of 4)
143	8.0123	phone	AP-wifi	EAPOL	135	Key (Message 2 of 4)
144	8.0161	AP-wifi	phone	EAPOL	169	Key (Message 3 of 4)
145	8.0163	phone	AP-wifi	EAPOL	113	Key (Message 4 of 4)

IMSI is sent the first time already on the second EAP message from WPA2-Supplicant to AP (compare with Figure 2.3, message 2.) This part is presented in more detail in Capture 5.3.

**Capture 5.3** *First detailed indication of IMSI, captured from WPA2-suppliant*

```

Frame 129: 7.983047
  Type: 802.1X Authentication (0x888e)
  Version: 802.1X-2004 (2)
  Type: EAP Packet (0)
  Length: 5
  Extensible Authentication Protocol
    Code: Request (1)
    Id: 50
    Length: 5
    Type: Identity (1)
    Identity:
Frame 130: 7.983223
  Type: 802.1X Authentication (0x888e)
  Version: 802.1X-2001 (1)
  Type: EAP Packet (0)
  Length: 21
  Extensible Authentication Protocol
    Code: Response (2)
    Id: 50
    Length: 21
    Type: Identity (1)
    Identity: 123201000000000000

```

We might notice the difference on used 802.1X versions: AP uses version 802.1X-2004 in its request while the smartphone uses 802.1X-2001. Here it does not have any noticeable effect.

The last line has the important identity field received from the SIM. Its length cannot directly be seen, but when EAP message's length (21 octets) is reduced by fixed space needed for Code(1), ID(1), Length(2), and Type(1), it yields 16 octets

for the identity. Therefore the identity is not coded as a numeral but instead as a string and that brings more flexibility in the protocol as the identity can include alphabets too. It also minimizes misunderstandings, if context gets lost. When we remember from Section 2.5, that IMSI can not be more than 15 octets, the extra prefix '1' has an extra meaning. Here it denotes that we are talking now about EAP-SIM identity instead of EAP-AKA or EAP-AKA'.

EAP client's identity is transformed at authenticator (Figure 2.1, Chapter 2) from 802.1X's EAPOL format into RADIUS format and sent to RADIUS server. The captured frame between AP and Radius server is shown in Capture 5.4.

**Capture 5.4** EAP client's identity transformed, capture from RADIUS server

```

Frame3: 7.988616
Radius Protocol
  Code: Access-Request (1)
  Packet identifier: 0xa2 (162)
  Length: 193
  Authenticator: 055ff370b9e793c1e39d375aade8033c
  Attribute Value Pairs
    AVP: l=18 t=User-Name(1): {1232010000000000 }
    AVP: l=7 t=NAS-Identifier(32): musta
    AVP: l=27 t=Called-Station-Id(30): 66-66-B3-8A-68-B3:simtest
    AVP: l=6 t=NAS-Port-Type(61): Wireless-802.11(19)
    AVP: l=6 t=NAS-Port(5): 1
    AVP: l=19 t=Calling-Station-Id(31): 5C-51-4F-E7-FA-F4
    AVP: l=24 t=Connect-Info(77): CONNECT 54Mbps 802.11g
    AVP: l=19 t=Acct-Session-Id(44): 5491885C-00000037
    AVP: l=6 t=Framed-MTU(12): 1400
    AVP: l=23 t=EAP-Message(79) Last Segment[1]
      EAP fragment
      Extensible Authentication Protocol
        Code: Response (2)
        Id: 50
        Length: 21
        Type: Identity (1)
        Identity: 1232010000000000
    AVP: l=18 t=Message-Authenticator(80):
                                     04ea7e507d72bdb1acf515ef19ac9527

```

Here interesting part is the first RADIUS AVP. While encapsulated EAP fragment naturally carries the Identity="1232010000000000" field, it was surprising that RADIUS has captured that field and filled its User-Name field to the very same, "1232010000000000". IMSI is "2320100000000000" and it is prefixed with an '1' as earlier explained.

In WPA2-Supplicant configuration file (see Appendix 7) both the identity and credential section had the identity field, but the latter creds block was later learnt to belong to cases, where the same credentials would be used to multiple networks and therefore it would save to trouble of always rewriting that block. Another thing to be noted is, that AP has followed conventions on converting EAP into RADIUS message and put identity field into User-Name Attribute Value Pair (AVP) too. Similar convention can be seen when analyzing EAP encapsulation and message size. The last RADIUS (AVP) is Message-Authenticator, which presents limited safety against message corruption. Limited, because it uses MD5-hashing which is not safe against malicious attackers anymore.

Meanwhile, HLR simulator was listening requests from Authentication server's internal EAP-handler through a local socket. Its configuration is shown in Appendix 7 and there one can see the simulated Milenage parameters including SIM secrets.

The AuthN request (SIM-REQ-AUTH), which in production version would go to real HLR-AuC, included the IMSI and parameter "3", which indicates, that the requester wants three triplets. While one triplet would equal 64-bit key used for challenges, three triplets will make the key 192 bit long. Session keys are derived from this keymaterial. Crypto analysis of Patel [34] noted in 2003, that achieved session key security length would not be even 128 but 64 bit, but it is not known to us, whether this was taken in account in 2005's version. Format of triplet received is RAND:SRES:Kc.

```
Received: SIM-REQ-AUTH 232010000000000 3
Send: SIM-RESP-AUTH 232010000000000
a5dc7c1a177ee418:fea4260f:6634b5081c74b5872b49f37fc387ddb5 \
0faa08f223510ef6:e6d0f3f4:3d7559287e5bd2ec3fb77b1f7d097d8f \
832475ad3e7bea2b:3fe28cc8:1be8b4f1ab247ec732d15cf63ad57390 \
```

### 5.3 Simulation on smartphone Jolla

Because we had source code available, we also tried to move the simulated SIM application to the real smartphone. Jolla is a new Finnish smartphone, running on ARMv7 processor, having its roots on Nokia's last mobilephone line that used Linux Distribution MeeGo in beginning of 2010's. The OS is called Sailfish and it includes user interface code made by Jolla, open source components from Mer, and some other software components, which are either free or proprietary. The name Mer comes from MEego Reinvented, or meritocratic community and it is a community driven by merits.



Jolla's base WPA-supPLICANT belongs to Mer core and it was here re-built for simulator use. The build process was done inside MerSDK virtual machine running under Virtualbox, because it provided not only simulated Jolla environment on x86 processor architecture, but also a cross compiling environment Scratchbox, that makes porting programs to Armv7 processor easy.

To compile wpa\_supplicant with EAP-SIM simulator, PCSC component was needed. So first pcsc-lite was downloaded and cross-compiled in MerSDK. Then wpa-supPLICANT was compiled with correct parameters in same place. Finally, the wpa-supPLICANT was stripped from debugging symbols and copied to Jolla together with the PCSC-library. Jolla phone's file hierarchy is very similar to hierarchy in test environment's Linux, which made the operation very easy. Table 5.1 describes the components and their versions that were used in the tests.

