Approaches for simplified hotspot logins with Wi-Fi devices

Master Thesis

Author(s):
Wiederkehr, Patrick

Publication date:
2009

Permanent link:
https://doi.org/10.3929/ethz-a-005899210

Approaches for Simplified Hotspot Logins with Wi-Fi Devices

by

Patrick Wiederkehr

A Master Thesis

presented to ETH Zürich,

Swiss Federal Institute of Technology Zürich

in fulfillment of the

thesis requirements for the degree of

Master of Science

in Computer Science, ETH

Supervisors:

Prof. Thomas Gross,
LST Computer Science Department, ETH Zürich

Dr. -Ing. Stefan Mangold,
Swisscom

swisscom

Zürich, Switzerland 2009

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Abstract

This thesis presents an overview of the different approaches for Wi-Fi login, with particular focus on public hotspots. This mainly involves solving the authentication problem. Common, as well as alternative solutions are presented and critically discussed with respect to the user's security and the usability. Today, public hotspots are not as secure as they could be if they implemented current standards and protocols for secure wireless networking. The issues with these new standards in public network environments are discussed and possible solutions presented. In order to facilitate this discussion, several underlying concepts, protocols and standards in the context of wireless LAN security are presented. The thesis concludes with an evaluation of the different approaches for hotspot authentication.

Keywords: *public WLAN, hotspot, authentication, wireless security, 802.1X, EAP, AAA, WISPr*

Zusammenfassung

„Methoden für den vereinfachten Login an Hotspots für Wi-Fi Geräte“

Diese Masterarbeit gibt eine Übersicht über die verschiedenen Ansätze für den Login in öffentlichen Netzen (Public WLAN / PWLAN / Public Hotspots). Dabei geht es hauptsächlich darum, das Authentisierungs-Problem zu lösen. Aktuelle, wie auch alternative Methoden werden vorgestellt und kritisch diskutiert, mit dem speziellen Fokus auf die Sicherheit des Kunden und die Bedienbarkeit (Usability) der Lösung. Noch sind öffentliche Netze nicht so sicher wie sie sein könnten, wenn sie aktuelle Standards und Protokolle für sichere, drahtlose Netzwerke einsetzen. Wir diskutieren die Probleme mit diesen neuen Standards im Zusammenhang mit öffentlichen Netzen und präsentieren mögliche Lösungensansätze. Die Arbeit schliesst mit einer Bewertung der verschiedenen Methoden für Authentifizierung an öffentlichen Hotspots.

Declaration

Hereby I declare that this thesis is entirely my own work. Cited sources of literature are perceptibly marked and listed at the. The work has not been submitted previously in same or similar form to another examination committee and was has not been published previously.

Patrick Wiederkehr, 30. April 2009

Intended Audience

This text is addressed also to non computer scientists interested in the topic, while still preserving a scientific level. However, basic knowledge of computer systems, network architectures and protocols is required.

Table of Contents

List of Figures

Abbreviations

3G	Third Generation (Networks)
AAA	Authentication, Authorization and Accounting
AuC	Authentication Centre
CA	Certification Authority
CHAP	Challenge Handshake Authentication Protocol
CPP	Captive Portal Page
CRL	Certificate Revocation List
DHCP	Dynamic Host Configuration Protocol
EAP	Extensible Authentication Protocol
EAP-FAST	Flexible Authentication via Secure Tunnelling
EAP-SIM	EAP based on the Subscriber Identity Module
EAP-TLS	EAP Transport Layer Security
EAP-TTLS	Tunnelled TLS
EAPoL	Extended Authentication Protocol over LAN
ETSI	European Telecommunications Standards Institute
GSM	Global System for Mobile communications
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IAB	Internet Architecture Board
IANA	Internet Assigned Numbers Authority
ID	Identity
IEEE	Institute of Electrical and Electronics Engineers
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
IP	Internet Protocol
IPSec	Internet Protocol Security
IPv6	Internet Protocol Version 6
IRAP	International Roaming Access Protocols
ISP	Internet Service Provider
LAN	Local Area Network
MAC	Media Access Control
MD5	Message Digest 5
MITM	Man in the Middle (Attack)
NAK	Negative Acknowledgement
NAS	Network Access Server
PAC	Protected Access Credential
PAC	Public Access Control Gateway
PAP	Password Authentication Protocol
PAP	Password Authentication Protocol
PEAP	Protected EAP
PIN	Personal Identification Number
POP	Point of Presence
POP3	Post Office Protocol Version 3
PPP	Point-to-Point Protocol

PSK	Pre-Shared Key
PWLAN	Public Wireless Local Area Network
QoS	Quality of Service
RADIUS	Remote Access Dial in User Service
RAND	Random Challenge
RFC	Request For Comments
RSN	Robust Secure Network
RSN	Robust Secure Network
SCTP	Stream Control Transfer Protocol
SIM	Subscriber Identity Module
SIM	Subscriber Identity Module
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
SRES	Secret Response
SSID	Service Set Identifier
SSL	Secure Sockets Layer
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TEK	Transient EAP Key
TIMSI	Temporal International Mobile Subscriber Identity
TISPAN	Telecoms & Internet Converged Services & Protocols for Advanced Networks
TLS	Transport Layer Security
TLS	Transport Layer Security
TLV	Time Length Value
TPM	Trusted Platform Module
Triple-A	refers to AAA
UAM	Universal Access Method
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication Systems
URL	Universal Resource Locator
USIM	Universal Subscriber Identity Module
VoIP	Voice over [the] Internet Protocol
VPN	Virtual Private Network
VSA	Vendor Specific Attribute
WEP	Wired Equivalent Privacy
Wi-Fi	Wireless Fidelity
WISP	Wireless Internet Service Provider
WISPr	Wireless Internet Service Provider Roaming
WLAN	Wireless LAN
WPA	Wi-Fi Protected Access
WPA	Wi-Fi Protected Access
WPA2	Wi-Fi Protected Access 2
WPA2	Wi-Fi Protected Access 2
WPS	Windows Provisioning Service
ZD Net	Ziff Davis Network

Chapter 1 Introduction

Mobile devices with integrated wireless local area network (WLAN) technology are becoming increasingly popular. Not only laptop computers are Wi-Fi (Wireless Fidelity) enabled, also mobile phones, gaming consoles and digital cameras are being equipped with wireless communication modules. Mobile Internet, mobile applications and services depend on connectivity to online platforms thus requiring instant Internet connection. While handsets can make use of third generation (3G) communication technology, most other devices cannot access these networks. Although there has been lately also a trend towards 3G, resp. Universal Mobile Telecommunication System (UMTS) communication interfaces being integrated into laptop computers and netbooks, 802.11, or Wireless LAN (WLAN) for short, has become fast and cheap enough to be integrated into everyday' things. While at home these devices are easily integrated into our home networks, they are almost useless when travelling, since open and free WLAN access points are rarely available. On the other hand, public WLAN hotspots and open network communities are also increasing in number and already cover large areas in the cities. Most train stations, airports, restaurants and hotels offer wireless access to the Internet through public hotspots.

Unfortunately, public hotspots are not as secure as they could be. For home and enterprise wireless networks strong cryptographic mechanisms do exist, providing confidentiality, data integrity and mutual authentication, hence respecting the user's privacy. Wi-Fi Protected Access (WPA and WPA2), also known as 802.11i and Robust Secure Network (RSN), provide network access control based on shared secrets between the client and the network. WPA-PSK, the private mode of WPA and WPA2 restricts access to users knowing a common passphrase (the so called Pre-Shared Key, hence PSK), and the enterprise mode of WPA and WPA2 relies on 802.1X port based access control and the Extensible Authentication Protocol (EAP) for authentication.

EAP is a framework for different authentication methods. Over 40 so-called EAP methods do exist, the most prominent ones being EAP-TLS (*EAP Transport Layer Security*, [RFC5247]), EAP-TTLS (*Tunneled TLS*, [RFC5281]), EAP-FAST (*Flexible Authentication via Secure Tunneling*, [RFC4851]), PEAP (*Protected EAP* [1]) and EAP-SIM (*EAP based on the Subscriber Identity Module*, [RFC4186]). Which EAP method and which mode of a specific method should be used depends on the security requirements and the administrative cost – the classical usability vs. security trade-off.

Until recently, public hotspot providers were clearly going for usability. Public hotspots do seldom provide link layer encryption and hence do not protect the privacy of the users. Everyone sitting next

to a customer logged into a public hotspot could listen to the communication between the customer and the access point with little effort. This can include private emails, credentials of an e-commerce site, Credit-Card information or anything the user sends across this insecure link. Most people are not aware of that fact and think they are secure. Fortunately, there is the possibility to use VPN clients to encrypt this traffic. This is how most of today's hotspot operators handle the problem. On the one hand, this is a reasonable solution, since it really protects the customer once set up, on the other hand, it delegates the responsibility of securing the communication channel to the customer. Furthermore, it is still possible to steal someone's credentials before the VPN connection is established.

Filling the gap

With the ubiquity of wireless LANs and the customers becoming more and more concerned about privacy and security issues, the current situation is no longer satisfactory. However, as mentioned above, this is a usability vs. security trade-off. Modern authentication methods coming from the enterprise world, such as EAP-TTLS or PEAP, need to be individually configured on the clients. Depending on the operating system, this can be cumbersome and, if done incorrectly, even open doors for attackers. Nevertheless, we think it should be possible to find a solution that on the one hand fulfils today's security needs at public hotspots and on the other hand is accepted by the customers. Since the biggest issue with 802.1X secure networking is the provisioning of the end-devices, we will propose different solutions to this problem. We think most people would accept to go through a one-time setup and configuration process in order to securely use public hotspots. Once the client is set up correctly, the log-in process could be automated and be transparent to the user.

In the context of this thesis, we present the different approaches to authenticate customers at public hotspots. However, authentication, network access and network security in general are very complex issues. While there is a propensity of books and literature providing detailed information about specific protocols, standards and architecture for large networks, many underlying concepts are often assumed to be known. This thesis also presents these concepts in order to achieve an understanding of the whole process. An example of such a concept would be public key cryptography. Apart from knowing what a digital certificate is, it is important to understand the implied infrastructure and cost associated with it.

Furthermore, the environment of public Wi-Fi networks is different from the well known enterprise networks that are the focus in most of the literature. Public Networks, such as Swisscom's Public WLAN (PWLAN), have different requirements with regard to security, usability, and accounting. Thus they have implemented their own proprietary solutions.

Authentication in wireless networks has been heavily researched in the years 2000 to 2006. The work resulted in standards that are now supported by most modern Wi-Fi enabled devices. However, for the last four years, most public hotspot providers are still sticking to the traditional browser based Captive Portal Page (CPP) method, hence not providing strong security and protection for the customer. There are reasons why PWLAN operators have not adapted to the new standards for secure networking which we will identify and discuss.

This thesis gives an overview of authentication methods with special focus on public wireless LANs. Current implementations as well as alternative approaches are discussed. A primary concern is the security and privacy of the users. State-of-the-art protocols for secure wireless communication are presented, as well as their weaknesses and their vulnerability in public hotspot environments. The thesis concludes with an evaluation of EAP methods with respect to their suitability for PWLAN.

1.1 Outline

Chapter Two discusses important concepts for authentication. In public wireless LANs, where the customer has to be billed for service usage, dedicated accounting mechanisms have to be present. The three concepts of authentication, authorization and accounting are known under the name “AAA“ or “Triple A”, which are introduced at the beginning of chapter two. In the context of public networks, authentication can be considered to be the most important aspect of AAA, hence we put our special focus on authentication. We look at how authentication can be performed and what mechanisms can be used. Digital certificates and public key cryptography in general play a very important role in authentication. Therefore, we provide an overview of the topic and explain how digital certificates work. Finally, we identify the primary goals of network security in general.

Chapter Three focuses on standards and protocols for wireless LAN. This includes the WPA and WPA2 standards for wireless LAN encryption which one might already have met in the context of home- or enterprise networking. The enterprise mode of WPA2 is based on 802.1X port based authentication and the Extensible Authentication Protocol (EAP) with its different methods, which will be presented in that chapter. The chapter finishes with an introduction to the AAA protocols RADIUS and Diameter, which are used for authentication in large networks.

With the knowledge about authentication and the general network security goals and standards presented in the second chapter, we put our focus on the special environment of public wireless LANs in **Chapter Four**. We analyze the different requirements of PWLAN and see how authentication is currently being performed by most PWLAN operators. Despite that there is no standard for PWLAN

authentication available, the different operators agreed on a common approach called WISPr. This is described in a recommendation document by the Wi-Fi Alliance and is presented also in chapter four.

Chapter Five is dedicated to PWLAN security. Unfortunately, current PWLAN implementations are not as secure as they could be. We analyze the vulnerabilities of the traditional authentication methods. The rest of the chapter focuses on 802.1X authentication at public hotspots, identifying the problems of this approach and discuss possible solutions.

Chapter Six provides an evaluation of the different approaches for PWLAN authentication. This includes the traditional solution which is performed at hotspots today, as well as the more secure alternative of 802.1X port based authentication with EAP. In the context of this evaluation, we compare the different EAP methods with respect to their suitability in public WLAN environments.

Finally, we make recommendations for PWLAN operators who are considering adopting the secure 802.1 X approach and presenting our conclusions.

Chapter 2 Concepts

The second chapter gives an introduction to authentication in general. The Triple-A (AAA) concept and its underlying models are presented. We identify the primary goals of wireless LAN security in general and how they can be achieved. Important aspects, such as key management and digital certificates, which form the basis for modern wireless LAN security, are presented and put into context.

2.1 Triple "A" (AAA - Authentication, Authorization and Accounting)

If you have a quick look at the Wikipedia entry about AAA, you will find approximately thirty different meanings of the term. AAA can simply refer to *"something that is of high-quality, premier or excellent"*. On the other hand, it is mainly used as an abbreviation. The American Automobile Association may be the most common one, there is a triple "A" syndrome in medicine, and we all know the AAA battery size standard, just to mention a few. Our interest, however, is on the triple "A" in the context of network access control. Here, the three A's stand for Authentication, Authorization and Accounting.

So-called triple-A-systems became popular when Internet Service Providers (ISPs) started to offer dial-up services to their customers. A dial-up connection is initiated by the user dialing the number of the ISP's connection pool which then establishes a connection. Since payment of this kind of service is typically based on the actual duration of the connection (i.e. until disconnection), there had to be a system in place that was able to do three different things:

1. **Authenticate** the customer, i.e. verify which customer is requesting the services
2. **Authorize** the customer to use specific services, such as the connection to the Internet, and
3. accumulate the usage of services in order to bill the customer properly (**Accounting**).

Therefore, a Triple-A-System should answer the three questions: *Who are you? What do you want?* And *how long do you want your requested service?* This is what AAA means.

As we find out in the context of this thesis, the tricky part lies in the "*who?*" question. Let me open a few questions, which are answered in the remainder of this text. How can the ISP be sure that the customer who is requesting a service is actually the claimed user? What if there is an attacker sitting

between the customer and the ISP manipulating the traffic? How can one protect the data traffic and the identity of the customer? Once these questions are answered and appropriate mechanisms are in place, i.e. the customer has successfully been authenticated, then authorization and accounting can be performed accordingly. Without secure authentication mechanisms however, authorization is useless and accounting just not possible.

Authentication, Authorization and Accounting are three different concepts that are intended to solve different issues. We first present the three A's individually in separate Sections. As we shall see later, it does make sense, and is often done in practice, to handle the three A's together in one framework. This is why the servers dealing with Authentication, Authorization and Accounting are referred to as so-called AAA-Servers, which is introduced later.

2.1.1 Authentication

As mentioned above, authentication is not only the first A in the triple A abbreviation, it is the first action to be performed when requesting a service. There exist various different approaches to solve the authentication problem, each addressing different requirements in different environments.

From a user's point of view, authentication can be seen as the log-in process where we provide our username and a password. While this method is very popular, there are cases where this approach is not sufficiently secure, or just not possible. From a more technical point of view, authentication today involves much more than just sending a username and password pair to a server. In the early days of the Internet, when users had leased lines or a dial-up connection via a telephone line, no one was really concerned with security and privacy. User credentials (typically a username and a password) were sent in clear text without considerations to security (the protocol used was the Password Authentication Protocol, PAP [RFC1334]). An attacker would have had to tap into the telephone system in order to listen to the ongoing traffic. This is by far more difficult than sniffing wireless LAN traffic. Wireless networks today demand more sophisticated and more secure authentication methods.

Authentication is closely related to network security, since almost all security mechanisms protecting the communication partners rely on authenticated users and shared secrets only known to the communicating parties. Consequently, deriving shared secrets is often performed within or immediately after the authentication process. Since this thesis is not only concerned with authentication, but also network security, the topic of key management as part of the authentication process is also covered. But for now, let us continue with the other A's in the triple A standard. We shall come back to authentication shortly.

2.1.2 Authorization

Authorization is often considered to go hand in hand with authentication. Once a user is authenticated, access to the network is granted, and he or she can use all services on the network. For a home or enterprise network, this might be true. However, sometimes more sophisticated mechanisms for authorization are required. Consider a hotspot provider offering paid access to the Internet. Depending on the user's subscription, a user could be restricted to use only limited bandwidth or is not allowed to do Voice over IP (VoIP) calls. A better paying costumer could be offered higher Quality of Service (QoS) or unlimited usage.

These new requirements demand more than just a simple look up table to make the decision whether to grant access or not. A consistent, effective and scalable solution is to use a policy framework. A policy framework allows to manage the many different types of policies, such as security policy, roaming policy, services policy and so on. Having a reliable framework in place is essential to make consistent decisions. All attributes of a user, such as the identity, the network usage, the subscription type etc., are stored in a user profile that is loaded and checked against the policies each time a user requests a service.

2.1.3 Accounting

Last but not least the third "A" in the triple is accounting. Accounting is often considered to be all about billing. Collecting information about the user's network usage, counting calls and/or data packets and generating a bill that is sent to the customer is believed to be accounting. This is true, yet there is more than just billing the customer. The main applications of accounting are:

- **Auditing**
  To know what and when a resource or service has been used by whom and providing mechanisms to prove the correctness of this information is the core application of accounting.

- **Cost allocation**
  Understanding the cost structure behind a telephone call or a data session is important for optimizing the network architecture.

- **Trend analysis**
  In order to plan for future capacity requirements, a detailed trend analysis is essential.

Among these three main purposes of an AAA accounting system there are monitoring and reporting tools, resource consumption measurement and, an important one, a policy checker. All these applications have different security and redundancy requirements and thus ask for carefully designed network architectures.

2.1.4 AAA Servers

Now that we have seen what the three A's mean and what the differences between them imply, the question could be asked why they are supposed to run on a single AAA server. The term single server stands for the logical entity server which could, of course, be distributed over several machines, since fail-safe redundancy and load balancing can be handled in this way. Therefore, the architecture of such an AAA server park can be very complex. AAA servers have to be somehow integrated with the business entities of the company. Authentication often involves access to the customer database, authorization requires policy servers and accounting is closely coupled to the billing servers. Yet, a generic AAA architecture has not yet been standardized.

The reason why one speaks of the AAA triple, AAA servers or AAA protocols, despite the fact that they solve different problems, is because the three A's are closely linked. A good example that shows these dependencies are prepaid services. Prepaid services allow users to use services without having a long-term contract with a provider. One simply buys a value-card (or calling-card, prepaid-card) for a certain amount that can be used until the amount is consumed. Let us look now at the chain of actions that take place when a user wants to make a call with a prepaid calling card.

First of all, after the user has inserted the prepaid SIM into the mobile phone and powered it up, the phone tries to log on to the network using the credentials and keys stored on the SIM card. This is where the authentication mechanisms are used. When the customer now begins to make a call, it has first to be checked with the accounting server if there is still enough credit associated with that specific account. Depending on this information, authorization is granted or not. After the call, the cost of the call need to be recovered.

This example shows how the mechanisms of authentication, authorization and accounting have to go hand in hand, and this should all take place within seconds. Thus, an AAA server can be very complex with strong requirements with respect to scalability, reliability and robustness.

2.2 What is Authentication

The origin of the word authentication comes from the Greek word *authentes* which means author. This already indicates that authentication has to do with something or someone coming from an author or an original source. The actual meaning of the word refers to the process of verifying or confirming whether someone or something is who or what it claims to be. In the context of computer networks, the simplest form of authentication is to provide a username and a password. The

assumption here is that only the person with the provided username knows the secret password and thus must be who he claimes to be.

However, authentication did not originate from computer networks. In the past, emperors used seals to authenticate important letters or contracts. Seals served as proof to the recipient that a letter was authentic, assuming that the seal could only be applied by the sender. Furthermore, the seal also protected the content of a letter as long as it was not broken. If the seal is still intact, no one can have altered or read the content, since this would have destroyed the seal. Relying on the safety that a seal gave, emperors were able to start attacks and win wars. Nowadays, in the digital world we live in, delivering proof of authenticity of a message has become one of the first and most important security requirements.

As we have seen in the above example of the seal, authentication is used in different contexts. On the one hand, the seal proved that the sender was who he claimed to be, and on the other hand that no one has read or altered the content. These are actually three different aspects. Ensuring that the no one can read the content is often referred to as confidentiality, and making sure that the content is still the same as it was when it was sent is known as the integrity of a message.

Authentication, however, is not only used for authenticating users. Actually, it refers to validating a claimed identity. People and network devices have an identity. One has to be specific when performing user- and/or device authentication. The term client authentication is also often used, which in the majority of cases means either user- or device authentication, without further specification. There are many different terms used with the same meaning, and on the other hand, there are several meanings of one term within the same context. In order to achieve a common understanding, we will now separately look at the different aspects of client-, user-, device- and message authentication.

2.2.1 Client Authentication

As mentioned above, the term client can have two completely different meanings. On the one hand it can refer to a human user, which would require user authentication, and on the other hand it can refer to a device. Often one does not further specify which one is meant, and simply talks about the client. Nonetheless, the two are seldom the same. Let us have a look at two scenarios, where the difference between device- and user authentication becomes clear.

2.2.1.1 User Authentication

Think of a computer room at a university where all students can log into the computer system and have access to all sorts of services. The computers are all locally connected via wired cables, all connected to a switch or a router. The students log in with their personal credentials consisting of a username and a password. These credentials are then checked against a database, and, if valid, the user is authenticated. Nevertheless, the actual device, here the computer terminal, is NOT authenticated. The user simply trusts the network setup of the campus. We see in the next Section what the consequences of non-authenticated devices are.

2.2.1.2 Device Authentication

An example where it is actually not important who is using the system but device authentication becomes crucial would be a cable modem connected to the ISP (Internet Service Provider). These devices are manufactured in a way that each device contains an individual set of credentials that cannot be changed. When the device connects to the ISP, these credentials are used to check if the device is allowed to access to the network, i.e. if there is a valid contract with a customer. Here, it is not important (from the viewpoint of the provider) who is actually using the connection, since this is often regulated in the contract (providers often do not allow the connection to be shared with others, e.g. the customer's neighbours).

Device authentication can be achieved with digital certificates or cryptographic keys that are stored in the device prior to service initiation, and should be transparent to the user. It has to be assured that these credentials are protected and cannot be altered.

To see what the dangers of non-authenticated devices can be, we consider the campus computer room example again. Even if it might seem a bit unlikely at a campus, an attacker could set up a compromised terminal. As soon as an unaware student types in his credentials, the attacker can easily steal the student's credentials (username and password) just by storing the log-in data. Afterwards, the attacker gains access to the network with the stolen credentials.

Non-authenticated devices in networks can be a security risk, because it cannot be controlled if the attached devices conform to specific security policies. In an enterprise network, the administrators are responsible for correct configuration and maintenance of every device connecting to the network according to the company's security policies.

2.2.2 Message Authentication

While client authentication ensures that the end points during communication are those that they claim to be, message authentication makes sure that the message itself has not been tampered with. An authentic message is guaranteed to be the same at the receiver as at the point of entry. Message authentication is also considered as a data integrity protection. These mechanisms prevent the so-called Man-In-The-Middle attack (MITM), where an attacker sits between the two communication partners and reads and/or alters messages.

However, message authentication is not of further importance in the context of this thesis. Therefore, for the rest of the text, the term authentication refers to either user- or device authentication, or client authentication in general.

2.3 Authentication Mechanisms

Since the intention of this thesis is not only to describe the latest authentication and security mechanisms, but also provide a general view of the topic itself, we look now at the different mechanisms that exist. In the Sections above we have read about users authenticating themselves by providing a username/password pair and devices being authenticated trough digital certificates. These are only two different approaches to authentication and of course others do exist. Instead of giving further examples of how the authentication problem can be solved, we classify the different mechanisms.

The Internet Architecture Board (IAB) has published an Internet draft called *"A Survey of Authentication Mechanisms"* [2] which provides such a classification. According to the authors, there exist three fundamental criteria:

1. Authentication based on something only the authenticating party owns, such as a physical hardware token or a card.

2. Authentication based on something only the authenticating party knows, such as a secret or a password.

3. Authentication based on something the authenticating party is, such as a physical characteristic of the link it is attached to.

Based on these criteria, the following seven classes have been proposed by the IAB:

1. **Password in clear text**

   This form of authentication is the oldest and simplest method. The user simply provides his credentials in the form of a username/password pair. The credentials are then sent unprotected across the link to the verifying server. The server then looks up the username in a database or a password file to verify the password. Apparently, this method is vulnerable to lots of attacks. Even a simple sniffing tool could just record the username/password pair and start a replay attack using the stolen credentials. Other possible attacks are online password guessing, where the attacker guesses the password for a specific user, and offline dictionary attacks, in case a hash of the password is stored on the server instead of the plain password. The former password guessing attack can be alleviated by restricting the number of retries that a password can be entered. This, however, opens the doors for so-called Denial of Service (DoS) attacks.

2. **One-time Passwords**

   As the name suggests, a one-time password can only be used once. The user either owns an ordered list of passwords or a processor card, such as a SecureID [3] token, which generates a new password based on a predetermined algorithm. This method was invented to prevent replay attacks. Now that the password is only valid for one log-in, it can be transmitted in clear text without any worries. The before mentioned SecureID card from RSA Security [3] generates a time-dependent code every minute that is based on a hardcoded secret key (128 bit) and the AES algorithm. The code is then sent together with the username and an additional user password. This form of authentication is known as two-factor authentication and considered very strong, since authentication is based on something the user knows (the PIN or user password) and something only the user owns (the token card containing the secret key). Nevertheless, one-time passwords are still open to replay attacks during the time between the generation of the code and the generation of the next one. Furthermore, this method is still vulnerable to Man-In-The-Middle attacks (MITM).

3. **Challenge / Response**

   This method was designed to avoid possible replay- and sniffing attacks. The server (i.e. the authenticator) sends a challenge, typically a random number, to the client. The client then generates a so-called hash or digest out of the challenge and the secret key. Hashing algorithms are based on mathematical one-way functions which guarantee that is computationally infeasible to reproduce the input of the hash function, given the output, and that two different inputs cannot produce the same hash value. Hash functions that also use secret keys as the input are called secret hash functions.

The hash value is then sent as the response of the challenge to the server, without encrypting it. The server verifies the response by calculating the same hash digest. This method relies on two assumptions: first, the challenge that the server generates must have sufficiently large entropy, i.e. be random enough. If the challenge repeats itself regularly, then offline dictionary attacks[1] become possible. Second, the password or secret key must be strong enough, for the same reason. If a the password is too short, an attacker can run an offline dictionary attack just by sniffing one challenge/response packet.

4. **Anonymous Key Exchange**

   If the communication channel between the client and the authenticating server is protected by added encryption and integrity protection, then everything can be sent in clear text. This allows any legacy method, such as password in clear text, to be used securely. However, in order to achieve a secured channel prior to authentication, the involved parties need to have a shared secret. Diffie and Hellman invented a very clever mechanism, known as the Diffie-Hellman method [4], which allows to establish a shared secret between two parties that have no prior relationship with each other. The method is based on public keys that can be sent over insecure channels. The method however, is vulnerable to MITM attacks. This means that in the rawest form of Diffie-Hellman there is no possibility to verify the identities of the involved parties. When this is the case, one speaks of anonymous key exchange.
   One possible solution to the MITM problem is to use digital certificates that prove that the public keys belong to the claimed identities using a trusted, third authority. We shall see in more detail how digital certificates work in Section 2.6.3.

5. **Zero-Knowledge Password Proofs**

   Zero-Knowledge Password Proofs (ZKPPs) are based on so-called zero knowledge proofs, where an entity knowing a secret must prove that it knows the secret, without giving any information about the secret itself. This might sound paradox, but is possible [5][6]. However, most common methods are very resource intensive and heavily patented, thus not (yet) very popular. The earliest and simplest ZKPP is called EKE (Encrypted Key Exchange [7]). EKE enhances the standard Diffie-Hellman protocol by additionally encrypting the public keys that would be sent in clear text otherwise.

[1] A dictionary attack refers to a brute-force attack where one tries out all kind of likely passphrases using some exhaustive, pre-defined list of words (hence dictionary) in order to determine the secret key. If the attack can be performed without involving the system, then one speaks of an offline attack (compared to an online attack).

6. **Server Certificates plus Client Authentication**

   The idea of this method is that if one of the involved parties can authenticate itself, then a secure channel can be established and thus allow the client to use any legacy authentication method. This kind of authentication has gained popularity trough SSL (Secure Sockets Layer), which is used for e-shopping on the web. While SSL does not provide mutual authentication (only the server's identity is validated), true examples of server certificate plus client authentication are EAP-TTLS (see Section 3.5.3) and HTTP over SSL (HTTPS).
   In this scenario, the server presents its certificate to the client which then can verify that the server that sent the certificate is actually the one to which the certificate was issued. This can only be achieved using a trusted third authority (the Certification Authority, CA for short). More details on certificates and CAs follow in Section 2.6.3.

7. **Mutual Public Key Authentication**

   This form of authentication is probably the most secure if handled properly. Compared to server certificates plus client authentication, in mutual public key authentication the client also is in possession of a digital certificate. This allows mutual authentication (i.e. authenticating the server AND the client) in cases where the two parties never had any prior relationship or contact with each other.
   There are however two reasons, why mutual public key authentication can be problematic. The first being the complexity that is involved in issuing, maintaining and deploying the certificates and the PKI system that is needed. The second problem is protecting the client's private key. As the name suggests, the private key is a secret key that must be kept private all the times. While at the server side it is possible to protect the (server's) private key with tamper resistant, specialized hardware, protecting the client's private key is much more difficult. Client devices are less sophisticated and can be stolen. A stolen (or compromised) client's private key leads to a compromised certificate which can be a real security risk, depending on the authority associated with the certificate. Fortunately, there are two common solutions to this problem. Either the private key is stored in a separate, PIN-protected token, or the private key is derived using a user password as the seed to a cryptographic process that leads to the private key. The latter method however, is vulnerable to dictionary attacks, while the former introduces additional cost for the hardware tokens.

As one can see, there are several possibilities to achieve the same goals: unilateral or mutual authentication. There is no ultimate solution, however. More sophisticated mechanisms are also more complex and thus introduce additional cost. It has to be clearly specified which level of security is required and what attacks are still possible. There is no completely secure system available today.

Security thus has to deal with risk management and there is mostly a trade-off between security, usability and cost.

2.4 Authentication Models

In this Section we present two models for authentication that are very common. They serve as a basis for further discussion and introduce some terms that are often used in the context of AAA systems.

2.4.1 Two-Party Authentication Model

A typical setup for this model would be a client that is connected to a server over a direct line without any intermediate node. Only two parties are involved in the authentication process, hence the name. If we look at the wireless LAN example, the client would be a WLAN enabled device, and the server acts as the access point, the authenticator, and the authentication server in one. This means that the server performs several tasks: managing the link between the client and the access point and authenticating the client device or user.

This setup is typical for small networks where the authenticator (i.e. the access point) can itself decide to grant access to users or not. However, in large networks, especially in public hotspot setups, it would be administratively almost impossible to let the access points make this decision. This would require storage of the complete user database in the access point, which is almost impossible to maintain for a large user base. The solution to this is to have a central authority making the decisions. This model is known as the three-party authentication model introduced in the following text.

Another disadvantage of the two-party model is that there must be a direct link between the client and the authenticator. As soon as another node, such as a gateway or a proxy is involved in the communication, this model does not suffice anymore. An intermediate node not belonging to the operator cannot be trusted per se, and thus authentication becomes infeasible.

2.4.2 Three-Party Authentication Model

Large networks typically have more than just one so-called Point of Presence (POP). In the case of public hotspots, every single hotspot, i.e. the access point is a POP. POPs are located at the edge of the network and have to authenticate users requesting access to the network. As mentioned above, it does not make sense to have the user database that is used for validation of the credentials locally

present at each POP, since the amount of possible users can be up to several millions. This would not only be almost impossible to maintain, but would require very resource intensive hardware at each POP. Having a central authentication server and easy to maintain, low cost hardware at the POPs is the architecture of choice.

Since we now have three participants in the authentication path, this model is referred to as the three-party authentication model. The three parties are shown in the following diagram and described after Figure 2-1:

Figure 2-1 Three-party authentication model [8 p. 6]

- **The supplicant** is the user or the device trying to get access to the network. For public hotspots this could be any device equipped with embedded wireless LAN technology.
- **The authenticator** on the edge of the network representing the POP. This is the device the user connects to. It forwards the authentication request to the authentication server. The authenticator itself does not have any authority. In terms of the AAA model the authenticator is called Network Access Server (NAS) and acts as an AAA client.
- **The authentication server** is a central database with a list of all valid users and their credentials. This is the entity that actually decides whether a user will get access to the network depending on the credentials presented. In AAA methodology this entity is referred to as the AAA server.

The authentication process related to Figure 2-1 is performed as follows: the user (resp. supplicant) connects to the authenticator and asks for access. The authenticator then forwards the request to the authentication server which will decide whether access is granted. On success or failure, either result is sent back to the authenticator and the user.

Typically, the authenticator and the authentication server are in the service provider's network. The authenticator is located at each POP, and the authentication server at the core of the network. It is possible, more in small networks though, to have the authenticator and the authentication server co-located. This would then be referred to as the two-party authentication model seen before.

We discuss briefly the protocols used between the three participants in an AAA setup as presented above.

2.5 Authentication Protocols

Due to the fact that different media are involved in the communication between the three parties in the three-party authentication model, different protocols must be in place to allow end-to-end communication between the user and the authentication server. If we look again at Figure 2-1, we see that there are two green arrows, which stand for the protocols between the corresponding parties.

Suppliant – Authenticator/NAS Communication

Depending on the access technology offered by the service provider, the communication protocol between the client and the NAS can be of various types. This access link is in most cases an insecure, open wireless medium. If we look at a hotspot provider offering public WLAN access to its customers, there would be Wi-Fi technology, i.e. 802.11 in place. It offers a physical channel and a link layer protocol, and since the wireless access technologies such as 802.11 WLAN have their own framing mechanisms, no further layer 2 protocol such as Point-to-Point Protocol (PPP) is needed. Nevertheless, after the initial authentication process, an Internet Protocol (IP) level service should be established.

Authenticator – Authentication Server

The communication link between the authenticator (or NAS) and the authentication server (AAA server) typically lies within the operator's private network. This could be a wired, leased line dedicated for the authentication process alone, and hence can be trusted. Communication is based on the standard User Datagram Protocol over the Internet Protocol (UDP/IP) or the Transmission Control Protocol (TCP/IP). The protocol carrying the authentication messages has been standardized for reasons of inter-operability. The most widespread AAA protocol is RADIUS (which stands for Remote Access Dial in User Service [RFC2865]). Diameter [RFC3588] is a newer protocol that overcomes the limitations of RADIUS. The two AAA protocols RADIUS and Diameter are described in the Chapters 3.6 (RADIUS) and 3.7 (Diameter).

The assumption that there is only one hop between the NAS and the AAA server can be true, but several hops over different AAA proxies are possible. In that case the line from the NAS to the AAA server may no longer be private and trusted, which would require additional security mechanisms to protect the communication, e.g. the Internet Protocol Security (IPSec) for secure communication.

2.6 Public Key Cryptography

As we have seen in the previous few Sections, authentication is either based on a shared secret, i.e. a password, a symmetric key or on public keys used in combination with digital certificates. It has also been mentioned several times, that digital certificates introduce additional complexity and cost. We now present the fundamental mechanisms behind these methods in order to understand where this complexity comes from and how the cost arises.

Before getting into public keys for authentication, we look at symmetric keys (i.e. secret keys, user passwords) and their limitations.

2.6.1 Symmetric Keys

The term symmetric key comes from the fact that the same key is used for encryption and decryption of data, hence the word symmetric. As a consequence of this, typically both parties, i.e. the server and the client, share the same key. Symmetric key cryptography is also known as private key cryptography or secret key cryptography.

An easy to understand example of symmetric key cryptography is the Caesar Cipher, named after Julius Caesar. In this easy to break cipher, each letter of a message is replaced by another letter that is a fixed number of positions away in the alphabet. The key in this cipher is simply the number of positions that letters need to be shifted. While this is an early and very simple cipher, much more sophisticated methods that are much harder to break do exist.

The main problem with symmetric key cryptography is the distribution and protection of the secret keys. While symmetric keys are well suited e.g. for encrypting files on a hard disk to protect them from unauthorized access or encrypting a communication channel between two parties, they become impractical if several entities must be able to decrypt the encrypted data. Sharing the same key with many users increases the chance that the key might become compromised. The only way to make the system secure again, is to re-encrypt the data and distribute new keys to all parties involved. This makes symmetric key cryptography impractical for secure communication over the Internet. The more users involved in secure communication, the greater the problems of distributing and protecting of the secret keys.

On the other hand, symmetric key cryptography is much less resource intensive compared to asymmetric methods (presented in the next Section). This is why symmetric methods are used for securing the communication channel between the access point and the client device in WLANs. Since every single data packet needs to be encrypted and decrypted, the performance overhead of a cipher

becomes an issue. However, if symmetric keys are used, the keys are typically only used temporarily for one session and thus newly generated each time. This decreases the probability that a key becomes compromised.

The problem of transferring the keys over an insecure channel can be solved using techniques such as the Diffie-Hellman algorithm [4]. However, such methods are often vulnerable to MITM attacks. Asymmetric keys (public key cryptography, respectively) overcome most of the problems of symmetric keys and are thus better suited for authentication in large networks.

2.6.2 Asymmetric Keys

As the name suggests, in asymmetric key cryptography two distinct keys for encryption and decryption exist. One of them is commonly referred to as the private key that needs to be kept private (securely), and the other is called public key. The public key does not contain any secret information and can thus be distributed to anyone without protection. This solves the problem of securely distributing the keys as it is the case in symmetric key cryptography.

The most common algorithm for generating the key pair is the RSA algorithm [9] which was published by Ron Rivest, Adi Shamir and Leonard Adleman at MIT in 1977 (RSA are the initial letters of the authors' surnames). RSA is based on large prime numbers. If implemented correctly, it is infeasible to derive the private key given the public one. The mathematical background of RSA is beyond the scope of this thesis.

There are two different scenarios where public key cryptography can be used: for digitally signing documents and encrypting messages. Digital Certificates are based on digital signatures, see the Section 2.6.3. Let us now discuss the mechanisms behind digital signatures.

2.6.2.1 Digital Signatures

Digital Signatures can be seen as the digital equivalent of a written signature. A digitally signed document is guaranteed to be signed by the claimed signer, i.e. the one who owns the private key. However, digital signatures go further than traditional ones by also ensuring that the data itself has not been altered after signing. The following diagram shows the whole process of generating and verifying digital signatures:

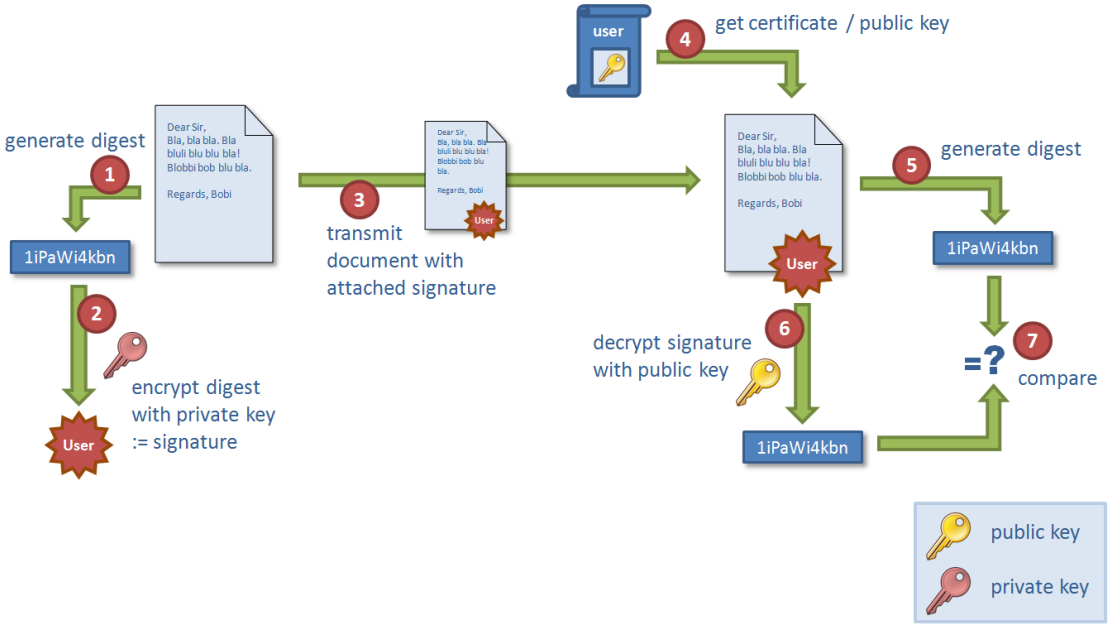

Figure 2-2 Digital Signature Scheme

1. A digest is generated out of the document to be signed.
2. The digest is encrypted using the private key of the signer.
3. The signature is attached to the document which is sent to the receiver. The signature contains the encrypted digest (the signature) and additional information about where to get the certificate which includes the public key of the signer.
4. The receiver downloads the certificate of the signer containing the public key either directly from the signer or from the URL specified in the signature information.
5. The receiver also generates the digest, using the same algorithm.
6. Additionally, the receiver decrypts the signature with the signer's public key, leading to the digest, which then ...
7. ... is compared to the digest that the receiver computed separately (5). If they are the same, then the document is guaranteed to be signed by the owner of the private key and has not been altered after having been signed.

The reason why certificates are used is because on the one hand, certificates contain the public key, and on the other hand, they proof that the key actually belongs to the sender. Certificates are also signed documents, only in this case the signer is a trusted third authority (the Certification Authority, CA). More details on digital certificates follows.

2.6.2.2 Asymmetric Key Encryption

The second use of public key cryptography is message encryption. Here, the message is encrypted using the receiver's public key. The only person that is able to decrypt the message is the one owning the private key. Again, digital certificates are used to ensure that the public key actually belongs to the receiver. There are further details about message encryption mechanisms which are omitted here.

2.6.3 Digital Certificates

We have seen that digital certificates are used to prove the authenticity of a public key. This means that a certificate guarantees that the public key belongs to the identity in the certificate (e.g. Bob). To make sure that the certificate itself has not been altered by an attacker, the certificate is signed by a trusted third party, i.e. the certification authority, or CA for short.

Issuing and management of certificates are complex task which involves large administrative overhead, since everyone using these certificates inherently trusts the issuer. There are many requirements that need to be met in order to get a certificate. The issuing authority must make sure of the identity of the requestor, which often requires the issuer to provide a passport or other official documents.

The following picture shows an abstract image of a digital certificate showing the most important elements that every certificate includes:

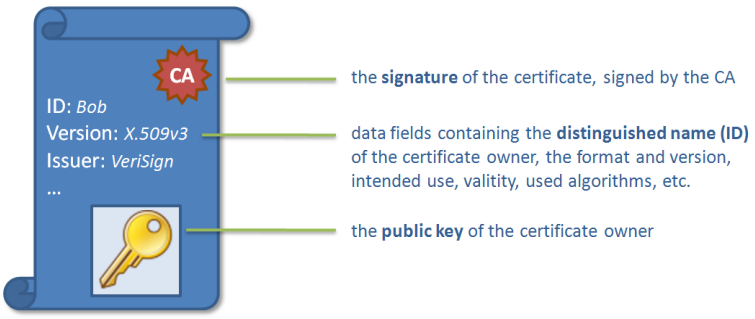

Figure 2-3 Abstract view of a digital certificate

Every certificate includes a distinguished name, which can refer to a person, a company or a device. In any case, it must be the owner of the private key corresponding to the public key, which is also included in the certificate. A certificate can hold a variety of other data fields and attributes, such as a version number of the certificate, a serial number, extensions, algorithm identifiers and others.

Some of the entries, including the distinguished name, the issuer (of the certificate), the validity and the public key itself are digitally signed by the issuing authority. This guarantees that the information in the certificate is genuine.

2.6.4 Public Key Infrastructure

The task of managing digital certificates does not only involve the issuing of the certificates. Once the certificates are signed by the CA, there must also be mechanisms in place that make the certificate (and the public keys) available to anyone wanting to use a specific certificate for secure communication. Scalable and robust solutions are a must.

Another issue with certificates is the management of the lifetimes of certificates and so-called Certificate Revocation Lists (CRLs) which allow revocation of a specific certificate that may have been compromised. This can happen due to loss or compromise of the private key, or the termination of a relationship between a company and its employees. CRLs must also be accessible at any time, since they must be accessed each time a certificate is validated.

As already mentioned, the CA is the third party whom the communication partners using the certificates must trust. The typical situation where certificates are used is when two communication partners want to communicate securely without having a prior relationship of any kind and have never seen or met each other before. The CA as the trusted third party helps the two by guaranteeing that the public keys used for secure communication actually belong to the identities claimed. The question now is how this trust relationship with a CA is managed.

Unfortunately, the answer is not that straight forward. The basic idea however, can be described as follows. Let us assume that we have received a certificate from our communication partner (e.g. an e-commerce server establishing a HTTPS connection). In order to check whether the presented certificate is valid, we have to check the digital signature of the certificate. To do that, we need the public key of the CA, which again needs to be verified. As one might guess, this requires another certificate for verifying that the public key actually belongs to the specific CA. Continuing in this way, the result is a chain of certificates, each verifying the signature of the certificate one level lower in the hierarchy. At the top of this hierarchy stands a so-called root-CA. The root-CA verifies all certificates down the hierarchy, thus is the only one that must be trusted. This reduces the amount of certificates that need to be trusted to a relatively small set of root certificates. These root certificates normally are pre-installed in the clients. In case of the e-commerce example involving an SSL server presenting its certificate to the client, which in this case is a customer using a standard web browser, the root-CAs are installed in the browser itself. One should be aware of that when one

downloads a new browser from an unknown source. A compromised browser installation packet might contain a manipulated set of root-CAs. This would allow attackers to use invalid certificates that will be automatically accepted by the browser without the user's knowledge.

As we have seen above, a certificate itself is almost worthless without the whole infrastructure in the background. Systems that are able to issue, manage, distribute, and validate certificates are known as Public Key Infrastructures or PKI for short. Such a PKI has to meet very high requirements with respect to security, robustness and scalability, and requires constant maintenance. Thus, a PKI system is not only complex, it is also very costly. For that reason, using digital certificates should be carefully considered. Certificates provide robust and secure means for secure network communication, but on the other hand they require a costly PKI. This includes qualified IT staff, robust server hardware, registration and licence cost and much more. According to VeriSign [10], one of the largest PKI solution providers, a company requiring every employee to use certificates for authentication must expect to pay up to 100 USD per user yearly, depending on the number of employees.

Of course, there is much more to know about PKIs than just the basics discussed above. For the purpose of understanding the basic mechanisms of certificates however, this should suffice.

2.7 Key Management

The goal of authentication in network access is to prove that the identity that one claims to be is true. As we will see in the next Section about wireless security goals in general, cryptography is an important tool that helps to achieve this goal. Cryptographic methods rely on shared secrets that are either known by the involved parties prior to communication or established with the help of a trusted third party.

The generation and distribution of the secret key has to be performed very carefully, since authentication and also channel encryption are based on the fact that the key is only known to the two communicating parties. If keys or certificates become compromised, any security mechanism based on these keys is worthless.

In order to understand the available frameworks that provide authentication and key establishment methods, such as EAP, it is necessary to first understand the basic problems they try to solve. The next Section gives a short overview of the main issues of key management.