**Table 5.1** Versions of components in tests and their architecture.

version(s)	component	tested on (machine arch)
2.3	hostap(RADIUS-EAP)	laptop(x86_64)
2.3	hlr_auc_gw	laptop(x86_64)
2.2, 2.3	wpa-supPLICANT	laptop(x86_64)
2.4	wpa-supPLICANT	Jolla (armv7l)
1.1.9.28	Sailfish OS	Jolla (armv7l)
1-8.13	pcsc-lite	Jolla (armv7l)
1-8.13	pcsc-lite	laptop(x86_64)

### 5.3.1 Cross-compile the programs inside a virtual machine

MerSDK is shipped as a virtual image which runs inside Virtualbox. Starting the image opens an ssh server, which by default listens on port 2222 on localhost. One can use an SSH keypair authentication to get there. In following example, file *mersdk* represent the private SSH key of type RSA.

```
# ssh -l mersdk -p 2222 \
-i ~/emu/sailfish/vmshare/ssh/private_keys/engine/mersdk
localhost
Last login: Sun Nov 15 09:50:47 2015 from 10.0.2.2
[mersdk@SailfishSDK ~]$ uname -rm
3.6.11-10.1.17.jolla i686
```

As can be seen, the virtual machine's architecture is i686, i.e., 32-bit x86. We moved to sandboxed armv7hl architecture (armv7hl) with sb2 command and changed the

user id to a virtualized root user. Earlier retrieved pcsc-lite package was compiled and installed.

```
[mersdk@SailfishSDK ~]$ sb2 -t SailfishOS-armv7hl -R
[SB2 sdk-build SailfishOS-armv7hl] root@SailfishSDK ~ # uname -rm
3.6.11-10.1.17.jolla armv7l
[SB2 sdk-build SailfishOS-armv7hl] root@SailfishSDK ~ # cd \
/home/mersdk/rpmbuild/SOURCES/pcsc-lite-1.8.13
[SB2 sdk-build SailfishOS-armv7hl] root@SailfishSDK ~ # ./configure ;
[SB2 sdk-build SailfishOS-armv7hl] root@SailfishSDK ~ # make install
```

The wpa\_suppllicant was compiled similar way. The needed option were added to configuration file before compiling.

```
CONFIG_EAP_SIM=y
CONFIG_SIM_SIMULATOR=y
CONFIG_USIM_SIMULATOR=y
CONFIG_PCSC=y
```

After wpa-suppllicant was compiled and linked, it was stripped from debugging symbols to make its size to merely 15% of the original size.

```
# ls -s wpa_suppllicant-pcsc
5788 wpa_suppllicant-pcsc
# strip wpa_suppllicant-pcsc ; ls -s wpa_suppllicant-pcsc
876 wpa_suppllicant-pcsc
```

Finally, programs wpa\_suppllicant-pcsc and corresponding libpcsc-lite.so.1 library were copied to the jolla phone, which was connected to the computer with an USB cable.

```
scp wpa_suppllicant-pcsc nemo@jolla:
scp libpcsc-lite.so* nemo@jolla:/usr/local/lib
```

### 5.3.2 Running the new binary on real smartphone.

We now could use inside the real physical smartphone this modified wpa\_suppllicant, which uses a configuration file instead of a SIM card. To make sure, that it

does not conflict with an existing `wpa_supplicant` on the phone, we first disabled the system version and used our version manually.

```
% ssh nemo@jolla
Last login: Sun Oct 11 17:44:37 2015 from 192.168.2.1
,---
| SailfishOS 1.1.9.28 (Eineheminlampi) (armv7hl)
'---
[nemo@Jolla ~]$ systemctl disable wpa_supplicant
```

Because the *libpcsclite* libraries are in non-standard place (in `/usr/local/lib`), we had to start `wpa_supplicant` with the correct dynamic library path environment value otherwise an error occurred.

```
[nemo@Jolla ~]$ export \
                    WPASUPP=/home/nemo/wpa-pscs/wpa_supplicant-pcsc-stripped
[nemo@Jolla ~]$ ${WPASUPP} -v
./wpa_supplicant-pcsc-stripped: error while loading shared libraries:
libpcsclite.so.1: cannot open shared object file: No such file or
directory
[nemo@Jolla ~]$ LD_LIBRARY_PATH=/usr/local/lib ${WPASUPP} -v
v2.4 (malinen 2003-2015)
```

From here the same tests could now be run on the smartphone that earlier needed the computer. Even now, we did not use the real SIM card, but it probably would have been possible, if the configuration file did not give the secrets but instead a PIN to unlock the smartcard.

## 6. ANALYSIS, RESULTS AND DISCUSSION

We have now gone through theory, design and implementation so let's discuss some results and findings. We concentrate on security, deployment difficulties, costs, platform issues, usability of multiple APs, and cases where we need to extend the number of SSID's. The facts collected in the thesis are once more wrapped up in Discussion section. Last, we speculate on some alternative design solutions for adding trust and management models.

### 6.1 Security considerations

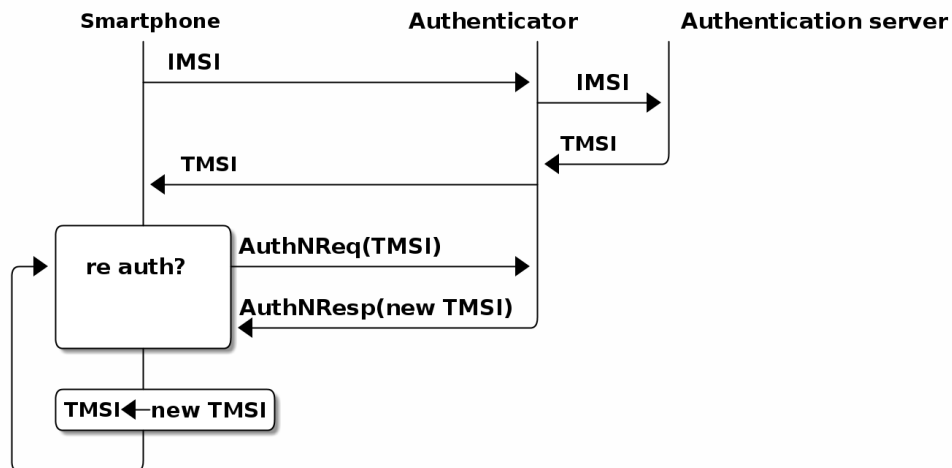
There can be multiple ways to attack the described methods of the home network management delegation. Following subsections divide them into confidentiality (privacy), integrity, and authenticity. Accessibility is also discussed.

#### 6.1.1 Confidentiality (privacy)

Confidentiality means that no one else other than message's intended recipients may get to know information inside the message. The purpose of the message confidentiality in authentication phase is to hide the identity of the smartphone and possibly the delivered secrets from eavesdroppers. Hiding IMSI enhances the privacy of the smartphone user.

Recall from Section 2.5, that IMSI is sent in clear during the starting phase of 802.1X authentication and that is a privacy issue, because TMSI which hides IMSI cannot be used before a session has been set up. [18, p.66]. After the first full authentication, the client and the authenticator know TMSI and can use it in further communication. TMSI can even be re-changed using re-authentication as shown in Figure 6.1.

The authenticator is responsible for converting TMSI to IMSI if it later needs to ask for full authentication from the MNO. During that time, IMSI can be caught using the device called IMSI-catcher. The very same happens also in a regular GSM



**Figure 6.1** Changing TMSI without the authentication server (or MNO)

network with non-EAP traffic. IMSI can be caught by listening the GSM network for phones that are registering themselves to the operator when they are powering on. The fault lies there, that GSM specification does not require the mobile network to authenticate itself to the phone and so GSM allows man in the middle attack. The attack follows, when the IMSI-catcher impersonates itself as a (cellular) base station. When the smartphone tries to attach to the fake base station, the smartphone reveals its IMSI number. Further, because the base station is responsible for chosen encryption, the base station can order the phone to not encrypt traffic at all or to use only weak encryption thus revealing all data, calls, and texts. Mitigation for IMSI-catching would be to disable GSM (2G) usage altogether from phone if that is possible[42]. Some development has been done to detect IMSI-catchers, most notably by the project AIMSICD [4].

Most EAP methods themselves do not provide identity protection, i.e., the end-user hiding. This feature can be achieved with PEAP (Protected EAP) or TTLS, which chains different EAP-methods together and protects the inner EAP with an outer EAP. The outer identity tells just the realm, where AuthN can be checked and inner identity reveals the real identity. The inner identity is encapsulated inside the outer identity which functions as an envelope. For example EAP-MSCHAPv2 (Microsoft's Challenge Handshake Authentication Protocol, version 2) can be used inside PEAP.

EAP-SIM would provide identity protection, if it was used together with PEAP. There PEAP would anonymize the outer identification and EAP-SIM was used in inner authentication. Currently, it is not known for the author that such imple-

mentations exists for EAP-SIM except Tseng's proposition [44] for a new EAP type EAP-USIM, which would extend EAP-TLS type.