A key management framework provides functionality that allows the generation, distribution, control, record keeping and destruction / revocation of cryptographic material. Generally, one can separate the following main issues:

- **Selection of** the appropriate **cryptographic algorithms** and key sizes.
- **Key management policy**, defines how the keys are used, how long they are valid and how compromised keys can be made invalid for further usage.
- **Key establishment schemes**, deal with the problem of generating and distributing the cryptographic material.

The first two listed aspects of key management, namely the selection of the algorithms, key sizes and associated policies, depend on network management and security requirements. Since this is typically defined by network designers and administrators, we shall not go into further details here. Key establishment mechanisms on the other hand, are closely coupled to authentication protocols. Generating and distributing keys can be done in different ways, which can be categorized as follows:

- **Key transport**
  Key transport is used whenever a key (or key pair or certificate) is generated by one single entity (i.e. the server) and the material has to be transported to the other entity (the client).
- **Key agreement**
  In this scenario, both the client and the server contribute to the key generation. The derived key is never transmitted over the communication channel. A very popular method for deriving a shared secret without transmitting it is the Diffie-Hellman mechanism, which relies on public key cryptography.
- **Manual key establishment**
  Key transport and key agreement often rely on having a pre-shared secret or at least requires some manual intervention of an administrative authority. Manual key establishment is mostly done over an out-of-band channel, such as sending a password or a security token with postal services, acquiring the credentials over the phone, or using a different access technology (e.g. wired Ethernet, USB-Dongle, Bluetooth, Infrared, etc).

It is important to note here that a shared secret cannot be established, whether through key transport or key agreement, out of nothing. In order to transfer a key securely, one has to use a protected communication channel, which requires already sharing secrets. On the other hand, key

agreement with Diffie-Hellman requires some mechanisms for authentication of the involved parties to avoid MITM attacks. All this can only be achieved with pre-shared symmetric keys or public key cryptography. Digital certificates do not work without a trust relationship to a third party (i.e. the CA).

Having pre-shared keys also requires the two parties to have established some sort of trust, since they both share a secret. A trust relationship of any kind, be it a contract with a service provider, a certification authority signing certificates, or in its simplest form, getting a username and a password from an administrator is not only a requirement for key management, but also for authentication in general. We look at how key management is performed in the EAP framework in Section 3.4.3.

2.8 Goals of Network Security

Before we focus on the log-in process in (wireless) LANs, we begin by identifying the security goals that need to be met. These are: confidentiality, integrity, availability, authenticity, non-repudiation, and accountability [11]. Depending on the application domain, the transmitting media and security policies, these goals can be weighted individually and further requirements, such as anonymity, might be added. However, for network security in wireless LANs, these goals can be considered sufficient.

Let us now define what each of them means, and how they can be achieved.

2.8.1 Confidentiality

Confidentiality has been defined by the International Organization for Standardization (ISO) as *"ensuring that information is accessible only to those authorized to have access"* [12]. In the context of computer networks this involves the following key aspects:

- Protection of the message content against being read by a non-authorized third party.
- Anonymity of the communicating parties. This goal is sometimes listed as a separate security goal. Anonymity here only requires that no one listening and analyzing the traffic can identify who is actually communicating.
  Anonymity can also mean that the sender and the receiver do not have to present their real identity to each other, i.e. stay anonymous. This, however, contradicts authenticity, and thus cannot be achieved in wireless networks requiring authentication.
- Protection against analysis of traffic- , usage- or signaling data.

These goals are met using techniques of modern cryptography, i.e. encrypting and decrypting the message content, and/or control and signaling data.

2.8.2 Integrity

Data integrity refers to the requirement that messages are being received as they were sent and cannot be altered by an attacker. However, since it cannot be avoided that messages can be changed while on the network path, it is only possible to detect whenever a message has been changed.

One has to be specific though what kind of changes we want to detect. There are transmission errors that occur due to a noisy channel causing single bits to flip, and on the other hand there could be an attacker trying to change a message deliberately. The former can be corrected using checksums and redundancy. If a checksum that is computed over the entire message by the sender is not the same when computed by the receiver, then either the message can be reconstructed to some degree depending on the amount of redundant data that has been sent, or the whole message needs to be retransmitted.

To prevent attackers to deliberately alter a message, so-called Message Authentication Codes (MAC) or Message Integrity Codes (MIC) are used. These are based on secret hash functions.

Message authentication is often used as a synonym for data integrity. However, an authenticated message additionally must be proven to come from the right sender. This requirement is defined as a separate security goal, namely authenticity.

2.8.3 Accountability and Non-Repudiation

Non-repudiation ensures that an authenticated user cannot disclaim the transactions that he has made. It has to be verifiable to a third party that a certain transaction has actually happened in case the user denies that fact.

Accountability plays a special role in public wireless networks where customers need to be charged for their network usage. The operator of the network must be able to provide a proof of the validity of the user's service usage data. Both, accountability and non-repudiation are achieved by logging all transactions. Further, the user and the network need to be authenticated before network access. This however, is the goal of authenticity.

2.8.4 Availability

This security goal requires a network or service to be available every time a request for it is made by an authorized user. The Denial-of-Service Attack (DoS-Attack) is a possible attack where an attacker tries to overload the system either by sending thousands of requests per second or to use up the system resources of the recipient's system. That way, the service is no longer available. However, 100% availability is simply not possible due to the fact that earthquakes and other catastrophes could potentially put down any system. Nevertheless, mechanisms that detect DoS-Attacks should be in place and important components should duplicated (in case one component stops running, the redundant system must resume) to guarantee availability as high as possible.

2.8.5 Authenticity

As mentioned before, authenticity requires that the sender of a message can be verified. In order to achieve authenticity, both communication partners should be authenticated before the actual communication takes place. This is not as easy as one might think. It becomes even trickier when the two communication partners have never seen (resp. communicated with) each other.

Even if there are mechanisms providing data integrity, it would still be possible for an attacker to impersonate the receiver, then read and/or change the message, and send it out to the recipient, claiming to be the original sender. This form of attack is known as the Man-In-The-Middle-Attack (MITM). We see later how this attack can be prevented, again using cryptography.

Almost all security goals that we have described above rely on authenticated users and shared secrets between the involved parties. The distribution of the shared secret, known as key management, becomes a critical issue in wireless networks, since all traffic can easily be intercepted and listened to. Key management is often performed within or just after the authentication process. Once the keys are distributed, all other mechanisms based on cryptography can do their job. While we will not go into too much detail about how each and every encryption method works, we put our focus at the authentication process itself, including key management.

Chapter 3 Standards and Protocols

This chapter introduces the different protocols and standards that are involved in the authentication process. This provides the technical background for further discussion.

We start with a quick look at the history of wireless LANs. We shall see what the issues with the WEP standard were and why today there exist two security standards, namely Wi-Fi Protected Access (WPA) and Wi-Fi Protected Access 2 (WPA2, also known as Robust Secure Network, RSN and 802.11i). Special attention is paid to the enterprise modes of WPA and WPA2, since they are also suited for public WLAN environments. This includes 802.1X port based authentication, the Extensible Authentication Protocol (EAP) and its different flavors, such as PEAP, EAP-TTLS, EAP-SIM and others, as well as the AAA-protocols RADIUS and Diameter.

3.1 History of Wireless LAN Security

Security in wireless LANs has become a very important issue. However, this was not always the case. In the early days of the WLAN technology, one was just not aware of the dangers that opening a network to the outside implied. Most of the networks, even those of enterprises, were just left unprotected and anyone passing by on the street could just log in and do whatever he wanted.

Fortunately, this has changed. Despite the ubiquity of wireless LAN you will no longer easily find an open access point when walking through the streets. Owners of wireless access points are protecting network access with strong cryptographic mechanisms thus also encrypting the traffic in the air.

The WEP standard was the first attempt to provide *'wired equivalent privacy'*, the standard was introduced by the Institute of Electrical and Electronics Engineers (IEEE) in 1997 as part of the 802.11 standard. Its intention was to provide confidentiality, comparable to that of traditional wired networks. Unfortunately, due to its flawed design, it was possible to find out the WEP key just by sniffing ongoing traffic, using simple traffic analysis, standard hardware and some mathematics. A detailed analysis of the flaws in the WEP protocol was carried out in 2001 by N. Borisov, I. Goldberg and D. Wagner [13]. The weakness of the WEP protocol is a good example for the complexity of the topic and how hard it is to design secure wireless protocols.

Unfortunately, the WEP encryption standard was already widely deployed and despite the fact that newer standards became available, WEP encryption was still configured as the default setting in consumer access points for several years.

When the weakness of WEP was identified in early 2001, a newer and better standard had to be developed – without losing too much time. Since the IEEE standards committee was not able to act fast enough, the Wi-Fi Alliance [14] took the existing work that already had been done by the IEEE committee and deployed it to customers in 2003 under the name Wi-Fi Protected Access (WPA). WPA was able to run on existing hardware after a firmware upgrade, which was an important requirement.

The official standard by the IEEE committee was ratified in 2004 as 802.11i, also known as RSN (Robust Security Network). The Wi-Fi alliance [14] has implemented the mandatory parts of WPA that were missing in WPA, and call it WPA2, which is fully interoperable with the IEEE standard. However, older hardware (pre-2003) will not work with the new standard.

3.2 WPA Home vs. Enterprise

WPA, WPA2 and RSN (802.11i) provide two distinct modes of operation. The private mode (WPA-PSK) is intended for home and small business networks and is based on a so-called pre-shared key. Security in private mode heavily depends on the strength of the PSK. WPA-PSK is vulnerable to simple brute force attacks if a simple passphrase is used.

The enterprise mode of WPA (WPA enterprise) is based on 802.1X port based authentication (see Section 3.3), requiring a separate authentication server. The EAP (Extensible Authentication Protocol) protocol is used to authenticate users. In this chapter we look closely at the EAP protocol.

The reason for the existence of two distinct modes of operation is due to the different security requirements in different environments. While in home and small business networks only few users access the network, enterprise networks must be able to manage thousands of clients. In PSK mode, the users themselves are not authenticated. This means that it cannot be determined which user actually wants to access the network. Knowing the pre-shared key suffices. The main issue here is that all users share the same PSK. This implies that when the key becomes compromised, or an existing user must be excluded from the network, the only solution is to change the PSK and re-distribute it to the clients. However, it has to be noted at this point that even if all users share the same PSK, they cannot decrypt the traffic of the others. This is due to the fact that the PSK is not

directly used for data encryption. The encryption key is derived out of the PSK including additional information, such as the client- and server's MAC address. This was different with WEP encryption, where knowing the passphrase revealed the traffic of all users.

In an enterprise environment, the situation is completely different. Employees come and go, hence it must be possible to have a central database that is easily manageable. Such a user database is typically called the authentication server, as we have seen it in the three party authentication model (see Section 2.4.2). A central database stores individual credentials for each user. However, this requires that each user must be identified, or authenticated to be precise. This is where 802.1X port based access control and the EAP protocol come into play.

3.3 802.1X Port Based Network Access Control

The 802.1X standard is part of the 802.1 group protocols. It provides authentication mechanisms by establishing a point-to-point connection for authenticated users or preventing data traffic into the LAN if authentication failed. It perfectly fits into the three-party authentication model presented earlier. Although 802.1X is very popular in 802.11 WLANs, it is designed to provide authentication for all 802 type links. Actually, 802.1X was originally designed with wired networks in mind.

802.1X controls the access link between the supplicant and the authenticator (i.e. the access point). The 802.1X model consists of a switched Ethernet hub connected to the local network offering multiple switches. For every client, a separate (logical) switch is assigned. A switch consists of two ports, namely an uncontrolled port and a controlled port. Before authentication, only the uncontrolled port is open. However, the uncontrolled port restricts communication to 802.1X authentication messages. The controlled on the other hand, does not allow any communication prior to successful authentication, thus protects the network from unauthorized clients.

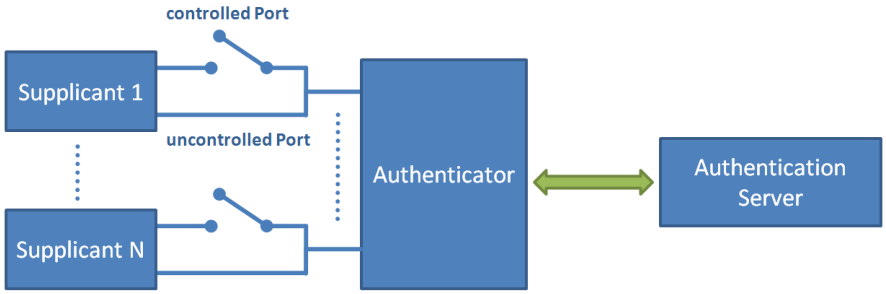

Figure 3-1 802.1X Port based authentication [8 p. 259]

The authentication process is performed as follows [8]: The authenticator starts the actual authentication which is based on EAP, more precisely on EAP encapsulation over LAN (EAPoL). According to the specific EAP method that is used for authentication (details on EAP follow), several EAP messages are then sent between the supplicant and the authentication server, with the authenticator (NAS) in between. The NAS does not understand the method specific EAP messages - it only translates them between the corresponding protocols (EAPOL using 802.1X between the supplicant and the NAS, and typically EAP over RADIUS between the NAS and the authentication server). Besides forwarding the EAP messages the NAS waits for an EAP success or EAP failure message indicating the end of the authentication process. If an EAP success message is received, the controlled port between the supplicant and the NAS is opened, and the client can access the network (or the Internet in case of PWLAN). If authentication fails, the port stays closed, not allowing any traffic into the network.

After authentication succeeded, there is no further security mechanism that protects the communication between the supplicant and the NAS. This is mainly because 802.1X was designed for wired networks, where the wire could not as easily be hijacked as with wireless technologies. What is needed in wireless environments is a key management mechanism to derive encryption keys in the authentication phase in order to encrypt the traffic after the authentication process. An implementation of 802.1X that solves these issues for wireless communication is 802.11i, using EAP.

3.4 Extensible Authentication Protocol

EAP stands for Extensible Authentication Protocol and is designed to provide a generic framework for authentication and network access. EAP itself is not an authentication mechanism. It provides a framework with common functions and negotiation mechanisms to select and run a so-called EAP method. There are currently over forty different EAP methods known. The most prominent ones are EAP-MD5, EAP-TLS, EAP-TTLS, PEAP, LEAP, EAP-FAST, and EAP-SIM, which are all defined separate in IETF RFCs.

EAP was originally designed to run as an extension to the Point-to-Point Protocol (PPP) typically used in dial-up settings. It gained a lot of attention with the growing interest in authentication for wireless networks, i.e. 802.1X. The WPA and WPA2 standards have adopted five EAP methods as their official authentication mechanisms. These certified methods are EAP-TLS, EAP-TTLS/MSCHAPv2, PEAPv0/EAP-MSCHAPv2, PEAPv1/EAP-GTC and EAP-SIM. Each of these methods are discussed in Section 3.5.

Despite its popularity in wireless networks, EAP can run also on wired networks. EAP provides layer 2 security without requiring IP. It is capable to carry authentication messages over protocols such as PPP and the IEEE 802 protocol family, or even AAA protocols, as used between an authenticator (NAS) and an authentication server (AAA server).

We shall look at EAP as an authentication framework in a hotspot environment, which is reflected by the general three-party authentication model presented in Section 2.4.2. This involves EAP over 802.11 Wireless LAN (Layer 2) for client to authenticator communication and EAP over AAA between the authenticator and the authentication server (see Figure 3-2). Note that EAP uses a different terminology than the one we know from AAA. The client we called the supplicant in AAA is called the peer, and the authentication server is an AAA- or EAP server.

Figure 3-2 EAP in the three-party authentication model [8 p. 55]

3.4.1 EAP Messaging

As mentioned earlier, EAP does not perform authentication itself. It is a framework that allows carrying messages of very distinct authentication methods. It is designed to support 'dumb' authenticators between the peer and the EAP server. This means that only the peer and the server communicate with each other, with the authenticator in between just translating the messages between the protocols of the different link technologies, forwarding them without knowing the exact content of the messages. The EAP framework only has four message types, which makes it a very simple, yet extensible protocol.

The four EAP messages are:

- EAP request
- EAP response
- EAP success
- EAP failure

The first two types, namely EAP request and response, are used to transport any information of the chosen EAP method, while EAP success and failure are used to end an EAP conversation. EAP request messages are sent from the server to the peer, and EAP responses are sent back from the peer to the

server. The authenticator, or NAS, in the middle does not understand request and response messages and just forwards them. It only understands EAP success and failure.

EAP request and response messages have a type field that is used to indicate what type of information is transmitted. Examples are '1' for identity, '3' for Negative Acknowledgement (NAK), '4' for MD5 (Message Digest 5) challenge, and so on. However, most type numbers are used to identify EAP authentication methods. For example '15' stands for EAP-TLS and '21' is EAP-TTLS. These type numbers are assigned by the Internet Assigned Numbers Authority (IANA) [15].

EAP has become popular for several reasons. First of all, it is easily extensible (hence the name). The great flexibility comes with the introduction of the type field in the request and response messages. That way, new algorithms for authentication can easily be deployed without the need to change anything at the authenticator or the EAP protocol. On the other hand, EAP is very simple to be implemented by NAS devices (the authenticator) because there are only four message types of which only two must be understood. Furthermore, EAP is media independent, i.e. it is capable of running on a variety of specific link technologies such as PPP, Ethernet or 802.11.

3.4.2 EAP Conversations

Before we look at the specific EAP methods, let us give an example which shows common EAP message exchanges that every EAP method typically performs. For the sake of simplicity, we only have a peer and an authenticator, which is also acting as an EAP server (this would refer to the two-party model presented in Chapter 2). Such an EAP message exchange is called an EAP conversation, and typically looks as depicted in Figure 3-3:

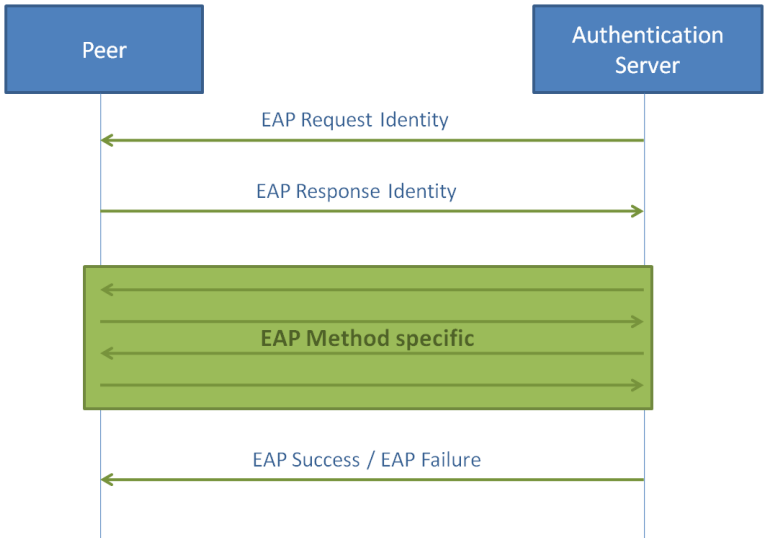

Figure 3-3 EAP Conversation - Message Sequence Chart

- The authenticator starts the EAP conversation by sending an EAP request for identity to the peer. To be more specific, this is an EAP request message with the type field set to 1 (Identity). Additionally, a message to be displayed to the user can be attached in the data field, however, whether the message will be displayed to the user, depends on the implementation peer's supplicant.
- The peer responds with an EAP response message of the same type, here again 1, containing the requested identity in the data field.
- Now it is the authenticator's turn again. It can send any EAP request of any type, and the peer has to respond with an appropriate EAP response message of the same type. The number of messages used for authentication depends on the specific EAP method. If the peer cannot provide the requested information, it sends a NAK (EAP response of type 3) to the authenticator.
- When authentication is completed, a final message, either an EAP success or EAP failure is transmitted to indicate the end of the conversation.

While this example shows the basic mechanisms of an EAP conversation, there are several things to be mentioned. First of all, as done in the example, it is not recommended to transmit the real identity of the peer at this point, since all information is still sent in clear text. The preferred handling would be to send the real identity within a method specific EAP message that provides confidentiality, such as EAP-TTLS, and use a pseudonym in the first EAP request/identity message (e.g. "anonym" instead of the real identity).

Furthermore, the general assumption with EAP is that the authenticator has to start the conversation, which implies that the authenticator knows that there is a peer present, requesting access. This is trivial to detect in wired environments. On the other hand, in wireless environments there needs to be a mechanism for peer detection in place. In 802.1X this is achieved by having the peer sending a so-called EAPOL start frame to a multicast address reserved for 802.1X authenticators. EAP over LAN, or EAPOL, is specially designed to allow EAP messaging over LAN protocols. When the authenticator receives an EAPOL start packet, it can start a normal EAP conversation by sending an EAP request/identity message to the peer. Another solution to this problem would be to have the peer sending an empty EAP-response message to the server.

The last thing to mention at this point is that EAP is a lock-step protocol. This means that the sender of an EAP request message is blocked, i.e. cannot send another request, until it receives an answer of the same message type.

3.4.3 EAP Key Management

We have stated earlier that modern authentication protocols should also provide mechanisms for key distribution. In Section 2.7 we have listed the fundamental functions of key management. We now describe how such functions are integrated into the EAP framework. However, the specific key generation, agreement or transport is performed differently by each EAP method.

EAP was originally designed to carry authentication messages. However, since it provides a generic framework capable of transferring messages to a backend authentication server over AAA protocols, it has also been used for key management procedures. Thus, EAP allows combining network access, authentication and key establishment mechanisms. This flexibility and extensibility makes it a good solution in many environments and scenarios, especially in the mobile world of today.

Consider the EAP protocol with key management in the general three-party authentication model, as depicted in Figure 3-4:

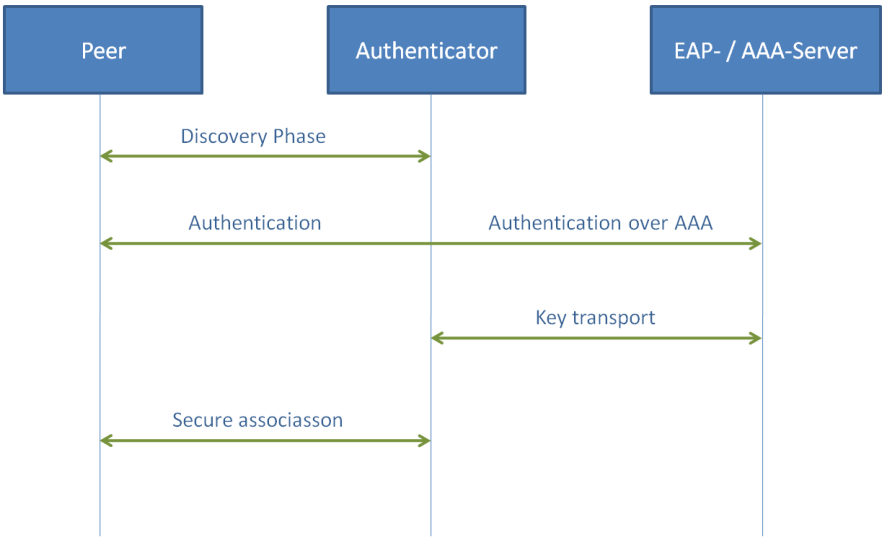

Figure 3-4 EAP key management process in the three-party model [8 p. 57]

EAP authentication and key management has three distinct phases [8]:

- *Phase 0:* **Authenticator discovery phase.**
  In this initial phase the peer tries to detect the authenticator and negotiates its capabilities, such as type of access technology, bandwidth rates and cipher suites. This includes the selection of the appropriate EAP method.

- *Phase 1a:* **Authentication phase.**

  After an EAP method (see Section 3.5) has been negotiated, the EAP conversation starts. The authenticator acts as a pass-through. It encapsulates the EAP messages into an AAA protocol message (typically RAIDUS) to allow communication with the AAA server. EAP authentication involves several message exchanges between the EAP server and the peer which is method-specific. Details can be found in the following Section (3.5). After authentication has finished, a normal EAP conversation would stop here. However, when key management is needed (in order to secure the channel between the peer and the authenticator), then the server and the peer compute the required keys. This includes a transient EAP key (called TEK) for further secure communication between the peer and the EAP server, and the keys required for securing the channel between the peer and the authenticator after the EAP conversation (often called the AAA key). Note that until now, the authenticator cannot be trusted by the peer, since it has just transformed the messages into the corresponding protocols and forwarded them.

  Phase 1b: **Key transport.**

  The computed key material is then sent from the server to the authenticator. Since this communication must be protected, a secure channel between the authenticator and the authentication server is assumed. This link typically lies within the private network of the operator and thus can be trusted. Otherwise, it must be protected, e.g. by IPSec [RFC4301] encryption.

- *Phase 2:* **Secure association.**

  After key transport, the authenticator and the peer share the same AAA key. The authenticator has received the key from the server and the peer has computed it itself. Now that the peer and the authenticator own the same keys they can trust each other. They can setup a secure channel according to the cipher suite negotiated in phase 0.

A strong requirement in EAP key management is that the peer and the network are both authenticated (mutual authentication). This is necessary because not only the operator cannot trust the user, also the user cannot trust the operator per se. It is possible, and not infeasible, that an attacker sets up a fake access point (acting as the authenticator) and simulates an EAP-/AAA server. This way an attacker could obtain the user's credentials and listen to the potentially insecure conversation that follows. This is why mutual authentication is indispensable when key management is performed. However, mutual authentication is based on long-term credentials, which means that

both parties share a pre-configured secret or use private key cryptography in conjunction with certificates.

3.5 EAP Methods

We have now introduced the general EAP framework and mentioned several times that the authentication and key management process is performed by specific EAP methods. In this Section we shall describe how these EAP methods are encapsulated in the EAP framework, and then finally present the most common methods.

In a first step, namely within the discovery phase, the appropriate EAP method needs to be negotiated between the parties involved. The authenticator suggests its preferred method to the peer by sending an EAP request message with the type field set to the corresponding EAP method (note that type numbers larger than 4 are assigned to EAP methods by the IANA [15]). The peer can either accept the suggested method by sending an EAP response of the same type number, or decline it by sending a NAK (EAP message type 3). The server will then propose another method until they agree.

Once a specific EAP method has been negotiated, the server and the peer communicate through EAP request/response messages of the negotiated type. From now on, all messages must have the same type number (except type 3 for NAK), and method specific information is encapsulated in the data field of the EAP message.

When we look at the involved protocol layers on the communication path, the special role of the authenticator (NAS) as a protocol translator becomes visible (see Figure 3-5). The NAS does not have to know anything about the specific EAP method in use. All it has to do is check the message codes whether they are of request/response type, or success/failure. In the first case, it just wraps the message into the appropriate protocol (here into the AAA or IP protocol). If it is an EAP success or failure message, no more messages will arrive and the NAS can end the conversation.

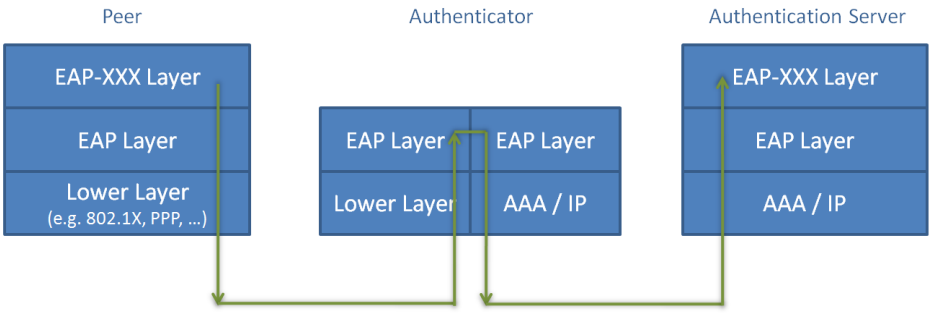

Figure 3-5 EAP layering model. EAP-XXX stands for a specific EAP method [8 p. 243]

This design allows deployment of new authentication algorithms (i.e. EAP methods) without any changes in the authenticator. Authenticators are typically at the edge of the network and also larger in number. This would make it much more complex if these devices needed updates. Only the server and the client need to be changed in order to support new authentication mechanisms.

3.5.1 EAP-MD5

EAP-MD5 is one of the earliest and simplest EAP methods. Authentication is only performed by checking an (MD5-) hash of the user's password. This might only be a strong enough mechanism for wired Ethernet networks where it is not easily possible to sniff data traffic. Moreover, EAP-MD5 does not provide a mechanism for dynamically deriving encryption keys (key management) which is crucial for secure wireless networking. Therefore, EAP-MD5 is declared to be a deprecated method.

3.5.2 EAP-TLS

Most of the early EAP methods (i.e. EAP-MD5) lack some important features like mutual authentication, key management and encryption. This stems from the fact that EAP was originally intended for wired Ethernet.

TLS (Transport Layer Security) and its predecessor SSL (Secure Socket Layer) are end-to-end cryptographic protocols that run directly on top of the transport layer, i.e. TCP. However, transport layer protocols are seldom present at the point of authentication. Since EAP runs directly on link layer protocols, encapsulating TLS packets into EAP messages solves problem of the the missing transport layer, and EAP-TLS was born.

As mentioned before, TLS is an end-to-end protocol. This means that communication can only happen directly between two devices without any intermediary station (same as PPP). This would

allow TLS to be used only in the two-party authentication model, having only a peer and a combined authenticator with server capabilities. Since this scenario does not fulfill today's requirement of having a separate backend AAA server, EAP-TLS needs to support the three-party model with an intermediary authenticator. Again, EAP solves this issue by encapsulating TLS into EAP messages. As we can see in Figure 3-6, EAP-TLS can support a NAS (or authenticator) between the peer and the authentication server. Here, the EAP server also acts as a TLS server, and a trusted and secured connection between the authenticator and the server is assumed.

Figure 3-6 EAP-TLS in the three-party model

Note that even if in the diagram above there are two separate red arrows indicating the secure TLS connection, the tunnel is actually end-to-end between the peer and the TLS sever. EAP encapsulates the TLS packets, which are then translated into the corresponding link technologies (Layer 2 or AAA).

EAP-TLS authentication is based on digital certificates. For mutual authentication, the server and the peer need to present a valid certificate in order to authenticate each other. This makes EAP-TLS one of the most secure authentication mechanisms, especially when the client side certificate is stored in a Smart Card, additionally protected with a PIN. While the use of digital certificates on the client (the peer) is considered to be the most secure authentication mechanism, it also is one of EAP-TLS's drawbacks. As soon as certificates are used for authentication, there must also be an appropriate PKI implemented to prove the ownerships of the private keys. This involves having Certificate Authorities (CAs) and Certificate Revocation Lists (CRLs). While all this is still manageable for server certificates, it is often too complex and too costly for client authentication. Last but not least, the user's privacy is getting more and more an issue. The user's identity should not be revealed (i.e. sent in clear text) during authentication. EAP-TLS does not support so-called pseudo-identities.

3.5.3 EAP-TTLS

Due to the drawbacks associated with EAP-TLS, an extension was soon introduced. It is called tunneled EAP-TLS or EAP-TTLS for short. It does not require the client to authenticate in the same way as the server. Instead, the peer waits for the secure TLS connection to be established, which is now based on the server's certificate only. Once the secure connection between the peer and the server is set up, the peer can authenticate itself using any legacy authentication method it likes. Since the communication is encrypted through the TLS tunnel, the peer can use very simple protocols like

CHAP (Challenge Handshake Authentication Protocol, [RFC1994]) or PAP (Password Authentication Protocol, [RFC1334]). PAP and CHAP are simple password only resp. challenge/response protocols coming from PPP that do not provide any security, identity protection or replay attack counter measurements.

The possibility to allow the client not to authenticate itself at all can also be useful, and would not be possible with EAP-TLS. Consider a user at a train station wanting to get access to some information server to look up the train schedules. Here, clearly server authentication is wanted, while there is no need to authenticate the client. Another example where client authentication is left out is e-shopping in the web. An e-shop typically provides a certificate that allows setting up a secure TLS/SSL connection between the client and the shopping server. However, the client is not authenticated, but can now trust the shopping server if the client believes that the certificate belongs to the same company running the e-shop. Based on this trust level, the user can provide its own credentials such as Credit Card number or billing address over the encrypted channel. In that way, EAP-TTLS can be considered to function the same way as HTTPS.

The EAP-TTLS specification introduces the notion of a TTLS server. The TTLS server is used to establish the secure TLS tunnel. While typically the TTLS server is integrated into the AAA server, it is possible to have the TTLS server located in a different domain. This is useful in roaming environments where the client connects to an AAA server of a different provider (i.e. the roaming partner of the customer's provider). The AAA server of the roaming partner cannot authenticate the client. It must contact the user's home AAA server (often called HAAA) to check the credentials. Having the TTLS server located with the roaming partner allows establishing the secure tunnel directly to the HAAA server without having a trust relationship with the visiting AAA server (VAAA).

3.5.4 LEAP

LEAP stands for Lightweight EAP and is a proprietary EAP method developed by Cisco Systems [16]. Since authentication is based on a modified version of the MS-CHAPv1 [RFC2433] and thus not using certificates and the implied PKI, they named it lightweight. MS-CHAP is a simple challenge/response handshake protocol from Microsoft [17]. Because the user credentials are not protected and open to dictionary attacks, in 2004, a tool called ASLEAP[18] appeared, that exploits the protocol based on the on-line dictionary attack. Since then, Cisco encourages the use of strong passwords, or to use the newer and better EAP-FAST (see next Section), also coming from Cisco Systems. However, because LEAP is relatively easy to install and maintain because no certificates are required, it became very popular.

3.5.5 EAP-FAST

After the exploit ASLEAP appeared, Cisco Systems developed a successor of LEAP, called Flexible Authentication via Secure Tunneling (EAP-FAST, [19]). Because Cisco wanted to stick to the lightweight idea of LEAP, i.e. not using certificates, the TLS connection is based on a shared secret, called Protected Access Credentials (PAC). While it is possible (depending on the implementation) to provide the PAC automatically, it opens a door for attackers to compromise the PAC. As described in Section 2.7 about key management, it is not possible to derive or transmit a shared secret out of nothing. The only way to do that would be to use certificates. Thus, the only secure, automatic method to distribute the PAC to the client is by using certificates. EAP-FAST does support server and/or client certificates, but this is often omitted (due to the administrative overhead and cost), thus manual distribution of the PACs is recommended.

3.5.6 PEAP

Protected EAP (PEAP, [1]) was jointly developed by Microsoft, Cisco Systems and RSA Security. As EAP-TTLS, PEAP is based on server side certificates to establish a secure TLS tunnel between the peer and the server. The main difference from a protocol standpoint between PEAP and EAP-TTLS is that EAP-TTLS allows any legacy authentication by the user (such as PAP and CHAP) while PEAP only allows methods that are defined for use with EAP. PEAP was proposed after EAP-TTLS was available. However, due to the size of the companies involved, PEAP quickly overtook EAP-TTLS.

There are several versions of PEAP in use, and many vendors have their own implementation. On one hand, PEAPv0 and PEAPv1 refer to the outer authentication method for the establishment of the secure TLS tunnel, and on the other hand EAP-MSCHAPv2, EAP-GTC and EAP-SIM refer to the inner authentication method used to authenticate the client through the secure connection.

PEAPv0/EAP-MSCHAPv2 is the most common implementation of PEAP. After EAP-TLS, it is the second most widely supported EAP method in the world. Unfortunately, Microsoft and Cisco have different specifications of this standard. Microsoft always uses EAP-MS-CHAPv2 as the inner authentication method, while Cisco additionally allows EAP-SIM to authenticate the client. Further, Microsoft supports PEAP-EAP-TLS and also markets it as PEAPv0. This implementation is very similar to EAP-TLS also requiring client side certificates. However, PEAP-EAP-TLS additionally encrypts parts of the client's certificate to further enhance security. PEAP-EAP-TLS is not very popular since only a few third-party clients do support it.

PEAPv1/EAP-GTC is an alternative from Cisco to PEAPv0/MS-CHAPv2. It supports the use of EAP-GTC (Generic Token Card) which uses a secure token to authenticate the user. The user is presented with

a challenge which is then answered with the secure token. Cisco further allows the use of EAP-SIM (see next Section) as inner authentication method in PEAPv1. However, Microsoft never added native OS support for PEAPv1 and thus it is rarely being used.

3.5.7 SIM Based Authentication

Coming from the cellular world (Global System for Mobile communications, GSM), authentication based on the Subscriber Identity Module (SIM) has become popular also for Wi-Fi Authentication. A SIM-Card is a special form of a smart card containing a small cryptographic micro-processor that is used to create a derived secret used for authentication and/or key management. Since SIM based authentication requires the physical presence of the SIM-Card (which is often additionally protected with a Personal Identification Number (PIN) Code) it can be considered a very strong authentication mechanism.

SIM based authentication uses a challenge/response mechanism, during which the SIM-Card is challenged with a random number called the RAND (Random Challenge) of bit-length 128, coming from the authentication server. The SIM-Card processor then creates the response to the challenge (Secret Response, or SRES) based on the shared secret key stored in the SIM, which is also known to the authentication server. The same calculation is done on the authentication server and the result is compared to the SRES received from the client. If the results match, the client is authenticated.

In addition to the challenge result (SRES), the SIM-Processor also creates a key used later for session encryption. The group (RAND, SRES, encryption key) is often referred to as the GSM-Triplet.

The algorithm used in the SIM-Card is often only known to the network-operator, which further increases the strength of the mechanism, compared to ‘only’ knowing the shared secret.

SIM based authentication is gaining attention since more and more cellular phones are equipped with GSM/UMTS and WLAN technologies, hence do have a SIM-Card interface integrated. This permits strong mutual authentication without user interaction, also for Wi-Fi networks. Compared to the common login procedure with a Captive Portal Page, where the user must type a username and a password into the browser for every connection attempt, this improves the user experience dramatically.

3.5.7.1 EAP-SIM

EAP-SIM [RFC4186] is the first SIM based authentication method for WLAN. It provides mutual authentication and session key agreement based on the Subscriber Identity Module (SIM-Card) of GSM networks.

EAP-SIM basically works the same way as described above in SIM based authentication. The client (i.e. a device with Wi-Fi capabilities and a SIM card interface) communicates with the authenticator which in a first step contacts the authentication server. The authentication server does not know the credentials from the SIM card. It has to forward the request to the GSM Authentication Centre (AuC) over the SS7 network. The AuC returns the GSM triplet, which is then sent back to the authentication server and compared to the result received from the peer.

3.5.7.2 EAP-AKA

EAP-AKA [RFC4187] is an EAP method for Authentication and Key Agreement based on USIM credentials. USIM is the SIM pendant of 3rd generation mobile networks – Universal Mobile Telecommunications Systems (UMTS) and CDMA2000. AKA stands for *Authentication and Key Agreement* and is based on symmetric keys. The AKA algorithm typically runs on the USIM or a CDMA2000 (removable) User Identity Module ((R)UIM).

EAP-AKA includes a GSM compatible authentication mode (not EAP-SIM, basic GSM-Auth), but the client can refuse GSM-only authentication. The authentication server has to know whether the client uses a SIM card or a newer USIM card.

3.5.7.3 EAP-SIM vs. EAP-AKA

Since EAP-SIM and EAP-AKA are both SIM based authentication methods and both provide key agreement, what are the differences and which one is more secure and/or better suited for a given environment?

As described in the RFC4186 document, the "*EAP-AKA protocol has been partly developed in parallel with EAP-SIM, and hence [the] specification incorporates many ideas from EAP-SIM, and many contributions from the reviewers of EAP-SIM.*" [RFC4186]

In fact, both RFCs (EAP-SIM: RFC 4186 and EAP-AKA: RFC 4186) are tightly coupled and often referenced by each other. Both use a challenge/response mechanism built into the EAP framework. The algorithms solving the challenge both run on the SIM-Card (resp. Universal SIM-Card, USIM) providing optional identity privacy. This means that the identity stored in the (U)SIM, typically the

International Mobile Subscriber Identity (IMSI), is never transmitted over the communication channel. Instead a temporary identity is derived from the IMSI, called the Temporary IMSI (TIMSI).

Both protocols support the option for fast re-authentication. EAP-AKA and EAP-SIM also share most of the attributes and message subtypes. EAP-SIM and EAP-AKA protocol numbers should be administered in the same IANA [15] registry too.

The reason for the two protocols to coexist is simply that SIM based authentication and key agreement is required in two different networks, namely 2G (GSM) and 3G (UMTS) networks. The question asked at the beginning of this Section and also the Section's title containing the term "versus" is misleading, since one cannot choose which one to use. This decision is dictated by the underlying network architecture.

3.5.8 EAP-TPM

EAP-TPM ([20] and [21]) is a special EAP methods that uses the *Trusted Platform Module*. The TPM itself was introduced in 2002 by the Trusted Computing Group (TCG) [22]. The TPM is attached directly to the motherboard, provides cryptographic functions and safely stores a so-called endorsement key pair, similar to a private/public key pair. Furthermore, each TPM contains an identity number which is unique worldwide. From this point of view, the TPM is very similar to a SIM card. However, the TPM provides further functionality, such as the ability to automatically generate digital certificates that can be used for network authentication.

However, the EAP method which uses the capabilities of the TPM is not yet standardized. The specification is available as an Internet Draft [20], and a first prototype is being built within a current Swisscom project [23].

Without going into more technical details of EAP-TPM, one can summarize that this method overcomes the drawbacks of traditional client side certificates. This includes automated generation and distribution of the certificates as well as a secure and trusted way of storing the private key, based on dedicated hardware components.

EAP-TPM is a promising method, since the TPM is already integrated in most modern computer systems, thus does not require additional hardware. Since it uses a variation of a digital certificate it can be considered to be a stronger mechanism than password based methods. Furthermore, the whole authentication process can be automated completely, thus does not involve the user in the process.

We have now introduced currently available EAP methods as well as one exotic protocol (EAP-TPM). Each of these has its advantages and disadvantages, as one might expect. There is no best method that suits all scenarios. Therefore, we will evaluate these methods for their suitability in public wireless LANs in Section 6.1.

3.6 RADIUS

RADIUS and Diameter are two protocols that are intended for communication between the authenticator (NAS) and the AAA server. RADIUS can be considered the de-facto standard for AAA. However, almost since the beginning of RADIUS over a decade ago, RADIUS has some serious issues especially with respect to security (see Section 3.6.4 for details). DIAMETER tries to overcome the drawbacks of RADIUS and can be seen as its successor. We look first at the basic RADIUS protocol and its extensions, and then list some issues with RADIUS. Later we introduce DIAMETER and show how it solves some of the problems of RADIUS.

3.6.1 RADIUS Basic Protocol

RADIUS stands for *Remote Access Dial-In User Service* and was originally intended to allow a NAS to forward a dial-up user's request for network access to a backend authentication server [8 p. 127ff]. At the time of the first specification, PAP [RFC1994] and CHAP [RFC1334] were used to transmit the user's credentials and authenticate the users in dial-up settings. This is why the basic message structure in RADIUS is based on an Access-Request, Access-Response type pattern. From the beginning, RADIUS was designed to be extensible. This extensibility was exploited many times to accommodate new requirements such as support for EAP message transport and IPSec security.

RAIDUS is a client-server protocol. It is important to understand that the client is NOT the user or device that tries to connect (i.e. the supplicant or peer); it is the authenticator (NAS) that acts as the client in the RADIUS protocol. During authentication, the NAS forward the user's messages by translating EAP messages from layer 2 protocols (typically EAPoL) into the RADIUS protocol. After successful authentication, the NAS establishes a secure communication channel between the peer and itself. From then on, if accounting is needed, it accumulates service usage data and sends them to the AAA server. However, authorization and accounting were not included in the first specification of the protocol. This only supported PAP and CHAP procedures for authorization and thus was revised several times. Most of the functionality available today, such as authorization and accounting mechanisms, is described in RFCs that extend the basic RADIUS specification [RFC2865]. Examples of

such RADIUS extensions are RADIUS Accounting [RFC2866] and RADIUS Attributes for Tunnel Protocol Support [RFC2868].

Now let us see how the basic RADIUS protocol works and how it can be extended with additional functionality.

3.6.2 RADIUS Messaging

The basic RADIUS specification [RFC2865] defines four message types for communication between the server and the client. The message set was later extended to eight message types (RADIUS Accounting, [RFC2866]):

- **Access Request**, used to forward client requests. Generated by the NAS and sent towards the server.
- **Access Challenge**, used by the server to ask the NAS or the peer something or perform some action.
- **Access Accept**, sent by the server to confirm requests.
- **Access Reject**, the negative confirmation of a request.
- **Accounting request and –response**, two messages used to carry accounting information.
- **Status Server and Status Client**, two experimental message types not in use.

Newer RADIUS specifications additionally define other message types. In order to preserve backward compatibility with the existing deployment base they are declared as informational specifications rather than standard ones. Furthermore, it is not necessary to introduce new message types in order to extend the protocol. The mechanism to allow the addition of functionality is included in the message format itself.

A RADIUS message consists of a header, including a message type code, an ID, the message's length and an authenticator field used to (weakly) protect some of the message content. The actual data is transmitted in the payload, which is structured as a list of Time-Length-Value (TLV) attributes (see Figure 3-7).

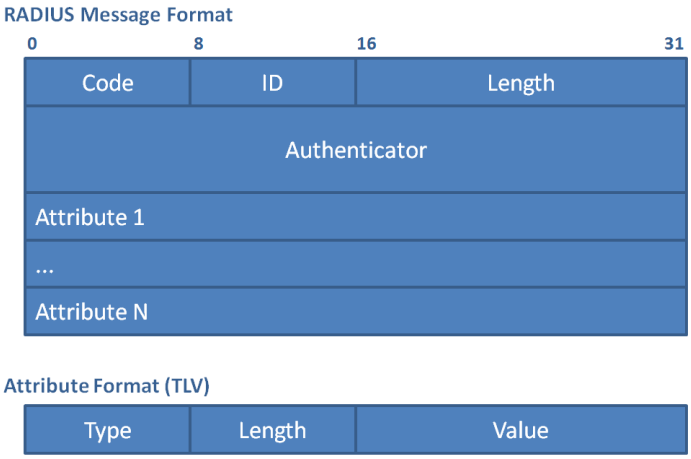

Figure 3-7 RADIUS Message- and Attribute Format [RFC2865]

The attributes are formatted as follows: each attribute consists of a type field (up to one octet), the attribute's length and the attribute's value that carries the data (hence the term TLV). The list itself can be of any size. However, since the attribute's type field is limited to 8 bits, only 255 different types can be defined. The basic RFC2865 defines about forty message types and lots of Vendor-Specific Attributes (VSA) are known. This message format gives RADIUS its flexibility and extensibility, which made it last long and is still heavily used nowadays. Nevertheless, as we see later, RADIUS has its limitations, and a successor is on its way.

3.6.3 RADIUS Transport Reliability and Security

At the time of the development of the first RADIUS specification, UDP was considered to be more appropriate than TCP due to the overhead of TCP's session establishment. Unfortunately, the lack of reliability of UDP caused some serious problems to RADIUS [8 p. 140]. Reliable communication is crucial for accounting since lost accounting information results in lost revenue for the operator or in customer disputes that cannot be solved.

RADIUS provides a minimal set of security mechanisms. Depending on the message type (Access Request, Access Response, etc.) some parts of the message are authenticated based on a shared secret between the authenticator and the server. The MD5 algorithm is used to create hashes of important elements[8 p. 132]. The hash is then transmitted in the authenticator field of a RADIUS message. A mechanism called Attribute Hiding is used to protect private information such as passwords (password hiding) from eavesdropping. However, the provided security mechanisms fall far short of today's requirements for secure communication. A common solution that overcomes the security vulnerabilities of the basic RADIUS protocol is to use IPSec [RFC4301]. IPSec provides

authentication and encryption for RADIUS messaging and thus is recommended to be used in practice (RADIUS with IPv6 and thus IPSec is discussed in [RFC3162]). However, IPSec does only provide node-to-node security. As soon as a proxy or a gateway is involved in the communication, the messages must be decrypted and re-encrypted completely at each node, thereby revealing private data. This is not only an unacceptable overhead, this also means that the operator must trust each single node on the communication path, which can be too much of a burden, especially in roaming environments.

3.6.4 RADIUS Issues

Nowadays the importance of robust and reliable AAA servers and network architectures is increasing. Quality of Service (QoS), mobility, roaming considerations and many other criteria are becoming primary design considerations of modern network architectures. Despite RADIUS's extensibility it cannot provide the required flexibility. Not only issues such as proxy chaining as described above, lacking security and privacy mechanisms and transport reliability are limiting, also the interaction of a RADIUS server (or AAA server) with other entities in the operator's network (intra-domain) demand more flexibility.