If it was possible to use anonymous identity on outer EAP authentication, then EAP-SIM AuthZ must also be done at HLR AuC. AuthZ cannot else be connected to the corresponding identity and AuthN itself is not enough because it only defines the users' authenticity, not their admin roles and so AuthN should work for any smartphone that has existing contract with their MNO. It still is the responsibility of the authenticator to check AuthZ and let only admin mobile access the management network.

If SIM is used as the only EAP without EAP-PEAP, then there is no mitigation for revealing the IMSI on the first message and it leads to privacy issue. If there was an IMSI-catcher involved, only IMSI would be revealed. The other parameters or encryption are not in danger in EAP-SIM authentication, because EAP-SIM will stop conversation, if it does not receive the correct MNO authentication message. EAP-SIM protocol, as do most of the other EAP-variants, provides a secure way to generate session parameters to WPA2-session and those are not leaked outside, because they are created individually on both endpoints: at the smartphone and at the Authentication Server. Finally, the fact that the secret key  $K_i$  stays inside the SIM makes it difficult to attack the session by pretending to be the smartphone.

RADIUS messaging in the wire is vulnerable too, because IMSI is transmitted in clear in EAP. IMSI is copied in to RADIUS's user-name attribute from EAP's identity field. RADIUS runs over UDP, but can be used also with TCP. With TCP, RADIUS may use RadSec extension, which features TLS-based message encryption, integrity checking and more advanced digesting functions than plain RADIUS. RadSec is also expandable and open for negotiating more secure ciphersuites that future versions of TLS might require [49]. This is more and more important today, as it already has been noted in 2005, that even SHA1 digesting has shown its weaknesses and can be manipulated. RADIUS servers usually support RadSec extension, but APs do not, so the part from the AP to RADIUS server will travel in plain unless secured otherwise.

Based on those facts, EAP-SIM cannot be considered confidential for identity during the first message exchanges, but later the identity can be hidden using temporal identity (TMSI). For other parameters, EAP-SIM is confidential. Thus, this paper considers TMSI only when using it for recurring authentication (re-auth) and offline cases in the implementation Chapter 5.

### 6.1.2 Integrity

The integrity issues were partly handled in Section 4.3 describing RADIUS message modification. Message digestion codes provide integrity for RADIUS protocol. If PEAP is used, it handles integrity through its usage of TLS [32].

There is also a fraudulent authenticator problem, which is an attack against both the integrity and privacy. The authenticator may present some information to the authentication server and other to the EAP-peer. Mitigation for that is, that EAP-peer would include some characteristics of the authenticator inside its EAP-message, which the authentication server then verified (RFC6677) [16].

### 6.1.3 Accessibility, DoS and scalability

Is home network immune against (distributed) denial of service (DoS) attacks? Besides DoS, does the solution scale up from home network to small and middle size companies? To answer this we can remember that backends (cloud and operator) are designed for thousands or even millions of users, so they hardly are limiting factors. Instead, local authenticator is the one whose performance might suffer, which comes from processing loads [23].

Traditionally, RADIUS has used connectionless UDP protocol that is light weight. Retransmission in UDP is tolerable, because the user is ready to wait several seconds for authentication to complete. Today, RADIUS can be also run over TCP, which has generally more aggressive retransmission rate. [30, Section 2.2.1]. UDP misses the reliability that TCP has, but, adding an alternative UDP RADIUS server can answer requests faster than waiting for TCP RADIUS's reliable delivery.

### 6.1.4 RADIUS weaknesses and strengths in limited use cases

RADIUS protocol itself is old and not very secure as of current standards(2015), because messages are not encrypted and they are transported on datagrams (UDP). Alternative RADSEC protocol uses TLS, and is backwards compatible with RADIUS protocol, so it can be used as secure RADIUS proxy such as *radproxy* [46].

RADIUS protocol provides some integrity checks with Message Authenticators as described in Section 4.3. RADIUS uses MD5 hashing and shared secrets, but that is not enough. MD5 hashes were first time broken by brute force already 20 years ago and today they can only be used as a data error detection tool [45, p.2]. If

a malicious actor (Man in the middle, MiTM) imitates RADIUS proxy, it can try to inject false messages. Duplicate MD5-hashes (collisions) are easy to compute today (for example MD5Attack [9]) and researchers like Xie in 2013 [53] are even accelerating the speed for computing.

Because of the weaknesses of MD5 hashing, the transport needs additional protection like tunneling or IPsec. TLS can be used for encryption and its signatures for integrity checking of packet payload.

In scenario III(Figure 4.3), there was a proxying RADIUS between the authenticator and the MNO. When the MNO notifies the authenticator that a smartphone has been authenticated, then the authenticator (AP, functioning as a RADIUS-client) hooks that message and usually just grants the smartphone the access to the network. After giving access rights, other provisioning parameters can be sent with RADIUS messages, for example a session time-out, a current admin user list, a state of OTP list, or a VLAN id.

### 6.1.5 Replay, re-use, re-auth, and brute-force challenges

Earlier, in RADIUS analysis, prevention of replied messages was mentioned. Reusing the same secret in different security context is also considered bad because mixing secrets between usage domains weakens the security. In GSM networks, IMSI identifies the subscriber on the first contact, later TMSI is used for call and SMS. In EAP-SIM those values are also used. IMSI naturally is the same, but TMSI should be different for call and EAP. Haverinen [19] explains how special RAND numbers can be used to differentiate the use of TMSI in 3GPP and Wi-Fi contexts.

Re-authentication and termination can bring unexpected results. Even, when sessions can be terminated, the client side may have option set to login automatically, transparent and without users control. Automatic re-authentication after disconnection must be considered here as harmful as well as automatic login when nearby suitable AP. An example from harmful behaviour was a case with Swiss mobile operator Swisscom, who provided two networks “Mobile” and “Mobile Eapsim” for its customers. The latter network did not ask each time customers’ permission for connection but used their smartphone’s SIM automatically. Unfortunately, it also charged users for using Wi-Fi connections without users’ knowledge. [25]

Although a direct connection to SIM card’s memory is restricted, if one can read and write data through SIM card’s API, one could still try to get information (SRES, Kc) by brute-force. Fortunately SRES and Kc are never sent in clear, but inside a



digested MAC. Additionally SIM card can be programmed to answer only limited number of challenge request, for example 65535 times. For normal use that would be enough, but in brute-force challenges it would soon be exhausted and SIM card would lock itself and not function anymore at all.

### 6.1.6 Hardware tampering

All this time it is assumed, that hardware does not lie. In case the hardware has been tampered, we could not trust its claims. For example, there have been attacks against SIM to reveal its private key after SIM have been copied. To verify, that a device has not been tampered, a method called attestation can be used.

A device, which has attestation capability such as hardware certificates or Trusted Platform Module (TPM) technology can function as a trust anchor. Such a device could be sent direct to customer with pre-configured secrets and methods to take a place as a trust anchor. That leads us again to the key distribution problem.

### 6.1.7 Mitigation methods against radio capturing

To mitigate risks for radio capturing, two methods are presented: hiding of wireless network and proximity. They are not perfect but can limit attack vectors in time and place.

Recall that the access to management network from the smartphone is needed only then when changes are made. Why then not just enable management radio network during those times? Then there were less networks for users to choose from. Enabling management network could be programmed through LuCI-interface, which is a web user interface to the Unified Configuration Interface used in OpenWRT routers. Preliminary tests showed, that activating new networks in AP also disconnects existing Wi-Fi connections and may even restart AP, which certainly would not be wanted. Some other methods need to be invented to avoid denial-of-service, when an intruder tries to connect by that method and would cause continuous AP outages.

One could also think of hiding the network by disabling the advertisement of management network SSID. That is called “network cloaking”. Smartphone would then need to know the exact target SSID name. Does disabling or hiding the management network bring real security or is it just security by obscurity? Security by obscurity means here, that hiding network as a security method would filter out only some

casual crackers, while at the same time it still is trivial for any serious crackers. Disabling or hiding merely gives one security layer more so it is not a real security method.