Diameter is considered to be RADIUS's successor and overcomes most of the shortcomings. However, RADIUS is still the de-facto standard for AAA communication and is widely deployed.

3.7 DIAMETER

Now that we have seen the limitations of the ageing RADIUS AAA protocol we look at its successor, the Diameter protocol. Diameter was selected as RADIUS's successor by an IETF working group called the AAA working group [24]. In early 2000 they defined a comprehensive requirement document [RFC2989] for future AAA protocols and compared several proposals. To cut a long story short [8 p. 147f], Diameter was considered well suited for future requirements of an AAA protocol.

Diameter is very versatile and addresses almost every aspect an AAA protocol should cover. On the other hand, this makes it rather cumbersome and difficult to understand. In the context of this thesis, we only consider a few key aspects of the protocol and compare it to RADIUS.

Diameter's specification is spread over several RFCs. The most important one is the Diameter base specification [RFC3588]. While it covers most of the basic building elements, such as a basic set of messages, attributes and a complete definition of roaming operations, it does not mention how to deal with NASs and does not define any authentication mechanism. Instead, the new concept of

Diameter applications is introduced. Diameter applications are services, protocols and procedures that use the facilities of a Diameter server, proxy or the base Diameter protocol itself, and are defined in separate RFCs. Surprisingly, the handling with NASs and support for authentication mechanisms like EAP, which is essential for AAA communication, is defined as a separate Diameter application (Diameter EAP Application [RFC4072] and Diameter NAS Application [RFC4005]).

One of the most important issues with RADIUS was its use of the unreliable UDP transport protocol. Diameter requires all communication between Diameter nodes to run on TCP or the Stream Control Transfer Protocol (SCTP). Together with additional requirements, such as fail-over handling and the mandatory use of IPSec or TLS over each connection, it provides a solid base for a reliable protocol.

Another advantage of Diameter over RADIUS is the explicit specification and definition of agents and proxies. In RADIUS, although the concept of proxy chaining via intermediate server is mentioned in the RFC, data object security and inter-domain roaming is not clear. Thus, there are many vendor-specific implementations that introduce interoperability problems. Diameter defines explicitly how intermediary nodes should behave, which makes it suitable also for roaming environments.

Of course there are many other aspects of Diameter that are important. However, we hope that the information presented here suffices to understand the main differences between RADIUS and Diameter. With respect to authentication, the most important feature they both provide is EAP message encapsulation, i.e. EAP over RADIUS and EAP over Diameter.

Chapter 4 PWLAN

Now that we have seen the basic concepts, protocols and standards of wireless LAN security and authentication, we concentrate on public hotspot environments. We identify the different requirements for home-, enterprise and public WLANs. The current PWLAN authentication solution at Swisscom is presented and serves as the reference architecture.

We look at the current approach to authentication in public hotspot environments. The most widespread mechanism today is the Universal Access Method (UAM) in combination with Captive Portal Pages (also known as WISPr). While this solution provides great flexibility for the different business models, it does not sufficiently protect the customer's privacy and is not the most usable solution.

4.1 Public Hotspots

When companies and private households started to use wireless LANs, people did not realize the dangers of opening their networks with WLAN. Although link encryption with WEP was possible, most devices were configured by default not to use any encryption at all, primarily to ease the installation of the devices. As a consequence, there were soon very many open networks available that anyone walking by could get in. Fortunately, companies realized that strong encryption and access control is necessary, and started to secure their WLANs. Nowadays, one seldom finds an open access point when walking through the streets. On the other hand, more and more devices are being equipped with Wi-Fi capabilities. Not only laptop computers, but cell phones, PDAs, Smartphones, gaming consoles and soon cameras and other devices will be searching for WLAN access points. Mobile Internet and many new mobile applications and services are becoming popular, all requiring almost instant Internet connection. While cell phones can use data services offered by their provider, such as 3G or UMTS connectivity, WLAN-only devices are usually only able to access private networks at home or at work.

On the other hand, ISPs are seeing this trend towards mobile computing and are offering WLAN access to the public. Such WLANs are known as public WLAN or PWLAN, and the actual access points are called public hotspots. Nowadays, you can find public hotspots in train stations, airports, hotels, cafés, restaurants and other public areas where people meet. Swisscom started in 2002 with their own PWLAN platform, now having over 1200 hotspots currently installed in Switzerland.

PWLAN is available not only for Swisscom customers, however. Typically, a hotspot operator offers the services to anyone willing to pay for it. Billing is either done through the monthly subscription for residential customers, while others have the option to use their credit card account or buy a pre-paid voucher card in retail stores.

Of course Swisscom is not the only player in the game. Providers such as T-Systems, AT&T, TheNet, Monzoon, TPN and others are offering PWLAN services worldwide. In order to cover an area as large as possible, the providers have roaming agreements that let customers of their roaming partners log into their networks.

Another approach to offer PWLAN services is taken by so-called network aggregators. Companies like Boingo, TheCloud and others use the existing hotspots of different providers worldwide and offer a monthly subscription to use their relatively large base of aggregated hotspots.

Last but not least, there is a growing number of user communities which are building their networks based on their members sharing their private network access with each other. By allowing other community members to use the Internet connection of their private access point, the members are allowed to access all the access points of the other community members, typically for free. The biggest open network today is FON. FON is a Spanish company with members in over 140 countries, increasing weekly by 20'000 new so-called FONeros (members of the FON community) [25].

While FON and other open access network providers are successfully enlarging their communities, they do all face a fundamental problem in the near future. Especially in many European countries, users are, by contract with their ISP, not allowed to share their connection with others. This has several reasons - the most significant one being authentication. The contract holder is responsible for anything that happens through his or her connection. Be it illegally downloading music or consuming child pornography, for any crime being committed through a customer's Internet access, the contract holder will be prosecuted. There are solutions to this problem, and legal regulations are discussed intensely, also by the European Union. Thus the future to open access networks such as FON is unclear. Due to this fact and the different approach such communities take, we do not consider these networks further within this thesis.

4.2 PWLAN vs. Home- and Enterprise Networks

In Section 3.2 we presented the two modes of the WPA standard, namely WPA-PSK and WPA enterprise. Further, we have identified the different requirements for home- and small business

networks on the one hand and a corporate network on the other. Enterprise networks require much more flexibility to manage their user base, hence they use centrally managed databases known as authentication servers. Furthermore, enterprises have much more stringent security requirements than private users. EAP as the standard authentication framework in WPA enterprise allows a variety of different network security policies. Ranging from simple (username, password) credentials, up to dedicated hardware tokens, everything is possible. However, not every method is optimal in every situation. Better and stronger security comes only with higher cost. Thus, network security is always a trade-off between the level of security on one hand, and cost, and usability on the other. The question is now, what is the best solution for public hotspot providers?

Before we try to find an answer to this question, we have to identify the differences between a PWLAN and an enterprise network. First and foremost, the main difference is that while an enterprise solution can manage 1000 users and more, a public hotspot provider must support several 10'000 of possible customers. An authentication server as used in large companies can, if configured correctly, manage several thousand users. However, the size of the user database actually does not matter at all, the question is how frequently the users connect, or how many logins per second can be handled.

Scalable servers and network hardware are not the only issues. The difference lies also in the diversity of the users. In companies, there are administrators in charge of configuring the client devices. Every single computer that is allowed to access the network must conform to the security policies defined by the company. This typically does not only involve the requirement to have accurate protection against viruses and malicious software, but also the correct setup of the Wi-Fi access. In an enterprise scenario, it is possible to install dedicated software clients that handle network connections according to predefined profiles. It is possible to install client certificates, and configure the PKI systems accordingly. In other words, the administrators are in full control of their client devices. This is clearly not the case in PWLAN environments.

In public hotspot networks, the operators have no control over their client devices. At the same time, they must support as many different devices as possible, without being able to directly configure them. This might seem unproblematic, since there are standards and common protocols available. However, as we shall see later, an incorrectly configured device can be open to attacks, even if the strongest mechanisms are being used.

Corporate networks must protect sensitive information. If business relevant documents leave the company's network, this can be very damaging. Hotspot providers on the contrary, do not really have

to protect their network against intruders. Nevertheless, they must prevent unauthorized access, since granting access to a user that is not paying means direct loss of revenue.

We have now identified the different assets that need to be protected in the different environments. Companies must protect the information accessible through their network, and hotspot providers must prevent unauthorized access in order to make money. But there is something more that should be protected, and we did not consider until now – the customer. Especially in public areas, users accessing public hotspots are just not aware of the dangers coming from wireless communication. Anyone sitting next to them, or with dedicated hardware even from several 100 meters away, can listen to everything transmitted through the air. If no counter measures are taken, it is possible to read emails that someone downloads or listen to chat conversations. Even more dangerous of course, credit card numbers, login credentials and other passwords can easily become compromised, without the user realizing the theft. Another issue is the user's privacy. Based on traffic analysis of a specific user, a detailed tracking path can be build. This problem is new due to public hotspots, since now users are not only accessing WLANs at home and at work.

Fortunately there are mechanisms available that protect the users accordingly. However, most hotspot providers do not implement them. While there are reasons for omitting them it is now time to put more focus on the protection of the customer. We discuss this topic of PWLAN security in Chapter 5.

4.3 Hotspot Login Mechanisms

Here, we present the common approach to PWLAN authentication at a public hotspot. We consider the solution currently used by Swisscom which serves as reference architecture. Other PWLAN providers offer very similar services and products.

4.3.1 Captive Portal Page

The most common method for authentication and user authorization at a public hotspot is the so-called Captive Portal Page (CPP) solution. A captive portal is a web page to which every unauthenticated user is redirected when trying to access the Internet through a browser. The CPP provides a form for entering the credentials, giving the option to use the credit card account and gives links to the roaming partner's login pages. The following picture shows the CPP of Swisscom:

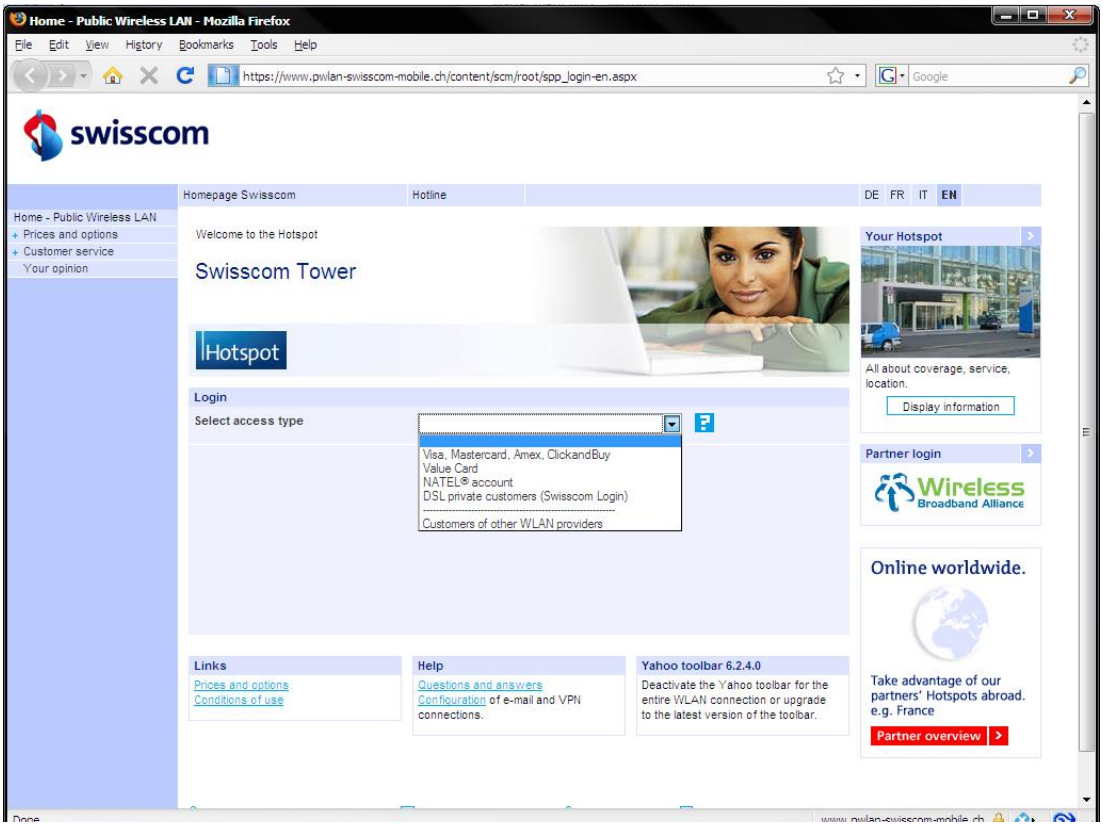

Figure 4-1 Captive Portal Page of Swisscom's PWLAN

The captive portal page can only be displayed if the customer first connects to the access point of the provider, at Swisscom having the Service Set Identifier (SSID) "MOBILE". This connection is unprotected (no WEP or WPA encryption enabled) and hence does not need any passphrase or username. However, leaving this link unprotected can be dangerous, as we shall see later. The captive portal page itself is authenticated using a standard HTTPS connection which is based on a server certificate. Using this HTTPS connection, the login data sent by the user is encrypted and thus protected against eavesdropping. One could argue that this is a safe login mechanism, but we will later see that only protecting the user's credentials is not enough. There are still specific attacks possible before the HTTPS connection is set up, and, much more important, there is no link encryption after the login process completed. Hotspot security is discussed in Chapter 5. Let us now look at the different options that a user has for authentication.

On the captive portal page the user can enter the credentials, which can be of several different types:

4.3.1.1 Pre-paid users (Vouchers)

Pre-paid users are customers that have no existing subscription with the provider (here Swisscom). As the name suggests, pre-paid users pay for the service before they can use it. The sessions are

typically based on the usage time of the service. For example, you can buy a value card which lets you access the Internet for one hour. Such value cards (or vouchers) can be physical, or virtual:

- **Physical Value Cards**

  Physical Value Cards can be bought at Swisscom retailers. Typical retailers are hotel reception desks, train station ticket shops, kiosks and Swisscom shops. Alternatively,Value Cards can be ordered online from Swisscom. The voucher has the credentials covered with a scratch field. These credentials can only be used once to log in to a hotspot. Once logged in, the customer can access services based on time or data usage (depending on the voucher) until the credit is depleted.

- **Virtual Vouchers (Credit Card)**

  A more convenient method to buy a voucher is to use a Credit Card Account. When a user selects the option to buy a virtual voucher, he will be redirected to a form where he can enter his Credit Card information. Once the data has been verified, the user is allowed to access the Internet. The user additionally receives a 10-digit code and a password that can be used to start another session later.

4.3.1.2 Post-paid users

Swisscom customers with an existing Swisscom subscription can use their subscription to use the PWLAN services in a post-paid manner. The customer will be billed for PWLAN usage directly through his or her monthly subscription bill. At the moment, there are two types of subscriptions that can be used for PWLAN login:

- **Mobile Phone Subscription (NATEL®)**

  By entering the mobile phone number into the login form, the user receives a one-time password (OTP) on his cell phone via SMS (Short Message Service) message. The OTP together with the mobile subscription number are used as credentials and to authenticate the user. The customer will be billed directly through his monthly mobile phone subscription according to service usage.

- **DSL Private Subscription**

  Private Swisscom customers that have a broadband subscription have received a username and a password when they signed the contract (Swisscom Login). These credentials can be used to log in at PWLAN hotspots of Swisscom.

- **Roaming**

  Typically, customers of other PWLAN operators can select their operator and get redirected to the CPP of their operator. There, roaming customers can log on as usual.

4.3.1.3 Session termination

An important issue with respect to accounting is the detection of the user's session termination. If the end of a user session would not be recognized by the system, it would not be possible to bill the customer accordingly. The mechanisms to detect session termination are either providing the user with an explicit log-off button in a separate browser window (a browser window that will pop-up after the authentication process), or requiring the client to periodically re-authenticate. The latter method is performed automatically without user interaction and is used to detect if a user terminated the session without properly logging off. This can happen for example when the user walks out of the range of the access point or shuts down his device.

4.3.2 WISPr Roaming

In the previous Sections we have described the captive portal page solution, which is the most common way for authenticating customers at public hotspots. The CPP solution is used by almost every provider of public WLAN, since it allows the provider to offer various types of log-in mechanisms, including the possibility to authenticate customers of a roaming partner.

However, there is no standard to define how roaming should be implemented. Also the captive portal page solution is not standardized. If two hotspot operators want to make roaming possible, they would need to specifically develop a solution together. This would make it almost impossible to have roaming agreements with several operators at the same time, and makes it very costly to add new partners to the list. Fortunately, the best common practices are put together in a document called the WISPr recommendation[26].

WISPr stands for *Wireless Internet Service Provider Roaming*. It is a document, specifying the best common practices for roaming of WISPs. However, it is not a standard. The document was published in 2003 by the Wi-Fi Alliance [14] and was very successful in harmonizing the market of public WLANs. Most operators of public WLANs are still following the recommendations, enabling simplified roaming between providers. Unfortunately, due to patent rights, the document was withdrawn by the Wi-Fi Alliance and is no longer available on the Wi-Fi homepage.

4.3.2.1 WISPr Recommendations

In order to give recommendations and suggestions on how roaming mechanisms should be implemented, a common vocabulary and a clear definition of terms used is essential. WISPr identifies all involved participants, such as the hotspot operator, the home entity (the provider that owns the account relationship with the customer) and the hotspot property owner. This gives the operators a common vocabulary that eases communication.

The WISPr recommendation document further suggests using the Universal Access Method (UAM) to authenticate users. This is exactly what we presented earlier under the name Captive Portal Page (CPP) solution. It is a browser based access method that allows the users to enter their credentials on a web page. WISPr defines how such a CPP should look, and what functionality must be implemented.

Along with a set or required functionality and the look and feel of the CPP, WISPr further gives suggestions on how the operator's network architecture should be built. It introduces a so-called Public Access Control Gateway (PAC), a logical entity that manages IP addresses, CPP redirection, AAA and others. With respect to AAA, WISPr defines RADIUS as the AAA protocol to be used for Wi-Fi roaming. Since the basic RADIUS protocol is not primarily intended for wireless roaming, it has to be further enhanced by message attributes and specific parameters. WISPr recommends a set of RADIUS attributes that each operator must support in its implementation.

With respect to security, WISPr clearly states that mutual authentication should be performed (the user and the network must be authenticated). In order to authenticate the network, the CPP must be secured using HTTPS, thus providing an SSL certificate and a PKI. This will protect the credentials entered on the CPP and ensures that the CPP of the PWLAN operator is legitimate. However, it is important to understand that this encryption is only protecting the login process, i.e. the entered credentials. There is still the possibility for attacks prior to the login process and there are no means of encryption between the client device and the access point after the log-in succeeded.

The only link encryption standard that was supported by most devices at the time of writing the WISPr recommendation was WEP. Since the WEP protocol for Layer 2 encryption has already been proven to be insecure, WISPr concurs with other recommendations by stating not to use the WEP algorithms. On the other hand, the 802.1X port based access method was already published. However, most client devices did not yet support the new access method that would enable WPA encryption. Nevertheless, the WISPr document already states what considerations should be made when deploying 802.1X.

Since WEP encryption is not allowed in WISPr, and WPA/802.1X was not yet widely enough available, the user's traffic after authentication is always sent as clear text. WISPr strongly suggests the following:

> *"… it is highly recommended that Home Entities promote the use of Virtual Private Network software by their users to protect the privacy of all sensitive over-the-air data and Internet transactions."* [26]

This solution is actually protecting the user's data traffic sufficiently. However, such software must explicitly be downloaded and configured by the user. Further, the Wi-Fi operator should provide a VPN server that the customers can connect to. Often, no such server is available, and the users have to know either a free VPN server, use the one of their company if any or pay for a commercial VPN service. Anyway, just promoting the use of VPN software places the responsibility for security with the customer. In our opinion, this is not appropriate, since most private users of public WLANs do not know anything about network security.

4.3.2.2 WISPr XML

While the UAM method using web pages for login is a good solution to authenticate new users that have no existing subscription with the operator, it is far less than optimal with respect to usability for regular users. The credentials have to be re-entered each time the user wants to start a new session. Furthermore, if non-browser applications require Internet connection, then the user explicitly has to open the web browser and log in on the CPP before the application can access the network.

The authors of the WISPr document have thought of that issue and proposed a sound solution using an XML-file based login mechanism. It is intended to be used by so-called Smart Clients, a simple software client that is provided, customized and configured by the WISP to automate the whole login procedure.

The solution promoted by WISPr is to use the same UAM HTTP interface as the CPP. This requires the Smart Client to know the exact URLs for login and logout requests. Remember that the goal of WISPr is to enable roaming with different operators, so hard coding URLs would require that all WISPs use the same login URLs. WISPr however decided to use a different approach. The basic idea is that the Smart Client sends a normal HTTP GET request to an arbitrary URL. The access gateway then responses with a HTTP redirect message, containing specific authentication parameters such as log-in and log-out URLs, a protocol version, message codes and so on. The Smart Client from then on knows

all information required for authentication. Let us look at the message flow of a successful, automated login procedure:

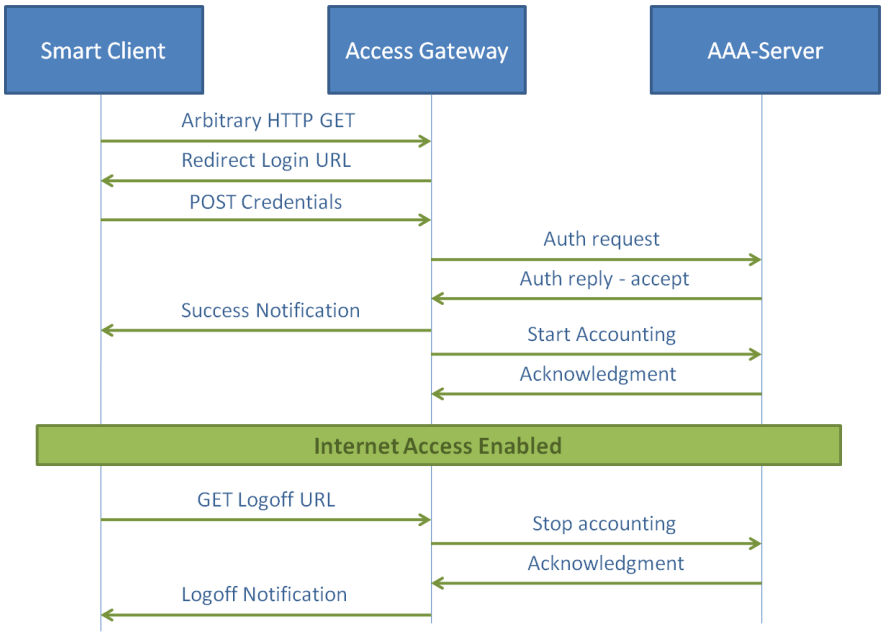

Figure 4-2 WISPr Smart Client - Message Sequence Chart of a successful login

There are several things that should be mentioned. First of all, it is very important that the HTTP POST message (the second message sent by the Smart Client containing the credentials) must be sent over secure SSL. Otherwise, the credentials would not be encrypted and could easily be captured. HTTPS connections require that the access gateway provides a digital certificate that the client can verify. This will not only protect the sensitive credentials being sent, it also guarantees that the gateway is genuine, and not a rogue access point.

Another issue is that XML was not yet properly implemented in the different browsers at the time of protocol specification. The access gateway would typically respond to the first HTTP GET request with a normally formatted HTTP page. In order to allow the XML based protocol to work, the access gateway has to embed the XML code inside the normal HTML page, typically in HTML comments, so that browsers do not become confused by the unknown code. Optionally, the first reply message of the access point containing the actual login URL can point to a URL that implements true XML.

4.3.2.3 WISPr today

According to Trustive's WLAN Roaming Research Results 2007[27], which are based on responses of over 100 hotspot operators worldwide, representing over 60'000 hotspots, over 80 percent of hotspot operators are fully WISPr compliant. The results further show that WISPr is widely known by

European operators, while the US- and Asia pacific markets are only slowly adapting. Nevertheless, WISPr can be seen to be the standard for public hotspot access, without actually being an official standard.