The SSID could also be renamed always, in essence to implement one-time-only network, but then the smartphone would need to get that secret somewhere, perhaps via an SMS and that would defeat the purpose of easy access. If that method nevertheless would be used, then hiding the one-time-only network name actually could add security. If the client knows beforehand the name of SSID (and maybe also checks AP's MAC), then AP does not reveal any information, before the client has tried to connect to it and that would minimize the time window for attacks. Because RFC5448 [7, p.12] forbids the change of network name between full and fast authentication to prevent usage of compromised keys, the client and server should use full authentication, when they notice, that network name has changed.

Hiding can also have privacy enhancing effects. While Wi-Fi client's normal action is to probe for SSIDs of lately visited and learned APs, anyone can analyze those probes and reveal client's earlier locations without very much effort. Lindqvist et.al. [24] present usage of hidden APs in protecting privacy of clients and preventing that scenario.

Regarding boundaries of home network, the Wi-Fi coverage gives one natural limit, which is 50 meters indoors or 100 meters outdoors, when no extenders (Wi-Fi repeaters) are in use. Proximity brings a minimal extra layer just like network cloaking for preventing attacks, as the attacker must be physically within those limits. This can be considered as an added factor in multifactor authentication or reputation, but it will not be enough, because attackers will have more sensitive radios available than normal users devices have. Also, if SIM-profile was used through Bluetooth, there were also range limits, but even shorter. Despite the claimed distance limits on Bluetooth, receiving can be extended with even to one mile with directed antenna [52].

## 6.2 Deployment difficulty and costs

To deploy the system, modifications must be done to the AP and the client. Additionally, a contract must be made with the MNO service provider producing AuthN.

Requirement for home network can be as small as having WPA2 Enterprise capable AP. Almost any AP will do, but as an exception, cable modem Bewan, which has been distributed to many homes from the cable modem operator Elisa, was found to

have only WPA2-PSK mode. WPA2-enterprise setting involves setting IP-address of RADIUS server providing AA and corresponding shared secret. For the client, a Wi-Fi profile must be added: used management SSID, protection mode 802.1X (or WPA2-Enterprise), and AuthN method EAP-SIM. Smartphone can have different profiles, also with a same SSID, but then the user needs to choose manually, which profile is wanted.

Additionally, the initial admin user's SIM-card has to be registered as an admin user in some way in home network configuration, i.e., IMSI must belong to the admin group. In this implementation, no extra application is needed in smartphone for primitive trust, but later for more serious use some application is needed. For added functionality, for example for logging admins out, OpenWRT based software can be used, although those functions have not yet been implemented.

While no service exists yet from MNOs, we estimate their costs based on Mobiilivarmenne. Using Mobiilivarmenne is currently free for clients, if usage is personal, but costs for service providers are unknown. Hardware costs can mostly be eliminated, while users already have smartphones and for infrastructure, existing hardware such as APs can be used.

Using SIM to local Wi-Fi AA adds value to the smartphone ecosystem. To further divide possible costs for EAP-SIM usage is difficult. EAP-SIM always needs an MNO for the first authentication, because only the MNO and a SIM-card manufacturer know what are SIM's Ki and the used A3/A8 algorithm variation for GSM/3GPP/LTE authentication.

It is difficult to see, if any commercial provider would implement SIM-key sharing so, that secret part was divided to a part that implements AuthN for an own operator and to a part, that is free to use by some other operator. Instead, the same functionality can be achieved with a dual-SIM phone, which allows inserting two SIM cards from different operators in to the phone. By using menu option in phone, or even a specific prefix code before call, an alternate SIM card can be chosen without booting the phone. Dual-SIM thus allows change of identity and IMSI without removing the SIM card.

There exists also private GSM networks. An interesting use case of them has been Chaos Computer Club's international CCC-camps [15], where organizers provide private GSM network for attendees of conference by distributing them separate SIM cards for 2 euros. Even, when GSM network used 1.8GHz radio channel under an experimental spectrum licence, GSM encryption also could be used, because the

SIM-card secrets were known to the organizing, private operator. On the other hand, empty GSM cards for testing can cost as much as 18 euros a piece [41].

### 6.3 Platform specific issues

For clients, there is no need for public key infrastructure (PKI) unless EAP-PEAP is used. Under PEAP there are either server certificate or additionally client certificate present. There are smartphones, that do not have EAP-SIM yet available. For example support for EAP-SIM (and -AKA) methods starts in Android only from version 4.x and in iOS from version 5.x. [21].

Generally, to support EAP-SIM with open source software in smartphone, needed components are *pcsc-lite* for accessing SIM card, *wpa\_supplicant* for WPA2 client, and possible used connection manager (*connman*, *wicd*, or *Network Manager* in Linux). This is in line, what was done in testing without *pcsc-lite*, because a file backend was used instead of a SIM card.

If OpenWRT platform is used for CPE, the memory size (32MB) restricts some use. WPA2 software included in basic OpenWRT installation is small, but that does not yet include RADIUS server part or EAP-SIM handling. Freeradius2 is not yet included in OpenWRT and if it was, it would also be based heavily on current Perl environment which itself needs a lot of space. Currently, as of August 9th, 2015, Freeradius is running on version 3 and EAP-AKA is supported through a module from hostapd project. COMP128(versions 1, 2, and 3), which is implementation of A3/A8 algorithms, is supported [11], and so EAP-SIM is available. Yet, Freeradius can be used as Authentication Center (AuC). Diameter (freeDiameter) can be compiled in OpenWRT, which is good, because on 3GPP networks Diameter protocol has more support than RADIUS. If nothing else works, as a backup old-fashioned WWW-authentication portal can be used for offline authentication.

To enable EAP-SIM method, it is necessary to use WPA2-Enterprise mode and thus RADIUS server, but at the same time, existing Wi-Fi network at home usually has been set using WPA2-PSK. Because it was not found, how authenticator could use the same network SSID for both WPA2-PSK (or open access) and WPA2-Enterprise, a separate SSID for dedicated management network was technically needed. Luckily, today's APs allow setting of multiple SSIDs.

Nevertheless, if Wi-Fi was limited to only one SSID, then we would need another way to indicate that user wants to get in to management network. Separation can be done for example at client's end by using different realm on AuthN identity. It

also can be done by adding hints for different destination to username (username decoration) or by using different authentication method. All these methods allow Authentication Server to distinguish, what the user wants to do; get normal access to home network or make trust anchor claim to management network.

If Lock-and-key method was used instead of EAP-SIM RADIUS, then separate management SSID would not be needed. Roles were given at authentication server or designated router after the smartphone has done login to it via normal access network.

## 6.4 Usability with access points

There are three main types of wireless access points: open access points without configuration on client's end, access points demanding one-time only setup of client such as WPA2-PSK or WPA2-Enterprise, and access points similar to captive portal, where user have to login manually first time through web-page login. It is well known, that the usability of the captive portal Wi-Fi network is burden, because a user needs to go through a web portal logins with a username-password authentication procedure and those are different for every network. Additionally, the user is often required to switch between cellular and Wi-Fi data access when they change their location, and that would disrupt the network connection.

An industry brand Hotspot 2.0 (HS2.0) addresses those issues and tries to simplify user's switch between Wi-Fi and cellular to automate the roaming experience. HS2.0 is driven by two alliances: Wi-Fi Alliance has a certification program (Passpoint) for HS2.0 compatible devices, while the Wireless Broadband Alliance has a program called Next Generation Hotspot (NGH), targeted to user experience [50].

HS2.0 enables the selection of the network based on the ownership, services and performance characteristics *before* a Wi-Fi client has been associated to a HS2.0 AP. The technology is built on IEEE 802.11u specification. In its second release version the operator would have control on which network the smartphone would carry its data transmission.

Ericsson's technology journal "Review" revealed in 2012, that their HS2.0 goal is "to make roaming between Wi-Fi and 3GPP/LTE networks smoother and to bring operator-grade to Wi-Fi by putting control in operators side". More than offloading traffic, plans were to bring to Wi-Fi other services too [38]. Developing HS2.0 a few steps further would add mobile traffic and internet off-loading on to Wi-Fi networks and that would be the missing link in interworking those two worlds.

In HS2.0, the cellular network may signal the smartphone and propose it to switch to Wi-Fi. The smartphone then would try to find a HS2.0 capable access point and continue using Wi-Fi instead of cellular network. In a similar way, the smartphone could receive signal from the cellular network, when controlling changes need to be approved. The smartphone would then make some tests to proof the local AP's suitability for HS2.0. If those succeed, then the cellular network would continue and order the phone to make a switch to the Wi-Fi network, authenticate there with EAP-SIM (or -AKA) and transfer services to Wi-Fi not forgetting the transfer in to the management network. This scenario could be studied further.

If HS2.0 was used here to automate the part, where the smartphone needs to change from cellular to local management Wi-Fi network and back, we probably would miss the user decision part. The user, not the operator, must give his consent to access the management network, so it is important, that the switch would not be automatic or forced. In a way, operator aided roaming between Wi-Fi and cellular works in a different level than here described trust-anchor method. The operator is interested on the access network, while we are interested in the side result of getting access, namely the achieved trusted access point.

## 6.5 Discussion

[\*Some part to be moved to the Conclusion chapter\*]

Although the core technology has been there for more than ten years and the hardware and the applications mostly support it, there can be many reasons why SIM-based methods are not in wider use. One can guess, that the reasons are similar that happened for example with WiMax technology, which was used for broadband network connections to rural areas. Technically that was well enough, but demand was not so large. Additionally lower speed technologies that were cheaper at that time, such as cellular modems were thought to be sufficient, even when inferior technically. It could also be, that market is waiting for future products, and does not want to invest on existing technology, which can be seen as "old".

SIM card of the smartphone, used together with Wi-Fi access to home network helps in verifying the changes, and for that we have presented some options. Location of AuthN and AuthZ components may vary depending on state of process. Always in the beginning, AuthN lies outside the home network, but later it can also use local point. AuthZ, on the other hand, may be located more freely.

Based on 802.1X standard, the management port is activated and RADIUS transfers the orders for correct AA. The smartphone traffic is directed in to an own virtual LAN segment (VLAN), which is dedicated to management. Thesis thus uses an old, yet simple method for problem risen in a modern environment home network.

Disconnection from a normal (Wi-Fi) access network must happens, before phone can get into the management network. It means, that all stateful network connections using Wi-Fi will close at that point. Smartphones do not have multiple wireless connections, but mobile data connections may stay up. Even then, the default routing in the smartphone may change.

EAP does by definition only AuthN part although the successful authentication often precedes ad hoc AuthZ if nothing else is demanded. EAP-SIM handles this part, but for AuthZ something else is needed and so some methods have been presented to add the right role to the authenticated identity.

In the design chapter (Chapter 4) it was questioned whether the proxying RADIUS server can read and alter the messages on their way or is the messaging secured by encryption, integrity hashes and digital signatures. Later it was learned, that the message's integrity is indeed protected, if only in a very light way, but not encrypted.

Altering or adding of RADIUS messages can be chosen from many attributes of RADIUS's vocabulary. They can carry extra information for provisioning in AuthZ phase. Exactly which of them are used remains to implementer's decision. Term *provisioning* can mean adding users to home network with correct attributes such as authentication method and identification. It also can mean identifying users later and giving them more attributes and access rights dynamically. Linking a user to a SIM card is also provisioning and that has happened earlier by the MNO.

Another problem using on-fly alteration of the AuthZ data in RADIUS ACCESS-ACCEPT message is the mapping between IMSI and TMSI identities. The authenticator does know the original user and mapping of IMSI to TMSI, but needs AuthZ information. That can be received from the remote operator, which would be easier for the authenticator or there might be a proxying RADIUS, which resolves the AuthZ information and generates an ACCESS-ACCEPT packet. The latter has in turn issues with TMSI.

When a proxying RADIUS gets the temporary SIM-identity (TMSI) instead of a beforehand known IMSI identity, there will be problem because the proxying RADIUS knows for sure only the originating server. The MNO in turn does not necessary know about the roles, and it can directly perform only AuthN.

It seems, that AuthZ data must be mapped during the first phase of EAP-SIM AuthN, when IMSI still is available, and in some way that map must be forwarded to the proxying RADIUS servers. These issues are fully avoided only in scenario II presented in Chapter 4, where there is local authentication server in home network. Partly avoidance can be reached, if only full authentication is used, i.e., the authentication is always checked from the MNO and no fast re-authentication is used, thus eliminating use of TMSI altogether.

## 6.6 Alternative management models

If we were to question or evolve the given model of change propagation, the trust anchoring would also need some modifications. Alternative method is that the changes could be marked some way, so that they need approving and then there could be a specific change-approval message, which must be sent through the management network. That approval would be akin to *commit* in database actions and its form could perhaps include a digest of change message as a verification. Because smartphone is not actively listening to the CPE, it cannot input those request. Together with the chosen method, there are four planned methods to distribute changes. The first uses the smartphone as a direct commander and it is the one described throughout the thesis. The second would not involve smartphone at all but depended on earlier set trust. The remaining two would depend on more complex setups involving tokens and are presented here just as ideas.

I) Changes are delivered from cloud to smartphone, which forwards them to devices located at management network after the authentication has succeeded. This is the method chosen here and already explained.

II) Changes are delivered normally from cloud to CPE (CPEs) without interaction from the smartphone. Such changes would not need AA at all or changes include credentials to login to targets. An example of change is a modification in network segment, which does not change network topology of other domains. It is still unclear to the author which type of changes represents majority of the requests: those that need approval, or those that may proceed without approval. An educated guess is, that every change set needs to check the trust anchor's presence.

III) Besides AP, also CPE would have direct connection to the Cloud. Changes are then delivered from cloud to CPE functioning as a central management station without interaction from the smartphone. Digest of what is going to happen would be sent to smartphone from the cloud over the air (OtA). Smartphone would authenticate in to the management network (if not already there) and send through

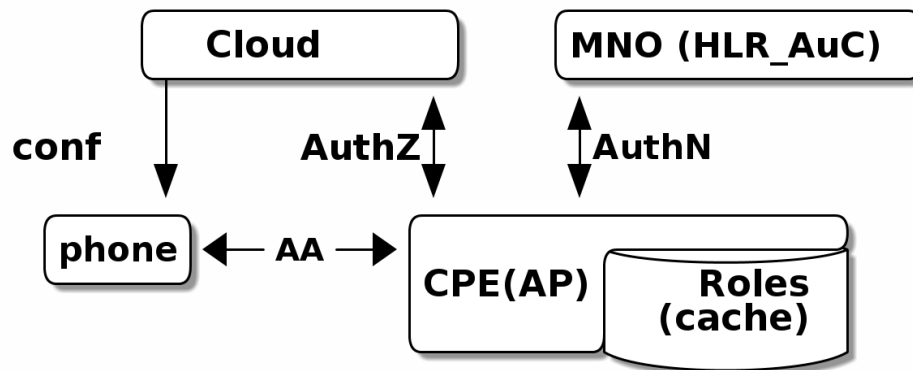


it the digest token it received from the cloud as an approval message to the central management station inside home network, which then forwards the configuration changes to other devices.

IV) Variation of III is that when CPE itself is an authenticator, it could proceed on propagating changes when it receives ACCESS-ACCEPT. Otherwise it must timeout waiting for phone's AA and drop received changes without forwarding them.

In last two planned ways, the smartphone may receive the AuthN token with a message explaining what is going to happen in the change. As the CPE and the authenticator may be separate devices, approving happens by sending the token from the smartphone to the CPE via the management network where the authenticator gives access. Further, only the initial bootstrap as well as the change of admin roles and some dangerous combination of commands would need AA with the smartphone.

In the future, other components than the local controller might have a direct connection to the cloud. Scenario VI (Figure 6.2) is similar to Scenario I presented earlier, but now AuthZ is checked direct from the Cloud by the CPE instead of the MNO. If there is no connection to the cloud, the fall-back is to work just like II. So also this scenario needs local store for caching admin IMSIs (roles).