Since the first version of the document published by the Wi-Fi Alliance was withdrawn, Intel and others have tried to make a similar proposal, called the *International Roaming Access Protocols* (IRAP [28]). IRAP is an open framework that defines a set of core protocols for global and unified network architectures within Wi-Fi operators. IRAP soon has been rolled to the *European Telecommunications Standards Institute* (ETSI, [29]) for standardization and is now integrated in the TISPAN Next Generation Networks Standards Body. The *Telecoms & Internet converged Services & Protocols for Advanced Networks* (TISPAN, [30]) is a body of ETSI, specializing in fixed networks and Internet convergence. TISPAN thus is not only concerned with Wi-Fi roaming and provides standard architectures for all kind of fixed and mobile networks. Nevertheless, WISPr is still the solution implemented by Wi-Fi providers and referred to as “the standard”.

Chapter 5 PWLAN Security

If you browse the Internet for the term “hotspot security” you will find very many articles that warn users of the dangers of using public hotspots. Such warnings are also published through other media channels, such as TV magazines and newspapers. As a consequence, the users often are frightened and will not use public hotspots in the future. The situation could be different, if PWLAN operators were to adapt to the 802.1X standard for authentication at their hotspots.

This chapter is dedicated to the security and privacy of public hotspot customers. We identify the vulnerabilities of current implementations (i.e. the CPP solution) and show how attackers can exploit them. We will argue why VPN (Virtual Private Network) is not the ultimate answer to these problems.

Despite the existence of secure mechanisms that do respect the user’s privacy, they are seldom available at public hotspots. Unfortunately, implementing the well known 802.11i (or WPA/WPA2 based on 802.1X and EAP) standard in PWLAN environments is not as trivial as one might think. Issues, such as deployability, the user experience, the diversity of devices and systems that need to be supported, administrative cost and maintenance issues are hindering hotspot providers to adopt the standard. Nevertheless, since people are getting more and more concerned about their privacy, we try to find individual solutions to these problems.

5.1 Party WLAN

We have seen in the previous chapters how hotspot users can authenticate at public hotspots. The most common solution that is used today by almost all hotspot operators is based on the WISPr recommendations. This involves having an open (unprotected) SSID broadcast, accepting all users without access control. The users are then redirected to an HTTPS secured CPP that lets the user enter the credentials. After the login process, all traffic towards the Internet is granted for that specific user.

We have also mentioned several times that this solution does not provide any means of protecting the user before and after the login process. First and foremost, after the user is authenticated, there is no link layer encryption that protects the data traffic between the client and the access point. As already discussed in Section 4.2, anyone sitting next to an authenticated hotspot user can listen to anything the user sends and receives, including private and sensitive information such as emails, instant messaging, passwords or even Credit Card information. Not having a secured channel in a

public environment can be seen equal to shouting out loud your name, your email passwords and Credit Card number at a party. Most of us would not do that. Unfortunately, using PWLAN without any further counter measures taken is such a Party WLAN.

Is it really that bad? While it is true that everything being transmitted without encryption can be heard by others, the question is what actually is transmitted. Most e-commerce web sites are additionally protected using secure SSL tunneling based on digital certificates. This ensures that all traffic to that specific web site is protected in an end-to-end manner. Hence, entering your Credit Card information in a secured eShop is actually safe, even without enabled channel encryption. On the other hand, email traffic is seldom protected additionally. Standard protocols such as POP3 (Post Office Protocol Version 3) and SMTP (Simple Mail Transfer Protocol) do not by default encrypt the application data. At the Black Hat 2007[31], a security conference also inviting hackers, Robert Graham showed in a live demo how he can get session IDs and authentication cookies out of the unencrypted traffic of an authorized hotspot user. He used the information to hack into the victims Gmail account (check[32] for details). The problem however, is not only that the attacker can read private emails. Often, people use the same (or a similar) password for several purposes and accounts. Furthermore, most otherwise secure websites with user accounts offer to send forgotten passwords via email. That way the attacker can access almost any online account of the victim. Last but not least, a private email account contains much information that can be used for social engineering [33].

People often think that it would take too much effort to actually hack a WLAN. This is not the case, because there is nothing to be hacked. Unencrypted traffic can easily be sniffed with free network monitoring tools such as Ethereal [34] or its successor WireShark [35] without the need of special hardware. Anyone without any special education or knowledge of network security can easily start an attack. One cannot even consider just listening to open traffic passively as an attack. It is actually not illegal to do that, only the saving and further use of the data is prohibited by law and is therefore a criminal offense.

5.2 The Evil Twin Attack

In the Section above we have argued that Credit Card information entered during eShopping typically is protected using HTTPS. The same is true for the credentials which are entered at the CPP for login in. Does this mean that we are also safe to use our Credit Card for authentication at public hotspots? Not so, because of the evil twin.

The evil twin is an attacker that sets up a rogue access point with the same SSID as your PWLAN operator. The attacker then provides you with a faked CPP on which the unsuspecting user will enter his Credit Card information, or subscription credentials. The evil twin just records the log in information and can then even let the victim connect to the Internet, so that he or she does not even recognize having been tricked by a MITM. The next thing will be a massive Credit Card bill or a massive subscription bill at the end of the month. Furthermore, since the victim is directly connected to the evil twin's computer, the attacker can further try to find security holes of the victim's system, e.g. browser exploits, which could allow him to place viruses, malware or directly access the victim's computer hard drive.

Why does this work? If the signal of the rogue access point is stronger than the original one, due to a stronger transmission power or just by locating it closer to the victim, the original signal will get suppressed. The only hint that one has to decide whether the access point is legitimate is the SSID being broadcast – which can easily be faked. If one sees a wireless network with SSID "MOBILE", "t-mobile" or "free WLAN", one will connect to it, and if one was successfully connected to a network, the WLAN client will probably connect to it automatically. The next step is to open a browser window that will display the CPP. How can one decide whether this site is actually coming from the hotspot operator? One could look closely at the digital certificate and understand the mechanism behind it. However, most people will recognize the look and feel of their operator's CPP and will easily be tricked.

Again, all the hacker needs to start this kind of attack is a notebook computer with a WLAN interface and dedicated software tools such as AirSnarf [36] to fake the access point. He then only has to redirect all traffic coming from the victim to a faked CPP, to let the victim enter his credentials or Credit Card information. Because it is so easy to perform this relatively simple attack, not only attackers with criminal intentions are doing it, also less skilled computer enthusiasts are trying it out, just out of curiosity. Until now, no hacker has yet been prosecuted for acting as the evil twin. This is because the victims may never realize that they were using an illegitimate hotspot, and once the evil twin shuts down his rogue access point, it is almost impossible to find him.

5.3 **Other attacks**

The above two attacks on wireless LANs are only two of many others. However, with respect to public wireless LANs, these are those that customers really should be afraid of, since they can easily

be performed without the victims noticing the attack. On the other hand, as we shall see below, these attacks can be prevented by adapting to the 802.1X standard for authentication.

One attack that looks very similar to the evil twin attack from the customer's point of view is called session hijacking. Here, the attacker waits until an authenticated user leaves the hotspot without explicitly logging off. The network will wait another minute or so (depending on the timeout value) for further traffic coming from the client. During this timeout period, the attacker can easily pretend to be the authenticated client by spoofing the MAC- and the IP address of the authenticated client. This information is sent unprotected within every data packet. The access controller has no other means of telling whether such spoofed traffic is coming from the previously authenticated customer, or from an attacker. The attacker can then use the customer's session to access PWLAN. This attack is not that dangerous, since the customer's credentials are not compromised as in the evil twin attack. Nevertheless, the customer will get charged for the traffic being generated by the attacker.

Another attack that is possible in wireless LANs is the Denial of Service attack (DoS). As the name suggests, the attacker tries to overload the system with a large number of generated messages in order to prevent other users from accessing the network. To be more specific, an unauthenticated user at a public hotspot who is not authenticated will still get an IP address assigned. This is necessary since without a valid IP address, the browser would not be able to access the CPP. An attacker can exploit this by requesting new IP addresses from the DHCP (Dynamic Host Control Protocol) server with faked MAC addresses. By faking the MAC address, the DHCP server cannot decide if the new requests belong to the same sender and thus cannot block further requests. The attacker continues to request addresses until the whole IP range is used up. The server is then no longer able to assign new IP addresses to legitimate customers. This specific DoS attack is called DHCP poisoning.

DoS attacks are not that dangerous in public wireless LAN networks since no sensitive data of a customer is in danger. However, the above described DoS attack would not be possible in 802.1X enabled networks. Client machines only get an IP address assigned after having authenticated the customer. On the other hand, it is never possible to completely protect against DoS attacks in wireless networks, because any radio source can be used to jam the communication channel by generating noise within the frequency band in use.

Of course there are lots of other attacks possible in wireless LANs. It is a never ending game between network engineers and the hacker community. However, most of the attacks are against the network, not the users. The attackers want to get into the network of a company in order to steal sensitive data that could be of interest to others. In public wireless LANs, the only entity to be

protected is the customer. With 802.1X authentication using strong EAP methods that provide mutual authentication, the situation would be a lot better from the viewpoint of the customer.

5.4 Why VPN is not the ultimate answer

Public WLAN operators are aware of the dangers that arise when not providing link layer encryption such as WPA or WPA2. Their answer is to use a Virtual Private Network (VPN) which provides confidentiality, data integrity and authenticity. VPN is a software solution that establishes a secure tunnel between two networks. A public wireless LAN user can install a VPN software client and then connect to a VPN server. This will encrypt all traffic between the client and the VPN server.

On the one hand, this is a good solution to overcome the missing channel encryption. Once the VPN connection is established, the user must not fear anymore that his traffic is being sniffed by an attacker. He can then securely send and receive mails, do online shopping and chat with friends. However, the VPN solution is not the ultimate answer, for several reasons.

First of all, just promoting the use of VPN software by customers – as WISPr recommends it – puts the responsibility of secure networking with the customer. Hotspot users must be aware that they are sitting ducks without VPN's protection. Then they have to know that there is something such as VPN that can be used to securely browse the Internet. However, just knowing it does not secure anything. The customer then has to find and download a VPN client, install and configure it correctly. This is not an easy task that just any using a computer user can handle. Furthermore, a VPN server that can be trusted must be known in order to connect to it. Nowadays, most large companies provide a corporate VPN server that can be used by employees in order to connect to the enterprise's network. What about the others? They can either choose a freely available VPN server, or pay for a commercial VPN service. Free VPN servers are not easy to find, and must be questioned whether they can be trusted, since any further traffic will go through this server. As you can see, all this is way too much for a Starbucks customer wanting to check his emails.

PWLAN operators often support their customers in the process by providing a VPN client that can be downloaded directly from the CPP and a VPN server that can be used for secure Internet browsing. Nevertheless, most people are already struggling with the authentication process itself and are so happy once they can access the Internet that they just forget to care about their security. In my opinion, it should be the operator's responsibility to provide secure networking, not the users'.

5.4.1 Link Layer still unprotected

There are other, more technical issues with the VPN solution. First of all, VPN can only go as far down in the OSI layering model as to the network layer. This leaves the link layer unprotected, allowing a variety of attacks to be performed. Whether VPN is used or not, each client gets an IP address assigned before authentication is performed. This allows the aforementioned DHCP poisoning attack, as well as so-called domaincasting [37] attacks which allow tunneling Internet traffic through the DNS service. There are other, more severe attacks possible that could shut down the network completely.

5.4.2 VPN Latency

Moreover, establishing a VPN connection takes time. Even if it "only" takes several seconds, this will become an issue when the user is mobile and wants to connect to another access point. As soon as the user moves out of the range of an access point through which his VPN connection is running, the connection will be aborted. He then has to first authenticate at the new hotspot, and then re-establish the VPN connection. This makes latency critical applications such as Voice-over IP (VoIP) infeasible. On the other hand, 802.1X authentication provides fast-handoff and fast re-authentication mechanisms which allow changing from one access point to another in less than a second.

Latency is not only an issue with VPN when establishing a new connection. Since VPN encryption, depending on the protocol in use, is relatively resource intensive. VPN clients are typically software based, hence require a good share of CPU and RAM for encrypting the traffic. This results in increased round trip times that are affecting time critical applications. Furthermore, if the VPN server is not located at the operator, additional hops are introduced depending on the geographic location of the server. Again, this may further increase latency.

5.4.3 VPN Scalability

An important issue with VPN is scalability. If all hotspot customers were to use the operator's VPN server, the server would quickly become a performance bottleneck and a single point of failure. Surprisingly, the VPN server that Swisscom offers to its customers is rarely used. On the one hand, typical hotspot users today still are managers that travel around the world using PWLAN to connect to their corporate network. They do that by directly connecting to their company's VPN server. On the other hand, an almost idling PWLAN VPN server at Swisscom shows that private users not having the possibility to use their corporate VPN server either do not know, or do not care about their security.

As already mentioned, VPN uses up resources, not only on the client side. The server has to open and maintain a session for each user separately. This dramatically restricts the number of concurrent sessions, hence limiting the number of customers that can use the VPN server to secure their network connection. If the server is down due to whatever reason, customers can no longer securely browse the Internet.

5.5 802.1X Authentication at Public Hotspots

According to Trustive's report on PWLAN usage in 2007 [27], 45% of hotspot customers were CEOs, Chairmen, Managers and Sales representatives. Swisscom states that over 90% of their PWLAN customers are business people. However, the ongoing trend towards more and more mobile devices with Wi-Fi capabilities will affect this usage pattern. Private users will increasingly look for available access points, and will also be willing to pay for PWLAN Internet access. Since they will not primarily connect to a corporate network using VPN, transparent, automated logins, security and privacy concerns will become important operator selection criteria.

At the beginning of this chapter, we have seen that current WISPr-style solutions are not providing the required security. No link encryption, vulnerability to the evil twin attack, uncomfortable login using a browser, all this clearly shows that the WISPr recommendation is no longer the appropriate solution. On the other hand, 802.1X port based authentication using the EAP framework and RADIUS messaging, solves all these problems. However, 802.1X is not primarily intended for public WLANs and thus the different requirements must be addressed.

At the time the first devices supporting the new WPA standard became available, several experts stated that the upcoming 802.11i (WPA2, RSN) standard with its different EAP flavors will be the perfect solution also for hotspot operators.

Microsoft states in the context of introducing the Windows Provisioning Service (WPS, will be discussed shortly), that:

> *"As lack of security is a growing concern, the wireless industry in general is in agreement that* ***public Wi-Fi hotspots need to be secured as well, using technologies such as 802.1x*** *and Wireless Provisioning Services."* [38]

George Ou, a former ZD Net journalist, IT consultant and network security specialist, said in early 2005:

> *"A recent story on "Evil twin" Wi-Fi networks that spoof legitimate hotspots or corporate networks* ***makes it clear that all public hotspots should immediately implement 802.1X and PEAP authentication****. Currently, with most Wi-Fi hotspots, there is no simple way to tell whether or not you are using a legitimate hotspot. If you don't think this is a big deal — since you're probably using VPN anyway –think again!"* [39]

Also the Wi-Fi Alliance writes in its early WISPr recommendation, that:

> *"The recently introduced IEEE 802.1x standard provides a protocol for authentication and port-based access control supporting enhanced access security, but has not been widely deployed in public access environments."* [26 p. 7]

These statements clearly show that 802.1X is also applicable to public hotspot environments, not only large enterprise networks. The problem in the beginning was that in order to support the new WPA2 standard, the hardware had to be upgraded. This includes the operator that had to update its access points, and the customers, required to have WPA2 compatible network interfaces in their devices. Naturally, it takes time until the standard is widely available. However, almost five years after the standard was ratified, every wireless network card must support the standards in order to get certified by the Wi-Fi Alliance. Have the hotspot operators also updated their networks? Unfortunately, the answer is no, with only a few exceptions:

T-Mobile, a mobile phone subsidiary of Deutsche Telekom AG, announced in October 2004 [40] that they upgraded their networks in the US to the new standard:

> *"The company […] said today that it has officially completed the upgrades to support 802.1X in the access points at its 4700 domestic locations, one year after announcing plans to do so."* [40]

However, in order to authenticate at T-Mobile's hotspots using the new authentication method, the customers need to download client software, called the *T-Mobile Hotspot Manager*. We will see later, why usually client software is needed, despite the fact that almost all operating systems have pre-installed a client that is able to connect to 802.1X access points according to the standard.

Another hotspot operator that took the challenge of upgrading his network is Swisscom. Swisscom jointly developed the EAP-SIM authentication method, which is used nowadays in combination with the *Unlimited Data Manager* client and dedicated UMTS/3G/PWLAN hardware. Unfortunately, EAP-SIM only works with devices that include SIM card readers. Such devices are gaining popularity, since more and more notebook computers and netbooks are equipped with UMTS/3G connectivity, thus

include such a SIM card reader. Nevertheless, most current devices do not support EAP-SIM. Non-SIM based 802.1X authentication on the other hand, is not yet possible in Swisscom PWLANs.

5.5.1 Reasons for not adapting to 802.1X

The two examples of a successful implementation of the 802.1X standard show that it is possible to deploy the enhanced security standard in public wireless LANs. Unfortunately, almost all other hotspots have not adapted yet. What are the reasons for sticking to the insecure Universal Access Method using Captive Portal Pages?

One simple reason is that PWLAN customers do not ask for the additional security, hence there is no need to upgrade. This has also to do with the fact that still almost all users of PWLAN are business customers, who are used to use VPN software. With 802.1X, they would still need VPN software – not primarily to secure their communication channel, but to connect to the corporate network. Entering Credit Card numbers into a CPP is a process which frequent business users get used to. Another option for corporate customers is to use services like iPass, which will let users log in using a third party connection manager (the *iPass Connect* client), which provides a one-click solution based on iPass' proprietary protocols and servers, connecting through PWLAN.

The problem is that people do not ask for the additional security, because they are not aware of the current situation. If one would demonstrate to a regular PWLAN guest what actually can happen, he or she would be so scared that he would decide not to use a public hotspot anymore, even if the networks had been upgraded in the meantime. What we are saying here is that security should be the responsibility of the operator. Why are so few private people using PWLAN? Often current price plans are too costly for sporadic usage. Prices, however, are beginning to drop, since there are operators already offering flat-rate based price plans for a few Dollars per month (e.g. Boingo [41]). Aside from the cost factor, there are other reasons why people seldom use PLWAN. Either they do not know that it exists, or because they do not want to use their Credit Card account to be charged for PWLAN usage. People are uncertain how much it will actually cost them, even if prices are clearly communicated. There is just too much mistrust against PWLAN, not only because of the bad news stories associated with public networks.

Another important reason why PWLAN operators have not yet adopted the 802.1X standard is the problem of correctly configuring the devices. Although modern Wi-Fi devices supporting 802.1X authentication have integrated 802.1X clients pre-installed in the operating system, they still require careful configuration. Due to the diversity of different operating systems and devices, there is no

common configuration that suits all. The consequence is that individual configuration procedures, profiles or client software has to be developed.

In the introduction we already mentioned that security is always a trade-off between the level of security on the one hand, and usability and cost on the other. The current Universal Access Method (UAM) is a solution that clearly goes for usability and lower cost, thus compromising the security level. However, we will see that an 802.1X implementation does not cost much more, and in terms of usability can be even better than the UAM method.

5.6 User Experiences

Let us now look at the different user experiences of customers authenticating at public hotspots. Supposed Alice is a Swisscom customer with an existing Natel® (the cellular phone of Swisscom) subscription. She wants to check her email, listen to a newly released music album on iTunes and finally buy and download it, everything using her Apple iPhone [42].

We have chosen the Apple iPhone platform since it is the best representing the growing market of mobile devices with Wi-Fi capabilities. It also stands for the new kind of the ultra mobile customer that will use PWLAN in the near future, using bandwidth demanding applications. The iPhone (from firmware 2.0 on) supports the full 802.11i standard, including WPA and WPA2, both in private and enterprise mode.

The experience would be similar using a laptop computer with a WLAN network interface, independent of the operating system.

5.6.1 Captive Portal Page User Experience

Let us first consider the current situation with a CPP. Alice opens the network GUI on her iPhone to check weather public WLAN is available. She will then see the access point with SSID “MOBILE” and tries to connect. The connection will be established without any further user intervention since no credentials have to be entered.

Alice now thinks that she is connected since the WLAN indicator of her iPhone indicates a working connection. However, Alice is not yet authenticated, thus does not have Internet access yet. Unaware of that, she will open the email client and try to check her mails. The attempt will time out without any hint why it is not working. A simple message, such as “cannot connect to the server” will be displayed. The same will happen, if she opens iTunes. However, since Alice is vaguely familiar with

network protocols, she wants to find out what the problem is, and tries to open the Google search. Once her browser (Safari) opens and tries to connect to the search page, the Captive Portal Page of Swisscom will appear. Now Alice realizes that she first has to log in. She enters her mobile subscription number and receives an SMS message shortly after. She enters the one-time-password that she received into the CPP which will grant her access to the Internet. She can check her mails and listen to the new music album. Alice will be billed directly through her monthly subscription bill.

The next time Alice walks into the range of a PWLAN access point, the iPhone will automatically connect to the SSID “MOBILE”. However, authentication will not be performed in an automated manner. This time, Alice does not want to use PWLAN because she just wants to check her emails using UMTS/3G connectivity. Unfortunately, the iPhone thinks that a Wi-Fi connection is available (since it is connected to the open access point) and disables UMTS/3G. Alice will not be able to check her emails, since she would first have to open Safari, request a new OTP and manually enter these credentials. The iPhone is stuck in a so-called “hotspot black hole”. Alice has to manually disable Wi-Fi connectivity temporarily to use 3G when she is in the range of a “known” hotspot.

5.6.2 Smart Client User Experience

There are several applications available that act as the UAM Smart Client as described in the WISPr recommendation (see Section 4.3.2.2). Such applications are available for a diversity of operating systems, including the iPhone OS. Example applications available in the Apple App-Store are AutoWi-Fi [43] and Devicescape Easy-Wi-Fi [44].

Such client software allows automatic authentication based on username and password credentials which you can get from your operator, if supported. For example, Swisscom customers can use their DSL subscription credentials to log into Swisscom PWLAN hotspots. Once the application has been downloaded and installed, the credentials have to be entered only once.

When Alice comes into the range of a PWLAN access point, she will again fall into the hotspot black hole, if she was once connected to the access point before. However, instead of having to open the browser and manually enter her credentials, she can now just open the third party application which then automatically authenticates her using the previously entered credentials using the WISPr XML protocol.

This solution however, still requires Alice to explicitly launch the third party application, since there is no possibility with the current iPhones to allow applications to be running in the background. If it were possible run such applications in the background, the application would be able to detect that a

known hotspot is in range and then automatically authenticate. Other operating systems, e.g. Microsoft's Windows XP allow such software, thus enabling true seamless authentication without any user intervention.

While this solution can be really comfortable for the user, there is still the issue that no link layer encryption is enabled. If Alice wants to securely browse the Internet and check her private emails, she still would have to configure and run a VPN client. This involves clicking, or "tabbing", another few unnecessary buttons. Furthermore, Alice has no clue if she actually is connected to a legitimate access point.

5.6.3 802.1X User Experience

802.1X promises to provide the same seamless experience as with Smart Clients, while also protecting the network and the user with mutual authentication and strong encryption mechanisms. Let us now consider the situation that Alice would experience if the operator offers 802.1X authentication. However, we now have to consider two separate cases. The first being Alice with her device already configured and the second when Alice tries to connect to a new hotspot for the first time.

Let us start with the former scenario, where Alice has a configured iPhone. When Alice walks into the range of a known hotspot (i.e. one of an operator for which she has configured her device), the iPhone will automatically connected to the hotspot within seconds. No third party application has to be launched, no captive portal page has to be opened, and no password has to be entered. This is seamless authentication, secure and without user intervention.

As simple and easy as this works for frequent customers with configured devices, as tedious it can get when new customers try to connect for the first time. This issue is the topic of the next Sections.

5.7 Provisioning 802.1X Clients

If Alice has never been connected to a PWLAN hotspot with her iPhone before, she first has to configure her device (this holds for all devices, not only the iPhone). A correct configuration of the device is a problem with 802.1X. Incorrectly configured devices will not be able to connect or be still vulnerable to the evil twin attack. This information originates from Joshua Wright, who presented these aspects of PEAP provisioning at the ShmooCon in 2008 [45].

5.7.1 Important Client Settings

There are several issues that must be considered when configuring the supplicants on the devices. While default configurations can work (in case of the iPhone the connection would successfully be established using the default settings), others, such as MS Windows XP, will not connect. However, all devices need some important settings in order to perform true mutual authentication.