*Figure 6.2 Future, direct to Cloud connected CPE. AuthN comes from MNO*

## 6.7 Bootstrapping with and without PKI

Homenet WG proposes the use of Public Key Infrastructure (PKI) at home. The public key cryptography is processor intensive and its asymmetric keys are usually used just in the beginning of communication. There they can be used to securely

negotiate symmetric keys which allows faster cryptographic processing. Rest of this section discusses possible PKI usage.

To use PKI for trust anchoring and AuthN in an empty environment, bootstrapping protocols are first needed. The bootstrapping protocols in trust finding are used to bring the first trust anchor in an environment and later use that anchor to attach other devices to the same trust circle. A draft from

Despite the etymology of name bootstrapping, “Lift oneself by his own bootstraps”, bootstrapping usually needs some input from outside. For that, AuthN of the trust anchor device would happen by checking device’s vendor certificates. Behringer [8] proposes, that one device at home is first chosen as a trust anchor using its unique properties and this anchored device later becomes home network’s Certificate Authority (CA) service. smartphone would play user interface role only, but not the trust anchor role. The rest of the home network devices would apply for certificates from this trust anchor to get under the same trust circle. Key creation, key exchange and their usage is explained in similar draft from Pritikin[36].

This model could be expanded to a full ticket enabled AA, where time-limited tickets (tokens) for both AuthN and AuthZ exist for different services. There, the end devices first try to get AuthN ticket, which does not include yet the AuthZ. With that ticket, the device may later apply for AuthZ ticket, which opens the correct port to change management service from CPE. That is very similar to Kerberos AA, which is based on Needham-Schroeder protocol. Another bootstrapping architecture is defined in GBA (General Bootstrapping Architecture) [?], which is 3GPP’s technical specification. It benefits from MNO’s or any other identity center’s knowledge of user’s SIM. In GBA, Trusted Third Party authentication center would be setup with the help of an MNO. One service would then authenticate an entity, here smartphone, and give it a time-limited ticket as a proof that the entity has been authenticated. When the entity wants to connect to the service, it asks from the central server again for ticket, but this time for the service by presenting the authentication ticket. In return it receives a service ticket which it can present to the wanted service, who verifies the access based on the ticket.

As can be seen, no public key infrastructure (PKI) is used before setup phase is ready. If EAP-SIM was applied in such environment, it would be used only once, namely in the bootstrapping phase to setup the CA trust anchor. Only after that, can the CA start delivering PKI. Kerberos needs shared keys, and EAP-SIM helps in providing those. The smartphone would hold a token with a validity time and had to present that when accessing services, here the management network. The

validity time would automatically disallow the management if applied later than allowed time.

Trust can also be requested with help of device's other unique properties than SIM. Recently, devices have appeared on the market, that have vendor-provided certificates inside them and this brings public key infrastructure as one possible alternative for learning trusted identity. The device proves its identity by presenting a certificate, which has been issued by a trusted vendor and which includes device's public key. Private part of a key pair is inside the device's trusted hardware store and does not ever leave the hardware. Vendor trust is needed for checking the issued certificates and so the trust verification of individual devices is merely transferred to trust verification of the vendor. Root CAs are trust anchors also and can be read in the same way from the device's read-only store. CPE could use vendor-issued certificate for AuthN of some earlier unknown device.

## 7. CONCLUSION

The environment described in thesis is a modern complex home network management, whose configuration management tools are external in the cloud. The thesis concentrates on three main parts: the smartphone driven authorization and authentication at the home networks, the connection to the existing change management model from the external cloud service, and the security issues in that environment.

The trust issues between the home network and the cloud are searched through a smartphone located in the intersection of both domains. Home network's future needs, for example the change of the authority and the delegation of the configuration management have been described. To solve those needs, a method to approve changes indirectly has been proposed. The approval follows from a successful authentication and authorization with EAP-SIM method by the smartphone and that also sets a trust anchor to the smartphone.

For testing purposes, a real working EAP-SIM test bed with fake credentials and a fake mobile operator representing EAP-SIM authentication flow was built. A dual-role model, which binds the smartphone to the home network and grants it rights to make changes there, has been proposed. An indirect way to approve changes is achieved by linking the authorized access to the management network. After the authorization, the smart phone is free to configure the devices with its local application. Another way to convince the devices about trusted source is to send approval tickets that can be verified on reception, but that would involve a more complex setup.

Complexity of existing models in interworking was one motivator for the work. The research on the subject did reveal some reasons for the complexity, that are difficult to overcome with simplistic methods shown here without losing security at the same time. There are some obvious weaknesses in the proposed solution such as missing continuous authorization after management access has been granted. The application for the smartphone is yet to be fully implemented. Possible usage must carefully check the safety limits even when for example the RADIUS protocol still has strengths in security today. The thesis only scratches bootstrapping problems and

the issues in the home network bootstrapping need to be studied more thoroughly. A one promising lead would be to use tickets in Kerberized way as in GBA.

After those shortcoming, the provisioning of manager users at home networks still would minimize with the proposed technique, as the users already own a smartphone, which is an identifiable and trusted object. As a positive side effect, the two-factor AuthN from a hardware based SIM would strengthen an existing security. Finally, the cloud management tools would benefit from the trust anchor on the smartphone and could be developed further to aid in resolving trust issues in future home networks.

## REFERENCES

- [1] “Home networking,” Jul 2011. [Online]. Available: <http://datatracker.ietf.org/doc/charter-ietf-homenet/>
- [2] B. Aboba, D. Simon, and P. Eronen, “Extensible Authentication Protocol (EAP) Key Management Framework,” RFC 5247 (Proposed Standard), Internet Engineering Task Force, Aug. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5247.txt>
- [3] B. Aboba and J. Vollbrecht, “Proxy Chaining and Policy Implementation in Roaming,” RFC 2607 (Informational), Internet Engineering Task Force, June 1999. [Online]. Available: <http://www.ietf.org/rfc/rfc2607.txt>
- [4] AIMCID-community, accessed: 2015-09-02. [Online]. Available: <https://github.com/SecUpwN/Android-IMSI-Catcher-Detector/wiki>
- [5] J. Arkko, A. Brandt, O. Troan, and J. Weil. (2014, Oct) "ipv6 home networking architecture principles". [Online]. Available: <https://datatracker.ietf.org/doc/rfc7368/>
- [6] J. Arkko and H. Haverinen, “Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA),” RFC 4187 (Informational), Internet Engineering Task Force, Jan. 2006, updated by RFC 5448. [Online]. Available: <http://www.ietf.org/rfc/rfc4187.txt>
- [7] J. Arkko, V. Lehtovirta, and P. Eronen, “Improved extensible authentication protocol method for 3rd generation authentication and key agreement (EAP-AKA’),” RFC 5448 (Informational), Internet Engineering Task Force, May 2009. [Online]. Available: <http://www.ietf.org/rfc/rfc5448.txt>
- [8] M. Behringer, M. Pritikin, and S. Bjarnason. Bootstrapping trust on a homenet. [Online]. Available: <http://tools.ietf.org/id/draft-behringer-homenet-trust-bootstrap-02.txt>
- [9] M. Chiba, G. Dommety, M. Eklund, D. Mitton, and B. Aboba, “Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS),” RFC 5176 (Informational), Internet Engineering Task Force, Jan. 2008. [Online]. Available: <http://www.ietf.org/rfc/rfc5176.txt>
- [10] A. DeKok and A. Lior, “Remote Authentication Dial In User Service (RADIUS) Protocol Extensions,” RFC 6929 (Proposed Standard), Internet Engineering Task Force, Apr. 2013. [Online]. Available: <http://www.ietf.org/rfc/rfc6929.txt>

- [11] A. DeKok, “COMP128 implementation in FreeRADIUS,” accessed: 2015-05-10. [Online]. Available: [https://github.com/FreeRADIUS/freeradius-server/blob/master/src/modules/rlm\\_eap/libeap/comp128.c](https://github.com/FreeRADIUS/freeradius-server/blob/master/src/modules/rlm_eap/libeap/comp128.c)
- [12] W. Diffie and M. E. Hellman, “New directions in cryptography,” *Information Theory, IEEE Transactions on*, vol. 22, no. 6, pp. 644–654, 1976.
- [13] “Chapter 3 - communication security: Remote access and messaging,” in *How to Cheat at Securing Your Network*, ser. How to Cheat, I. Dubrawsky, Ed. Burlington: Syngress, 2007, pp. 65 – 104,75. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/B978159749231750006X>
- [14] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman, “Network Configuration Protocol (NETCONF),” RFC 6241 (Proposed Standard), Internet Engineering Task Force, June 2011. [Online]. Available: <http://www.ietf.org/rfc/rfc6241.txt>
- [15] “GSM - 28C3 public wiki,” accessed: 2015-05-10. [Online]. Available: <https://events.ccc.de/congress/2011/wiki/GSM>
- [16] S. Hartman, T. Clancy, and K. Hoeper, “Channel-Binding Support for Extensible Authentication Protocol (EAP) Methods,” RFC 6677 (Proposed Standard), Internet Engineering Task Force, July 2012. [Online]. Available: <http://www.ietf.org/rfc/rfc6677.txt>
- [17] J. Hassell, *RADIUS*. O’Reilly, Oct 2002.
- [18] H. Haverinen and J. Salowey, “Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM),” RFC 4186 (Informational), Internet Engineering Task Force, Jan. 2006. [Online]. Available: <http://www.ietf.org/rfc/rfc4186.txt>
- [19] H. Haverinen, “Interworking between wireless LAN and GSM/UMTS cellular networks: Network access control, mobility management and security considerations,” Ph.D. dissertation, Tampere University of Technology, 2004.
- [20] IEEE, “IEEE 802.1: 802.1X-2010 - Revision of 802.1X-2004,” 2010, accessed: 2015-05-10. [Online]. Available: <http://www.ieee802.org/1/pages/802.1x-2010.html>
- [21] Infocomm Development Authority of Singapore, “SIM-based connection guide,” accessed: 2015-05-10. [Online]. Available: <http://www.ida.gov.sg/Infocomm-Landscape/Infrastructure/Wireless/Wireless-at-SG/For-Consumer/SIM-based-Connection-Guide>

- [22] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, “The Trickle Algorithm,” RFC 6206 (Proposed Standard), Internet Engineering Task Force, Mar. 2011. [Online]. Available: <http://www.ietf.org/rfc/rfc6206.txt>
- [23] S.-H. Lin, J.-H. Chiu, and S.-S. Shen, “Authentication schemes based on the eap-sim mechanism in gsm-wlan heterogeneous mobile networks,” in *INC, IMS and IDC, 2009. NCM '09. Fifth International Joint Conference on*, "Aug" 2009, pp. 2089–2094.
- [24] J. Lindqvist, T. Aura, G. Danezis, T. Koponen, A. Myllyniemi, J. Mäki, and M. Roe, “Privacy-preserving 802.11 access-point discovery,” in *Proceedings of the Second ACM Conference on Wireless Network Security*, ser. WiSec '09. New York, NY, USA: ACM, 2009, pp. 123–130. [Online]. Available: <http://doi.acm.org/10.1145/1514274.1514293>
- [25] F. Maissen, “Kostenfalle für Swisscom-Kunden,” accessed: 2015-05-04. [Online]. Available: <http://www.srf.ch/konsum/themen/multimedia/kostenfalle-fuer-swisscom-kunden>
- [26] J. Malinen, “Linux WPA/WPA2/IEEE 802.1X Supplicant,” accessed: 2015-05-07. [Online]. Available: [http://w1.fi/wpa\\_supplicant/](http://w1.fi/wpa_supplicant/)
- [27] “EAP - Moonshot Wiki,” accessed: 2015-05-10. [Online]. Available: <https://wiki.moonshot.ja.net/display/Moonshot/EAP#EAP-HowMoonshotusesEAP>
- [28] M. Nakhjiri and M. Nakhjiri, *AAA and Network Security for Mobile Access - Radius, Diameter, EAP, PKI and IP Mobility*. Wiley, 2005.
- [29] K. Narayan and D. Nelson, “Remote Authentication Dial-In User Service (RADIUS) Usage for Simple Network Management Protocol (SNMP) Transport Models,” RFC 5608 (Proposed Standard), Internet Engineering Task Force, Aug. 2009. [Online]. Available: <http://www.ietf.org/rfc/rfc5608.txt>
- [30] D. Nelson and A. DeKok, “Common Remote Authentication Dial In User Service (RADIUS) Implementation Issues and Suggested Fixes,” RFC 5080 (Proposed Standard), Internet Engineering Task Force, Dec. 2007. [Online]. Available: <http://www.ietf.org/rfc/rfc5080.txt>
- [31] OpenGroup. [Online]. Available: <https://collaboration.opengroup.org/projects/jericho/>
- [32] A. Palekar, D. Simon, J. Salowey, H. Zhou, G. Zorn, and S. Josefsson, “Protected eap protocol (peap) version 2,” Oct 2004. [Online]. Available: <http://tools.ietf.org/id/draft-josefsson-pppext-eap-tls-eap-10.txt>



- [33] W. R. Parkhurst, *Cisco router OSPF*. McGraw-Hill, 1998.
- [34] S. Patel, “Analysis of EAP-SIM session key agreement.” [Online]. Available: <https://www.ietf.org/proceedings/57/slides/eap-11.pdf>
- [35] F. Pereniguez, G. Kambourakis, R. Marin-Lopez, S. Gritzalis, and A. F. Gomez, “Privacy-enhanced fast re-authentication for eap-based next generation network,” *Computer Communications*, vol. 33, no. 14, pp. 1682–1694, 9/1 2010.
- [36] M. Pritikin, M. Behringer, and S. Bjarnason, “Bootstrapping key infrastructures,” Tech. Rep., 2014, accessed: 2015-02-04. [Online]. Available: <http://tools.ietf.org/id/draft-pritikin-bootstrapping-keyinfrastructures-01>
- [37] H. Rachidi and A. Karmouch, “A framework for self-configuring devices using tr-069,” in *Multimedia Computing and Systems (ICMCS), 2011 International Conference on*, April 2011, pp. 1–6.
- [38] S. Rayment and J. Bergström, “Achieving carrier-grade Wi-Fi in the 3GPP world,” *Ericsson Review*, 2012, accessed: 2015-05-05. [Online]. Available: [http://www.ericsson.com/res/thecompany/docs/publications/ericsson\\_review/2012/er-seamless-wi-fi-roaming.pdf](http://www.ericsson.com/res/thecompany/docs/publications/ericsson_review/2012/er-seamless-wi-fi-roaming.pdf)
- [39] C. Rigney, S. Willens, A. Rubens, and W. Simpson, “Remote Authentication Dial In User Service (RADIUS),” RFC 2865 (Draft Standard), Internet Engineering Task Force, June 2000, updated by RFCs 2868, 3575, 5080, 6929. [Online]. Available: <http://www.ietf.org/rfc/rfc2865.txt>
- [40] B. Silverajan, J.-P. Luoma, M. Vajaranta, and R. Itäpuro, “Collaborative cloud-based management of home networks,” in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*. IEEE, 2015, pp. 786–789.
- [41] “Smartjac TEST (U)SIM card webshop,” accessed: 2015-05-10. [Online]. Available: <http://www.smartjac.biz/webstore/samples-to-order/smartjac-test-sim-configurable-options>
- [42] D. A. Sokolov, “Digitale Selbstverteidigung mit dem IMSI-Catcher-Catcher,” *C’t*, Aug 2014, visited April 2015. [Online]. Available: <http://heise.de/-2303215>
- [43] C. Sriharsha and S. Sandhya, “Role of DIAMETER stack protocol in IMS network architecture,” *IJITR*, vol. 3, no. 3, 2015. [Online]. Available: <http://ijitr.com/index.php/ojs/article/view/642>
- [44] Y.-M. Tseng, “USIM-based EAP-TLS authentication protocol for wireless local area networks,” *Computer Standards & Interfaces*, vol. 31, no. 1, pp. 128 –