The problem comes from the fact that 802.1X in combination with an EAP method that supports mutual authentication is typically based on server certificates that need to be verified. The server certificate is used to authenticate the network to be genuine, i.e. guarantees that the access point is legitimate. Based on the certificate, a TLS tunnel will be established between the server and the client that allows secure authentication of the user. If the server provides a certificate that can be verified by the operating system's pre-installed root certification authorities (root CA), then the user will not even be asked to check the certificate, since it will automatically be verified and accepted. On the other hand, if the certificate were not verifiable with root CAs of the OS, then the user would be asked whether the certificate should be accepted. However, how can the user tell if the certificate is valid? While common SSL certificates used for HTTPS contain a fully qualified domain name (FQDM) in the common name (CN) field, which has to match the URL of the HTTPS server, there is no such requirement for certificates used for network access. This means that an attacker could use a certificate that is either accepted by another root CA of the OS, or provide a private certificate that must be verified by the user. Either way, the common name in the certificate is not checked against the server addresses to be used for authentication. However, this check is essential for true mutual authentication, hence must be enabled in the supplicant. Furthermore, a list of legitimate authentication servers must be defined.

Last but not least, as in HTTPS, letting the user decide whether or not to accept untrusted certificates asks too much of the user. Most users will not understand the exception and just accept the certificate. This would allow an attacker to perform the dangerous evil twin attack. Consequently, such certificate exceptions must be disabled.

Let me summarize what settings need to be configured in order to avoid the evil twin attack completely. Every supplicant has to be configured in such a way that the operator explicitly defines the following settings:

- **Server certificates must be verified**. Without this property set, mutual authentication is disabled.

- **The root CA that must be used to validate the server certificate must be specified**. If this setting is left unspecified, any root CA can potentially be used to validate server certificates, thus enabling the evil twin to provide his own certificate.

- **Deactivate user exceptions for unknown certificates**. If such exceptions are allowed, the evil twin could use any certificate, and the user must decide if it should be accepted. This decision cannot be made by the naive user.

- **Define a list of Common Names (CN) of RADIUS servers that must match the CN of the certificate**.
  This ensures that only legitimate authentication servers are allowed. Since the CN of the certificate is checked against the list, an attacker can no longer provide a certificate with an arbitrary CN.

Only if these settings are made, true mutual authentication is performed, and the evil twin attack is no longer possible. All this may seem a bit confusing, complicated and cumbersome. This is true to some extent, and is also one of the reasons why PWLAN operators are not going for 802.1X authentication. Letting the customers configure their devices themselves by providing detailed tutorials would not be acceptable. On the one hand this would ask too much of the customer and would be prone to user errors, and on the other hand this would drastically increase help desk calls. The only way to solve the problem is to find individual solutions for each device. Nevertheless, once the configuration is done, all advantages of 802.1X will shine.

At this point we want to closely look at the correct client configuration for Microsoft Windows XP's built-in supplicant. We choose the Protected EAP (PEAPv0 with MS-CHAPv2) as the EAP method, because PEAP is the most common EAP method and is supported by all Wi-Fi certified products, hence the perfect candidate for public hotspot access. A detailed discussion about which EAP method is best suited will be the topic of Chapter 6.

5.7.1.1 Manually configuring Windows XP for PEAP

Microsoft Windows XP is still one of the most widespread operating systems on the market. A public hotspot operator must support devices running this operating system. However, the WPA2 standard which uses advanced cryptography (AES) for network authentication is only available since Service Pack 2. This is why we consider MS Windows XP with Service Pack 2 or later installed. Older XP versions prior to SP2 would require an update patch[46].

The client settings for wireless networks can be opened by right clicking on the *wireless status indicator* located in the taskbar, then selecting *View available networks* and finally clicking *Change advanced settings*. On the second tab called *Wireless Networks* select the network with the PWLAN operator's SSID (typically "MOBILE" for Swisscom PWLAN) and click *Properties*.

The second tab called *Authentication* reveals the following window (Figure 5-1 left):

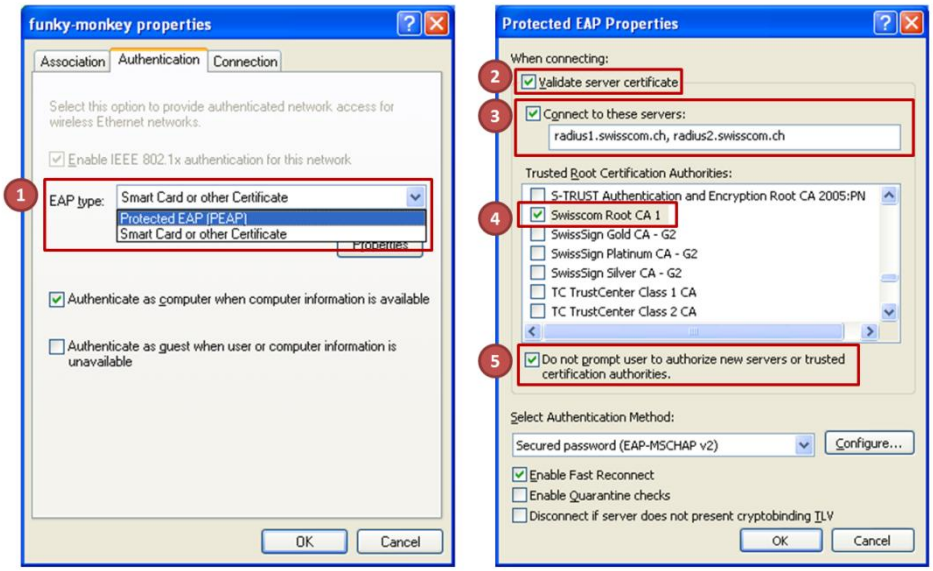

Figure 5-1 PEAP Client Configuration for MS Windows XP

In order to enable PEAP authentication, the EAP type has to be selected accordingly (1). Clicking on the *Properties* button opens the *Protected EAP Properties* window (Figure 5-1 right). Here we can set all properties as described further above. The first check mark will (2) enable server authentication (set by default), (3) is restricting the authentication server to specific addresses and (4) selects the root CA to be used for certificate verification. Finally, (5) deactivates certificate exceptions which would allow the user to accept unknown certificates.

As you can see, these settings cannot be made by the naive user. A missing check mark in the *Trusted Root Certification Authorities* list would result in unsuccessful connection attempts, which would abort without an error message. The customer would either call the support hotline (which can cost the operator up to CHF 60.- per case), or he would give up trying and never use PWLAN again. Hence, the only way to correctly setup the client is to let the operator do the configuration. Naturally, the operator cannot manually configure every single device requesting network access. The question is how can these important settings be made without having direct access to the device? As it turns

out, this is the most serious problem with 802.1X in hotspot environments. Nevertheless, there are solutions to this problem which we now will look at in detail.

5.7.2 Automated Provisioning of the Supplicants

Unfortunately, there is no universal mechanism available that allows the automated provisioning of supplicants (i.e. automatic configuration of the client device). This means, that individual solutions have to be found for the different systems, devices and operating systems. In the context of this thesis, we will analyze two prominent platforms, namely MS Windows XP and Apple's iPhone.

5.7.2.1 Provisioning in MS Windows XP

In enterprise environments, administrators are in full control of all devices and have the possibility to automatically provision the supplicants using Microsoft Active Directory. MS Active Directory allows administrators to define Group Policies that can be pushed to the clients (only to Windows XP/Vista clients running the Windows Zero Configuration (WZC) service). These policies can also include the settings for true mutual authentication for PEAP as described in the last Section. Unfortunately, this solution is not applicable in hotspot environments. Or is it?

5.7.2.1.1 Windows Provisioning Service

In 2003, Microsoft teamed up with several hotspot operators (including Swisscom) and promised a sound solution for PEAP provisioning at public hotspots. The service is called *Wireless Provisioning Service (WPS)* and was introduced in October 2004. WPS is installed on all Windows XP systems with Service Pack 2 or later natively. With WPS, Microsoft introduces a so-called provisioning server into the hotspot operator's network architecture. This server can be seen as a normal HTTPS web server that contains profiles that will automatically be downloaded before the authentication process. However, these profiles contain more than just a correct PEAP configuration. WPS also helps hotspot operators in the process of subscribing new users. Based on the information contained in the profile, a Windows styled sign-up wizard will pop up that allows the unregistered user to enter his name and address, billing information, Credit Card number and let him select different subscription options.

Before we go into more details, look at the following picture which shows some screenshots of an example sign-up process as an unregistered user would see it.

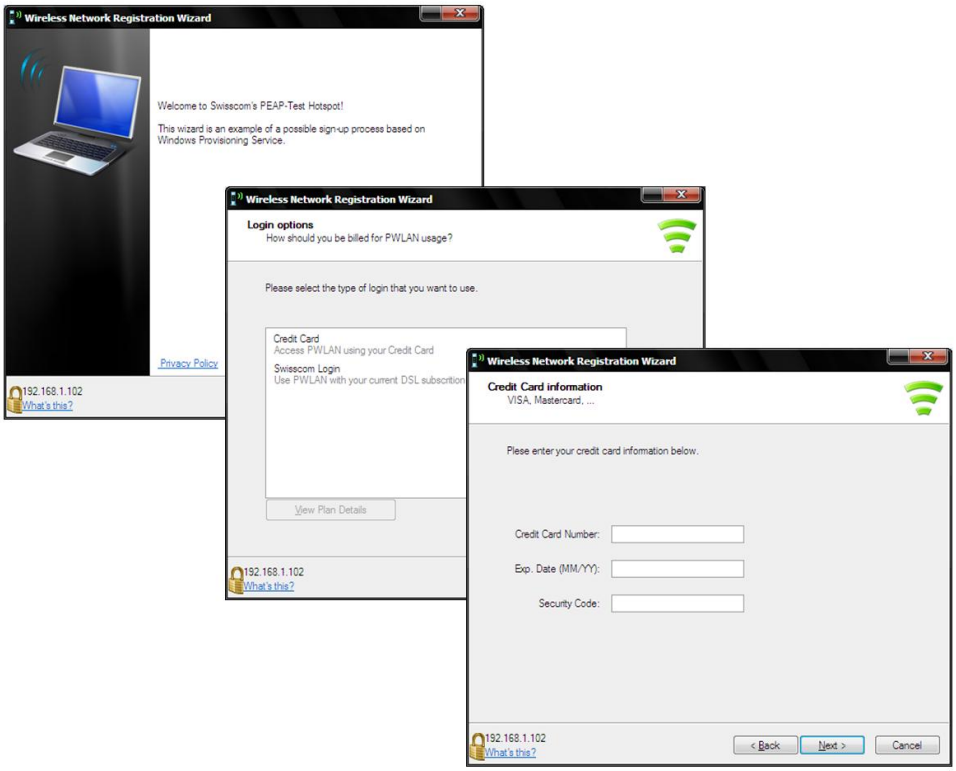

Figure 5-2 Windows Provisioning Service: Example Sign-up Wizard

The three screenshots are only the first three windows of the whole wizard, just to give you an idea. The operator has full control of all texts that are displayed, can brand the wizard with his own logo and define different subscription plans. Microsoft provides a tool which supports the operator in the wizard design process and generates the XML based profiles to be stored on the provisioning server. The tool is called *WPS Authoring Tool* and can be downloaded from Microsoft [47].

In order to see how WPS works from a protocol viewpoint, let us consider the case where a new hotspot customer visits the hotspot for the first time. The process is performed in three phases:

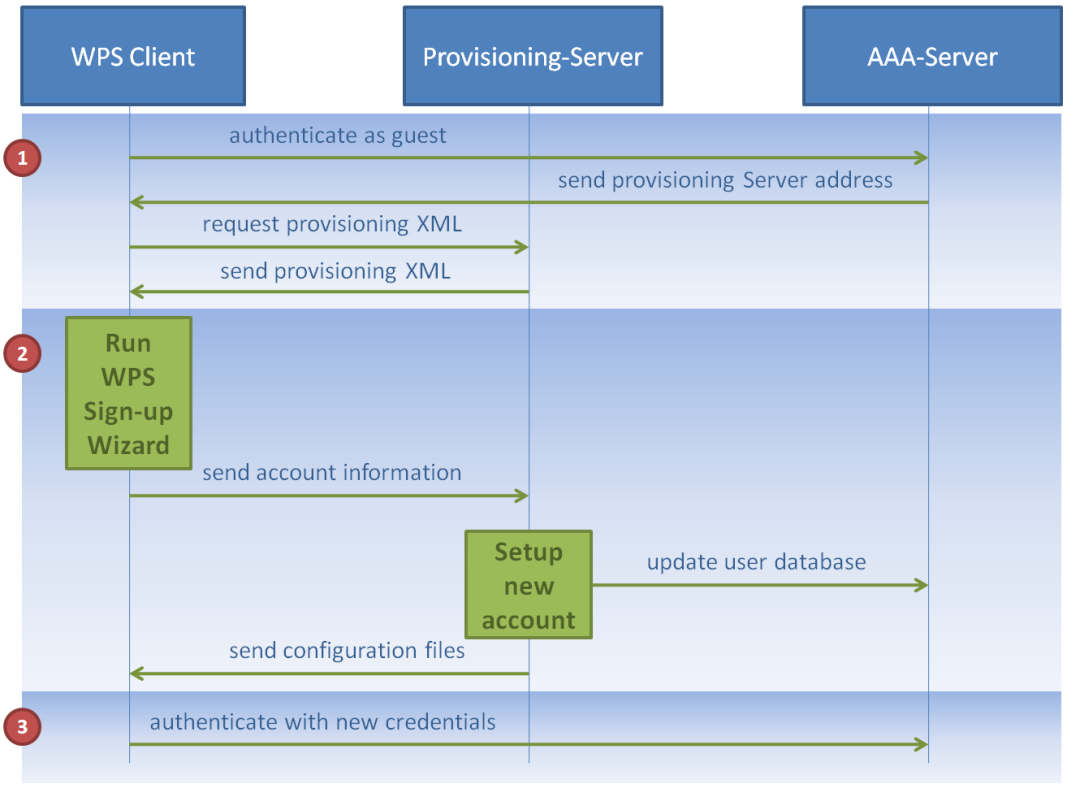

Figure 5-3 WPS Authentication of a new customer

1. The customer walks into the range of an access point of the hotspot operator. Windows XP will show the broadcasted SSID and let the customer connect. The client authenticates using a guest account. This is necessary, since the customer has no own credentials yet. The client automatically connects using no username and no password. The authentication server sends the URL of the provisioning server inside a PEAP-TLV message (PEAP-TLV is a mechanism of PEAP that allows arbitrary data to be exchanged between the client and the server inside a normal PEAP message). Additionally, the client gets a temporary IP address assigned. The client connects to the provisioning server and downloads the XML-files (the profile).
2. Based on these files, the WPS Sign-up wizard will be loaded which prompts the user for identity and other information needed to set up a new account. Once the user has completed the wizard, the information is sent back to the provisioning server. The server checks the payment options, sets up a new account and sends an update to the AAA server. The provisioning server generates a new XML-file which contains the correct PEAP settings, including the username and a password.
3. Once the client has received the configuration file, it automatically disconnects from the access point, and re-establishes a new connection, now using the username and the

password contained in the configuration file. The authentication server informs the access gateway of the successful authentication and assigns a new IP to the client.

It has to be noted here that in the above diagram several functional entities are omitted. Important missing entities are the Access Controller, the Access Point itself, a DHCP server as well as a Microsoft Active Directory server. The Active Directory server is used to manage the accounts and provision the clients. Other LDAP based servers can be used instead if they support the dynamic creation of new accounts. The AAA server in a WPS scenario is typically a Microsoft IAS (Internet Authentication Service) which is integrated in current Windows Server solutions

Of course there would be a lot more to tell about WPS. However, this would be useless, since Microsoft decided to no longer support the WPS service in current and future OSs (Windows Vista and Windows 7). [2]

5.7.2.2 Software Client Requirements

You might ask why we presented the WPS solution despite the fact that it is no longer supported by Microsoft. The reason is the following. With WPS, Microsoft provided a solution to several issues of 802.1X in the environment of public hotspots. It shows very clearly, what problems need to be solved in order to allow secure and seamless authentication in public WLANs.

As already mentioned, not only correct configured devices are an issue, also the user interaction with new customers is a problem. 802.1X tries to avoid CPPs, but how can new users set up a new accounts if no browser is involved at all? How can existing customers renew their subscription or change their password? Microsoft's WPS shows how it should be done: using dedicated client software.

The good news is that when Microsoft stopped their WPS efforts, they published a new APIs that allows developers to use the native wireless services and functions of the Windows OS. The API is called the *Native Wi-Fi API* and replaces the WPS functionality. It is available on Windows Vista, as well on future Windows 7 versions. Microsoft has also down-ported a subset of the API to Windows XP with Service Pack 2 and Hotfix KB918997, as well as Windows XP with Service Pack 3 [48].

[2] The main reason for no longer supporting WPS was because the PWLAN operators would have had to invest too much money into Microsoft's products and licenses. While this would have enabled seamless and secure provisioning of devices, this solution would have been restricted to devices running Microsoft's OSs.

However, a detailed look of the new Native Wi-Fi API is beyond the scope of this thesis. Instead, we will identify the issues that need to be addressed when developing a dedicated software client for 802.1X authentication in pubic hotspot environments. Among others, the most important design requirements are:

- Promoting the secure 802.1X authentication, deployment of the software client
- Interaction with new users, account management for existing customers
- Correct configuration of the devices / supplicant
- Cost control
- Credentials Management

Let us go through each listed item in the next sub-Sections.

5.7.2.2.1 Promoting Secure Authentication, deploying the Client Software

An important question is how customers are being informed about the availability of the secure 802.1X login and how they receive instructions on how to download and install possible client software. PWLAN customers are used to connecting to an open, unprotected SSID and being redirected to a CCP. If 802.1X would be available, a separate SSID would be required. However, how can a customer decide which SSID he should connect to? One option is to hide the 802.1X SSID and only make it accessible for clients with pre-configured devices, i.e. the software client installed. A new user (one without any configuration or software client) would still connect to the open SSID and get redirected to the CPP. He then can use any of the traditional (less secure) authentication methods that he is used to. Additionally, the CPP can be used to promote the use of secure communication and offer a direct download for the client software.

This approach is different from WPS, where the client software is already pre-installed, i.e. integrated in the OS. Naturally, it is not possible to have operator dependent clients pre-installed in the OS.

WPS solves the problem of connecting to the protected hotspot without having valid credentials by providing guest credentials (no username and no password). However, common supplicants of the different operating systems might act differently. For example, pre-WPS Windows OSs will try to connect to the protected access point using the Windows-Logon credentials. This kind of behavior stems from the corporate world, where the Windows Logon often is the same as the network access credentials. However, whatever credentials are being sent by non-configured supplicants, a hotspot authenticator could accept any username and password pair in order to establish a connection and allow assigning an IP address to the client. As with normal CCPs, such un-authenticated clients could

then be redirected to a CPP where the customer can setup a new account and download the client software or network profile. Of course, customers providing credentials that belong to an existing account are automatically authenticated and access is granted without redirection to a CPP.

5.7.2.2.2 Interaction with new users, account management for existing customers

The interaction with new users is already covered the last Section. However, suppose a customer has successfully downloaded the client and gets automatically connected to the secure hotspot each time he walks into the range of a known hotspot. How can he or she be informed about the current price plan of the hotspot? What if his or her subscription period is over and must be renewed? As in WPS, since the client is now installed on the end-device, it would be straightforward to integrate this functionality into the client software. User interaction can be handled comfortably for the customer through a well designed GUI.

However, this would require the client to interact with a server, thus introducing additional complexity, compared to a client-only solution. An alternative would be to provide a web link to a web application where the customer can log in and manage his account in the web browser. Since this also requires additional server capabilities and would require the customer to log in a second time, the former solution is preferable with respect to usability.

5.7.2.2.3 Correct configuration of the devices

The configuration of the device (i.e. the supplicant) is essential to secure networking, as discussed further above. Provisioning these essential settings to the client is the primary intention of the client software. Moreover, once the configuration has been stored on the device, the software client could be removed if the connection profile is kept. The device would still be able to connect to the hotspot because authentication and connection establishment is performed by the OS.

Furthermore, connection profiles (i.e. client configuration) can be stored on the network. The client can be set up to periodically update the settings, enabling the hotspot operator to make changes in the network architecture without bothering the customer.

5.7.2.2.4 Cost Control

One important issue with seamless authentication in public hotspot is cost control. In a true 802.1X enabled network, the client gets automatically connected as soon as he or she walks into the range of a hotspot. However, current price plans are typically based on connection time or data volume. Automated authentication implies that the user might be unaware of the fact that he is connected.

This can be dangerous, because nowadays almost every application will try to access the Internet and check for updates. Application updates can be several hundreds of megabytes large and would cost the customer a fortune, depending on his subscription plan.

Automated updates running in the background or traffic being generated without the user's knowledge can be reduced to some degree by the operator maintaining a blacklist of known update server URLs and ports. Nevertheless, this is no 100% solution since blocking too many ports and addresses could also restrict normal browsing and other applications which the customer wants to use.

The problem might be less severe if we look a bit into the future of PWLAN. Nowadays, customers pay for PWLAN usage based on time or data volume. However, the trend clearly goes towards flat-rates. If customers can use PWLAN as long as they want to with almost no limits, they would not have to worry about updates running in the background. On the other hand, the PWLAN operator still would want to minimize such traffic, since it uses up valuable channel capacity.

5.7.2.2.5 Credentials Management

Last but not least, it has to be mentioned that the storage of the customer's credentials can be problematic with respect to security. Some operators have security policies which currently do not allow user passwords for PWLAN access to be stored on the client. Actually, password based credentials are problematic anyway. Every system is only as secure as the weakest segment in the whole chain. Either passwords are human friendly and thus too weak, or they are strong passwords which the user cannot remember. Letting the user decide on the password is the worst thing that can be done because such passwords can easily be compromised using dictionary attacks and social engineering. On the other hand, strong enough passwords (randomly generated, including symbols, upper- and lowercase letters and 12 to 14 characters long) are cumbersome to be entered and are prone to be written down.

On the other hand, if the credentials can be managed by a trusted client developed by the operator, then they can be stored safely. The problem of human unfriendly passwords also gets alleviated if they have to be entered only once.

We have now seen what problems need to be addressed when 802.1X with EAP, independent of the specific EAP method, is used for public hotspot authentication. All these issues can be solved using a dedicated software client. Using Windows's Native Wi-Fi API, such client software can be kept relatively simple, small in size and be individually customized by the operator at the same time. Nevertheless, such client software has to be developed for each and every operating system and

device. This would at least include Microsoft's OSs, Apple OS X, Apple's iPhone, Nokia devices, Windows Mobile based devices, and maybe the Android platform. This is the challenge that an operator has to face when considering to adapt to 802.1X authentication. It is actually the main reason why 802.1X is still not common for public hotspots. The effort seems too high and too costly. However, as we will see in the next Section, individual solutions can be very simple.

5.7.2.3 Apple iPhone Provisioning

The Apple iPhone represents the new kind of mobile devices which will get very popular in the next years. It was the first device that allowed true mobile Internet browsing and is currently the largest platform today for mobile applications offering over 20'000 different applications (as of 11.2.2009). iPhone users regularly check their emails, download Google maps, watch Youtube videos, buy online music, twitter and upload photos and soon also videos to community web portals. These applications are bandwidth intensive and thus users would welcome seamless and easy to use PWLAN integration. We will now present a solution which does not use a dedicated software client to be installed on the device. Instead, we make use of so-called iPhone profiles [49][50].

iPhone profiles are XML based configuration files that are intended for enterprise administrators to deploy the employees' iPhones in a centrally managed manner. Such profiles can include security policies, VPN connection settings and configuration settings to be defined and deployed using the *iPhone Configuration Utility* which is freely available from Apple [49]. iPhone profiles can be downloaded from a (secured) web site accessed directly with the iPhone, or they can be mailed to the iPhone. Furthermore, profiles are also supported by Apple's iPod Touch devices.

The profile settings in which we are interested with respect to 802.1X authentication are the Wi-Fi settings. The following picture (Figure 5-4) shows screenshot snippets taken from the iPhone Configuration Utility installed on a MS Windows XP machine running as a web service that must be accessed through the browser. A native Mac OS X version is available which offers further deployment options.