- 136, 2009. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0920548907001213>
- [45] S. Turner and L. Chen, “Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms,” RFC 6151 (Informational), Internet Engineering Task Force, Mar. 2011. [Online]. Available: <http://www.ietf.org/rfc/rfc6151.txt>
- [46] S. Venaas, “radsecproxy,” accessed: 2015-04-22. [Online]. Available: <https://software.uninett.no/radsecproxy>
- [47] R. Ward and B. Beyer, “Beyondcorp: A new approach to enterprise security,” *login.*, vol. 39, No. 6, pp. 6–11, 2014. [Online]. Available: <http://static.googleusercontent.com/media/research.google.com/file/pubs/archive/43231.pdf>
- [48] J. Wey, J. Luken, and J. Heiles, “Standardization activities for IPTV set-top box remote management,” *Internet Computing, IEEE*, vol. 13, no. 3, pp. 32–39, May 2009.
- [49] S. Winter, M. McCauley, S. Venaas, and K. Wierenga, “Transport Layer Security (TLS) Encryption for RADIUS,” RFC 6614 (Experimental), Internet Engineering Task Force, May 2012. [Online]. Available: <http://www.ietf.org/rfc/rfc6614.txt>
- [50] Wireless Broadband Alliance, accessed: 2015-09-04. [Online]. Available: <http://www.wballiance.com/key-activities/next-generation-hotspot/>
- [51] Wireshark community, “WLAN (IEEE802.11) capture setup.” [Online]. Available: <https://wiki.wireshark.org/CaptureSetup/WLAN>
- [52] J. Wright, “I Can Hear You Now - Eavesdropping on Bluetooth Headsets,” SANS, 2007, accessed: 2015-10-03. [Online]. Available: <http://www.willhackforsushi.com/presentations/icanhearyounow-sansns2007.pdf>
- [53] T. Xie, F. Liu, and D. Feng, “Fast collision attack on MD5.” *IACR Cryptology ePrint Archive*, vol. 2013, p. 170, 2013.

## APPENDIX A: SHELL, THAT STARTS PROGRAMS AND LOGS

```
#!/bin/sh -x
# Shell to start programs needed to demonstrate EAP-SIM
# authentication on simulated PHONE and HLR AUC env.
# All used programs are from wpa (v2.3) reference package
# - wpa-supPLICANT
# - RADIUS-server
# - HLR-AuC
# External WPA2-RADIUS AP hardware is used
#
# options:
# -t more timestamps to xxx...
# -K include keydata to debug
# -dddd more debug
#
# usage:
#      ./apd [OPTION]...

# Variables
BASE=$HOME/gitdocs/di/testit
WPASUPPLICANT=$BASE/wpa_supPLICANT
HOSTAPD=$BASE/hostapd
HLR=$BASE/hlr_auc_gw
WPASUPPLICANTCONF=$BASE/wpa-simtest-owrt2.conf

# sim triplets, when EAP-SIM used
SIM=$BASE/hostapd.sim_db
# Milenage parameters, when AKA used
MILENAGE=$BASE/hlr_auc_gw.milenage_db
# HOSTAPD parameters
# settings for hostapd include wired, eap_server, eap-handler
HOSTAPDCONF=$BASE/hostapd-jmdemo.conf
# timestamped logs and confs
TIMESTAMP='date +s%y%m%d-%H%M%S'
TARGET=$BASE/demot/ap-$TIMESTAMP

mkdir $TARGET
cp $0 $HOSTAPDCONF $SIM $MILENAGE $TARGET
# reset and cleanup locks and sockets
pkill hlr_auc_gw; pkill wpa_supPLICANT; pkill hostapd
if [ -S /tmp/hlr_auc_gw.sock ] ; then
    rm -f /tmp/hlr_auc_gw.sock
fi
```

```

if [ -S ./eth0 ] ; then
    rm -f ./eth0
fi

### 1. HLR_AUC
# startup using SIM-triplet
# $HLR -g $SIM > $TARGET/hlr-debug &
# startup using MILENAGE. Works also with SIM
$HLR -m $MILENAGE > $TARGET/hlr-debug &

### 2. HOSTAP (in RADIUS-EAP-handler mode)
ifconfig wlan0 up
# captures for RADIUS (wired, AP-RADIUS) and
# EAP (wireless, client-AP)
tshark -i eth0 -w $TARGET/eth0-pcap &
tshark -i wlan0 -w $TARGET/wlan0-pcap &
echo start hostapd
# start RADIUS in background
$HOSTAPD -Kdt $HOSTAPDCONF > $TARGET/hostapdwired-debug &
echo "if_?_==_0_ then_ RADIUS_ server_ started_ ok"
sleep 1

### 3. WPA_SUPPLICANT
echo starting supplicant..
$WPASUPPLICANT -dK -iwlan0 -c $WPASUPPLICANTCONF \
    -D nl80211 > $TARGET/wpasupp-extapradius-debug &
### Live analysing
echo starting analyze..
cd $BASE/demot
cd 'ls -d ap-*|tail -1' /
sleep 1
# follow 3 log files, with color coding set in multitail.conf
multitail -F ../multitail.conf -N 10000 -CS eap-sim -ts host*debug
    ↪ -i wpa*debug
# alternatively, start this in own window
# xterm -e $BASE/demot/anamulti &

# if tests take over 15 mins, user had fallen in sleep. Commit
    ↪ logs.
sleep 900
pkill tshark
git add $TARGET
git commit -m "apd-tty_ tests_ $TIMESTAMP_"

```

## APPENDIX B: WPA2-SUPPLICANT SETTINGS

```
# EAP-SIM with a GSM SIM or USIM
network={
    ssid="simtest"
    key_mgmt=WPA-EAP
    eap=SIM
    # If uncommented, the REAL SIM would be used instead of simulator
    # pin="1234"
    # pcsc=""
    identity="1232010000000000" # IMSI. Can also be pseudonym
    password="90dca4eda45b53cf0f12d7c9c3bc6a89:
        ↪ cb9cccc4b9258e6dca4760379fb82581"
}

# credentials can also be presented on their own block. Not used
  ↪ here.
cred={
    imsi="1232010000000000" # format should be <MCC> | <MNC> | '-' |
        ↪ <MSIN> ,
                                # i.e., 35840-123456789, so this would
                                ↪ not work.
    milenage="90dca4eda45b53cf0f12d7c9c3bc6a89:
        ↪ cb9cccc4b9258e6dca4760379fb82581"
}
```

## APPENDIX C: RADIUS SERVER CONFIGURATION

```
# no wireless functionality here, only RADIUS/EAP
driver=none
# file for external AP RADIUS secrets
radius_server_clients=hostapd.radius_clients
# eap-handler enabled
eap_server=1
# mapping of eap credentials to SIM,AKA and AKA' protocols
eap_user_file=./hostapd.eap_user
# Inter-process communication with hlr_auc_gw process
eap_sim_db=unix:/tmp/hlr_auc_gw.sock
```

## APPENDIX D: HLR\_AUC\_GW PROGRAM'S MILENAGE FILE

```
# Parameters for Milenage (Example algorithms for AKA).
# The example Ki, OPc, and AMF values here are from
# 3GPP TS 35.208 v6.0.0 4.3.20 Test Set 20.
# SQN is the last used SQN value. These values can be
# used for both UMTS (EAP-AKA) and GSM (EAP-SIM)
# authentication. In case of GSM/EAP-SIM, AMF and SQN
# values are not used, but dummy values will need to be
# included in this file.

# IMSI          Ki
# OPc          AMF  SQN
232010000000000 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 000000000000
232010000000000 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 0000000000001
232010000000000 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 0000000000002
# e90
244052900226278 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 0000000000003

# nokia e71, old found SIM card
244052161294281 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 0000000000000
244052161294281 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 0000000000001
244052161294281 90dca4eda45b53cf0f12d7c9c3bc6a89
# cb9cccc4b9258e6dca4760379fb82581 61df 0000000000002

# These values are from Test Set 19 which has the AMF separation
# bit set to 1 and as such, is suitable for EAP-AKA' test.
555444333222111 5122250214c33e723a5dd523fc145fc0 981
# d464c7c52eb6e5036234984ad0bcf c3ab 16f3b3f70fc1
```