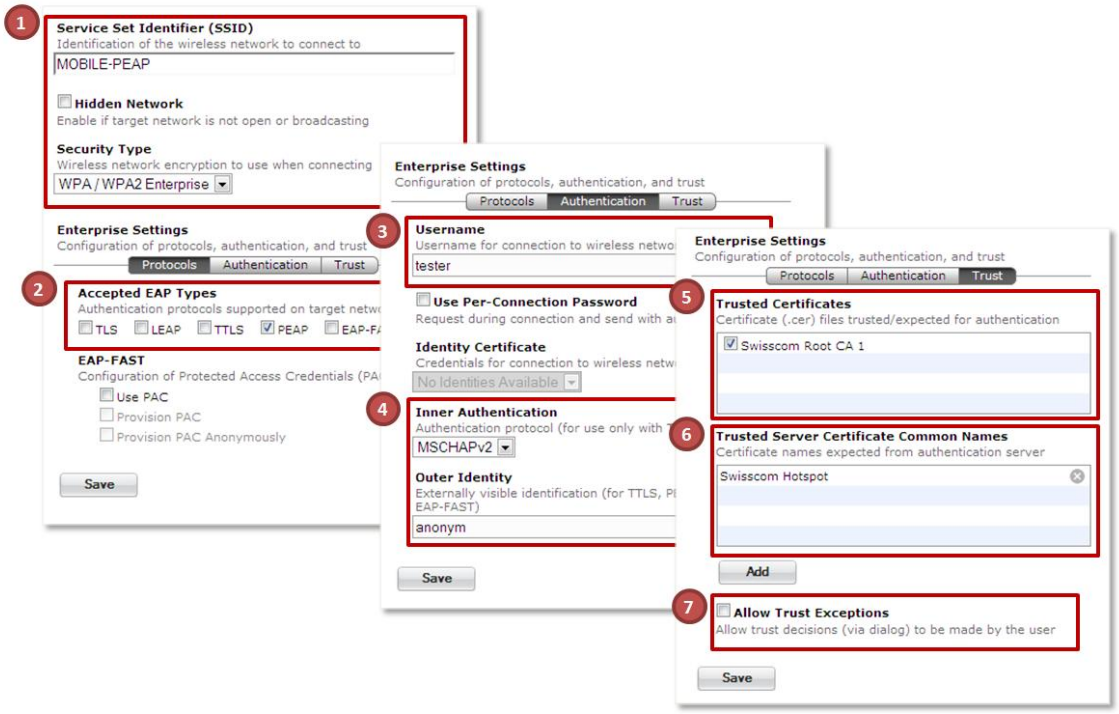

Figure 5-4 Apple iPhone Configuration Utility - Wi-Fi Settings

As with the Windows Supplicant (Section 5.7.1.1), all important settings can be set in iPhone profiles. First and foremost, the SSID of the hotspot and the encryption type can be set (1), the according EAP method be selected (2) and a username for outher- (3) and inner (4) authentication can be defined. The most important settings are related to trust, i.e. the certification authority (5) that will verify the server's certificate (can be included in the profile) and the common names of the authentication servers (6). Finnally, trust exceptions must be deactivated (7) as an important security requirement.

In order to ensure that the profile is legitimate it must be signed using a digital certificate and a private key. Depending on the certificate's origin, the profile will be trusted by the iPhone's pre-installed root CAs. As mentioned above, iPhone profiles can be published on a secure web server where users can download it (e.g. on the CPP) or they can be directly mailed to the phone. When a new profile has been downloaded, the user is prompted whether it should be accepted or not:

Figure 5-5 A new iPhone Profile has been received [50 p. 26]

Once the profile is accepted and installed, the customer can connect to the hotspot simply by hitting the “connect” button in the network settings. The user can then enter the password, which will be stored for future connection attempts. The iPhone will perform true mutual 802.1X authentication based on PEAPv0 with username and password credentials. The next time the user walks into the range of a known hotspot, the iPhone will automatically connect to it, without any user interaction.

This iPhone provisioning based on profiles has not yet been proven to be working as descibed. Actually, there are several issues that still need to be solved. One of them being that it is required to explicitly enter a username in the profile. Of course, this is not what hotspot operators would want to do, since the best solution would be to just have one common profile for all iPhone customers, not a separate one for each user. On the other hand, if iPhone profiles could be generated dynamically for each new customer, then the profile could also include the customer’s password. This would be very convenient for the user and would allow using very strong passwords.

However, these are details that must be addressed within a proof of concept in a next step. Furthermore, this solution does not address the problem of interacting with the customer (e.g. account and cost management). One possible solution would be to still redirect the customer after authentication to a CPP where he or she can see account information. Another approach would be to send an SMS to the customer stating that he currently is connected to a hotspot and what the current prices are. Moreover, the iPhone (and most other cellular phones) do support so-called flash-SMS and network status messages. The former is similar to a normal SMS, however it will be directly

displayed to the user without having him or her to open the SMS inbox. Network status messages are typically displayed in the status bar of the device.

Such short message based interaction with the customer would be also applicable for devices other than an iPhone. Customers connecting to a hotspot with a laptop computer could also receive an SMS indicating the connection status and current prices.

Last but not least, customer interaction could of course be done though a dedicated native iPhone application for account management.

Despite the fact not all issues with iPhone profiles in hotspot environments are solved, this example shows that it can be relatively simple to find individual, device specific solutions to the provisioning problem. This is why we presented my idea.

Actually, one day after writing the above paragraphs, Apple presented the new firmware 3.0 for the iPhone and the iPod Touch (on 17. March 2009). It will be available for customers in summer 2009. Most interestingly, Apple now does support EAP-SIM, which allows the iPhone (not the iPod Touch) to seamlessly log on to hotspots (those that support EAP-SIM) using the credentials stored on the SIM card. Furthermore, Apple introduced Wi-Fi auto login and on-demand VPN. At the time of writing, it cannot be said what exactly this is and how it works. Supposedly it is some kind of automatic profile loading mechanism that allows the automatic provisioning of the iPhone when walking into the range of a hotspot. Another guess is that the browser acts as the traditional WISPr XML Smart Client, making the SMS one-time password procedure transparent to the customer. However, if the update will include automated 802.1X provisioning (resp. profile loading) is still unclear at the time of writing.

Chapter 6 Evaluation & Conclusions

The last chapter summarizes the above findings and draws conclusions. An evaluation of current and future EAP methods will analyze their suitability for public hotspot authentication. Recommendations for PWLAN operators considering adapting to 802.1X authentication will be provided.

6.1 Evaluation

Apples efforts in automated hotspot login again show the trend towards more private customers wanting to access PWLAN with their cellular phone, Smartphone or PDA. The current solution, which involves receiving a one-time-password through SMS, is too cumbersome. On the other hand, PEAP authentication based on password credentials might not be secure enough for the operator, and EAP-SIM is restricted to devices with integrated SIM-Card interfaces. As can be seen, even if an operator chooses to provide 802.1X authentication on his hotspots, he still has to decide which EAP method to support. Each method has different features that must carefully be considered and evaluated. On the other hand, current WISPr implementations provide great flexibility. Actually, even if 802.1X authentication is supported by the hotspot, UAM with captive portal pages would have to co-exist in order to allow the legacy devices not supporting 802.1X to log in.

In this evaluation we include the current WISPr UAM method as well as the WISPr XML Smart Client solution. The focus of the evaluation however, is 802.1X authentication, considering the different EAP methods that provide mutual authentication. We will categorize the EAP methods according to their underlying concepts, e.g. password only, server certificates plus client password, etc. The goal of the evaluation is to find a favorite candidate for each category and to discuss the advantages and disadvantages of the different approaches.

6.1.1 Evaluation Measures

The evaluation does not include any empirical measures such as connection delays or log-in speeds. Instead we will compare the different approaches on a more abstract, conceptual level. Important aspects are the following:

- **Security and privacy of the customer**
  How and to what degree is the customer protected against common attacks? What kinds of attack are still possible? These questions will be answered as well as how well the customer's

privacy is protected. Depending on the authentication method, an attacker can sniff a unique user identity (username or ID) whenever the user tries to connect. This allows tracking a user based on the signals transmitted from the client. Other methods do not send the unique identity in the clear and use pseudo identities or anonymous IDs instead.

- **Usability** (from the customer's point of view)
  Usability is the key to satisfied customers. However, the usability vs. security trade-off often means that a higher degree of security implies reduced usability. As we will find out, this is not the case with 802.1X authentication when dedicated software clients are used. The questions to be answered are: How difficult or easy is it to set up a connection to a public hotspot? How many buttons have to be pressed, what kind of data has to be entered and must this same process be repeated each time a customer wants to re-connect?

- **Interoperability** (supported devices)
  Interoperability is a very important aspect especially for a hotspot operator, because he has to support as many different devices as possible. Different devices include a diversity of form factors, ranging from common laptop computers to Smartphones, PDAs and mobile gaming consoles. Furthermore, interoperability means also to support all common operating systems available, including MS Windows OSs, MAC OS X, Linux, Apple's iPhone, Symbian, Android and others.

- **Deployment**
  The term deployment stands here for distributing client software, network profiles or configuration settings to the client device. With traditional WISPr UAM authentication, this is not an issue at all, since the whole process is browser based. For 802.1X, deployment can be the weakness of the solution. Individual client software has to be provided, which makes deployment of the devices a challenge.

- **Advantages and disadvantages**
  Naturally, each approach has its individual advantages and disadvantages that can be determining factors.

6.1.2 Approach Categorization

In the context of this thesis, we have seen different approaches to PWLAN authentication. We have furnished several arguments, why 802.1X port based authentication should be used also at public hotspots. In order to allow a comparison of the different approaches known, we categorize the

methods according to their underlying concepts. The categorization we have chosen for this evaluation is the following, which includes the traditional WISPr UAM method.

- **WISPr UAM CPP**

  The Universal Access Method (UAM) is the traditional approach for PWLAN authentication. WISPr is a recommendation document that states how UAM should be implemented, i.e. using Captive Portal Pages (CPP).

- **WISPr XML Smart Client**

  In addition to the CPP approach, WISPr specified a WISPr XML Smart Client interface. Smart Clients are dedicated software clients that will be installed on the client device. The tool simulates the UAM HTTP traffic based on XML files.

  We will consider Smart Clients that perform WISPr XML authentication from a general perspective, despite the existence of several third-party solutions available today (BOINGO, WHISHER, etc.).

- **EAP: Password only**

  EAP methods that are based on passwords only are not suitable for public hotspots. Nevertheless, these methods are included in the evaluation due to their popularity and in order to state explicitly their weaknesses.

  Methods considered are: *EAP-MD5, LEAP, and EAP-FAST*

- **EAP: Certificate only**

  These EAP methods use digital certificates on the server as well on the client machine. Such methods are often used in enterprise environments, since they provide strong mutual authentication. One great advantage of client certificates is that users do not have to remember passwords.

  The only method considered in this category is *EAP-TLS*. Other methods, such as PEAP, are also capable of using client side certificates. However, EAP-TLS is the most widespread certificate only method today. PEAP, which is even more popular when the client authenticates using password credentials, is evaluated in the next category.

- **EAP: Server Certificate plus Client Password**

  EAP methods that use server side certificates to authenticate the network and let the client authenticate with username/password credentials are perfect candidates for hotspot authentication.

Methods evaluated are: *PEAP* and *EAP-TTLS.*

- **EAP: SIM based**

  SIM based authentication is of special interest to telecom providers that already have an existing cellular phone customer base. The credentials that are stored on the SIM card are primarily used for GSM authentication, but can also be used as PLWAN credentials.

  Today, only two SIM based EAP methods are known, namely EAP-SIM and EAP-AKA. As described in Section 3.5.7.3, EAP-AKA is the UMTS equivalent of EAP-SIM, thus can be considered identical from a conceptual point of view. Hence, we include only *EAP-SIM* in this evaluation.

- **EAP: Special**

  Since over 40 EAP methods are known today (not standardized), naturally, there are also exotic protocols that do not fall into one of the above categories. One such EAP method is *EAP-TPM* [20][21]. EAP-TPM is based on the Trusted Platform Module (TPM) which is integrated in most modern computer systems. Each TPM module includes a worldwide unique identity number which is stored in a protected manner. Based on this number, the trusted platform allows automatic generation and validation of a variation of a digital certificate that can also be used for authentication in wireless networks. However, EAP-TPM is still in Internet draft status which will expire in September 2009 [20].

6.1.3 WISPr UAM CPP

The WISPr Universal Access Method (UAM) is described in Section 4.3.2 and an example of a Captive Portal Page (CPP) is given in Section 4.3.1. Note that the terms WISPr, UAM and CPP are often used interchangeably with the same meaning, all referring to the login page that is displayed in the browser to let the customer authenticate. For the rest of this evaluation, we will use the abbreviation CPP.

CPPs are used today by most PWLAN operators to perform authentication. The main reason for this is that CPPs can be accessed with any device that provides a browser, independent of the operating system, device type or manufacturer. This makes this solution very interoperable, since no software clients have to be provided and no device specific support is needed. Moreover, it is possible to provide different versions of a CPP to accommodate different screen resolutions. For example, Swisscom's CPP detects whether a customer uses an iPhone or a PDA and automatically switches to a CPP which is optimized for small screens (see Figure 6-1 image).

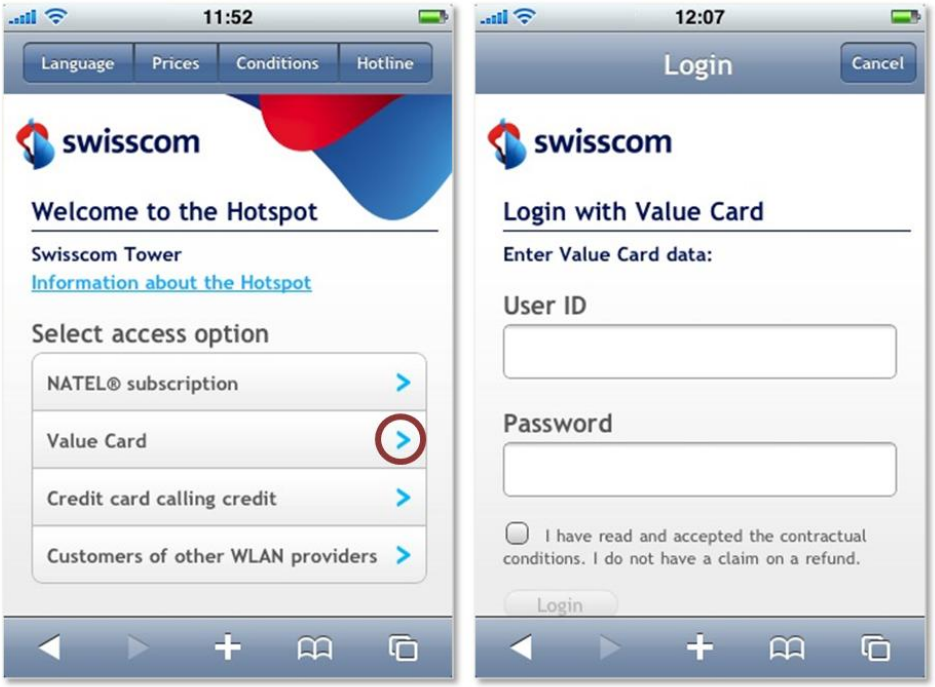

Figure 6-1 iPhone version of Swisscom's CPP

While this makes it more comfortable for the user to use the CPP with Smartphones, it does not change the way how the customer authenticates. A browser has to be started explicitly and the credentials have to be entered each time the user wants to connect. This process is fairly uncomplicated for first time access to a public hotspot, but way too cumbersome for frequent usage.

One of the most significant drawbacks of this method is its low security level. There is no way of telling whether the hotspot to which the user connects to is legitimate. Also, there is no link layer encryption. The authentication process itself is protected through HTTPS which is based on the server's certificate. However, after CPP authentication, everything is sent in the clear, except for traffic that is protected by the applications themselves (email with SSL, VPN, etc.).

6.1.4 WISPr XML Smart Client

Smart Clients according to the WISPr XML specification do support the user in the UAM authentication process. Credentials have to be entered only once, and the Smart Client can be configured to connect automatically to a hotspot as soon as it is in range.

Furthermore, Smart Clients have the possibility to provide special GUI components that allow direct interaction with the customer, independent of a browser. The customer can be notified of the current price plan, is notified that he is actually connected to a commercial hotspot and it allows the customer to manage his account in a convenient way.

Nevertheless, Smart Clients also use the insecure UAM method, hence there is no security mechanism protecting the customer. There is the possibility to automatically establish a VPN connection once the user is authenticated. While this will encrypt the traffic, it does not alleviate the problem of rouge access points and other attacks that can be performed before the VPN connection is established (see Sections 5.2 and 5.3).

Smart Clients provide the seamless PWLAN experience that many PWLAN customers request. There are already many such Smart Clients available, but they are usually not declared as being true WISPr XML Smart Clients. However, third-party applications, such as the Devicescape Easy-Wi-Fi Client [44] and the Whisher wifi.com Client [51] are usually closely coupled to their own subscription plans or have communities that also share their private WLAN access. For PWLAN operators, in order to be in full control of the user experience, a branded Smart Client which is customized by the provider would be strongly recommended.

6.1.5 802.1X with EAP

802.1X port based access is described in Section 3.3 and the different EAP methods in Section 3.5. 802.1X always uses EAP as the authentication framework. However, which specific EAP method has to be used is not defined. Besides providing strong security, one important advantage of 802.1X with EAP is that once the EAP framework is installed in the operator's network, any EAP method can be easily integrated. This means that at any time the operator can switch to another EAP method, without having to change the network architecture. This is also true for future EAP methods yet to be offered.

The question is: Which method is best suited for PWLAN that is available right now? Unfortunately, there is no best solution that fits all scenarios. Each method has its own features which can be of special value to specific operators. The rest of this evaluation considers most common EAP methods, categorized as described further above.

6.1.6 EAP: Password only

As already stated in the description of the categories, EAP methods that are based solely on passwords are not suited for PWLAN authentication. This stems from the fact that the network is not authenticated at all. This would allow the evil twin attack to be performed even without having to provide a faked CPP.

- **EAP-MD5**

  As already mentioned in the introduction Section of EAP-MD5 (3.5.1), this method is

deprecated. EAP-MD5 was designed for switched wired networks where it is not easily possible to sniff data packets. Furthermore, there are no mechanisms for key management, which would be crucial for encrypting the communication channel after authentication.

- **LEAP**

  LEAP is Cisco's first proprietary EAP method which also uses password hashes for authentication. In contrast to EAP-MD5, LEAP authenticates the network, also using password hashes. The problem with password hashes is that they are vulnerable to offline dictionary attacks. There is a tool called ASLEEP available which performs the attack within seconds. Cisco reacted to the tool with a successor of LEAP: EAP-FAST.

- **EAP-FAST**

  EAP-FAST uses a PAC (Protected Access Credentials) instead of password hashes. Since this does not require certificates, it is easier to maintain and less resource intensive than certificate based EAP methods. However, since no automatic (and secure) mechanism for automatic provisioning of the PAC without the use of certificates exists, the PAC would have to be distributed manually, e.g. on an USB-Stick or a CD-ROM. This is different to distributing password credentials, because the PAC is actually a file.
  Furthermore, EAP-FAST is a proprietary standard of Cisco. It would be necessary to also deploy an appropriate supplicant to the clients, because EAP-FAST is not as widespread as for example PEAP.

6.1.7 EAP: Certificate only

A strong requirement for 802.1X authentication in wireless systems where encryption keys have to be derived is mutual authentication. This means that not only the user is identified, also the network has to be proven to be legitimate. This is typically achieved by the server providing a digital certificate that must be verified by the client. If the client itself also uses certificates for authentication, we speak of certificates only methods because server and clients use certificates. However, the complexity of the management, distribution and verification of client certificates make this approach unsuitable for public hotspot authentication.

- **EAP-TLS**

  EAP-TLS is very common in large enterprises, since it is considered the strongest authentication method for wireless LANs. The problem with client side certificates however, is that the certificates have to be deployed to the client machines prior to first network access. In enterprises, this can be done by administrators who are in full control of all clients.

In public wireless LANs, the operator cannot touch the customer's devices (except if he also sells them). Furthermore, as already mentioned, using certificates involves having a sophisticated infrastructure (PKI) and expertise, thus introducing unnecessary expenditure. While this is not a burden when only server certificates are used (since there is typically only one server), client certificates require a much more sophisticated PKI, since the number of client certificates can be very large. Last but not least, another issue with client certificates is the safekeeping of the private key in the customer's device. Once a private key is compromised, the current key must be revoked (using a CRL that must be periodically updated) and a new certificate must be deployed. All this makes such solutions impracticable for PWLAN, where thousands of customers have to be managed.

6.1.8 EAP: Server Certificate plus Client Password

EAP methods that fall into this category are well suited for PWLAN authentication. They overcome the problems of client side certificates, because they let the client authenticate with username/password credentials, without being less secure.
The server first proves his identity with a certificate that can be verified by the client. This can either happen through private certification authorities or with pre-installed root CAs. The later is preferable, because private CAs would still require installing the CA's certificate on the device. Server certificates that can be verified by common root CAs can be bought from VeriSign (or similar) for a subscription of a few hundred dollars annually, depending on the intended usage.

Based on the server certificate, a secure TLS tunnel between the client and the authentication server is established. Since all traffic is encrypted once this tunnel is set up, the client can authenticate using any legacy method. An attacker has no possibility to sniff the user's identity or a password, thus greatly enhancing the customer's privacy.

- **EAP-TTLS**

  First of all, EAP-TTLS is very similar to PEAP. The only difference between the two protocols is that EAP-TTLS allows any legacy method such as PAP and CHAP as the inner authentication protocol, while PEAP allows any other EAP method to be executed within PEAP. The only non EAP method that is allows to be executed within PEAP as the inner authentication method is MS-CHAPv2 (see Section 3.5.3 for more details).

  Because PEAP and EAP-TTLS are so similar, they are both suited to PWLAN authentication. However, PEAP is much more widespread, since it is integrated in most OSs, the most

important one being Microsoft Windows. EAP-TTLS would require a separate supplicant to be installed on the clients.

- **PEAP** (with MS-CHAPv2)
  PEAP fulfils all requirements for PWLAN authentication. Mutual authentication, password based credentials, automated login, key management and link layer encryption. If configured correctly, even the user's identity is hidden during the authentication process, hence protecting the customer's privacy. Furthermore, PEAP is widely available and supported by most OSs. This means that the built-in supplicant of the OS can be used and no supplicant has to be installed. Nevertheless, also the supplicant of the OS still has to be configured, as discussed in Section 5.7.1.

  For all the reasons stated in the last Section, PEAP is considered best suited for hotspot authentication when password credentials are being used (in combination with server side certificates). However, if PEAP is used as the outer authentication method, one has still to select an appropriate inner authentication. This on the other hand, is not so problematic, since almost any method can be used, also insecure ones, since they are protected by a TLS tunnel. MS-CHAPv2 is very commonly used as inner authentication method and also suits into the public hotspot environment.

Unfortunately, methods such as PEAP and EAP-TTLS which use server certificates also have some issues that must be considered. A general problem with certificate based mechanisms is that they typically must perform relatively many message exchanges between the server and the client. This comes from the fact that they have two phases when mutual authentication is performed. In the first phase, the TLS tunnel is established based on the server's certificate, and in the second phase, the client authenticates. For example a PEAP authentication with the very basic EAP-MD5 inner authentication method requires 6 round trips, hence 12 messages to be exchanged. If the server additionally has to contact a remote RADIUS server in roaming scenarios, there will be further delays.

Another issue is the simple fact that passwords are used. The problem is not that the passwords often are too weak, since EAP-TTLS and PEAP protect the passwords by transmitting them in the secured tunnel. However, the username and the password must be somehow communicated to the customer and manually be entered. This introduces an additional step in the sign-up process for new customers. Furthermore, passwords can become compromised by malicious software running on the client's computer. This is a fact that one has to be aware of. A possible solution to this problem

would be to protect the password through carefully designed client software, which is needed anyway for the configuration of the protocol. An even better idea would be to not use passwords at all and rely on dedicated hardware instead, such as a SIM card.

6.1.9 EAP: SIM based

As described in the Section about SIM-based authentication 3.5.7, EAP-SIM (and EAP-AKA) use the credentials stored on the SIM card. The SIM card can be accessed by a cell phone, or by dedicated SIM card readers available also for standard and laptop computers (using USB, PC-Card, PC-Express, etc.). SIM cards are a special form factor of a smart card. Smart cards are considered to be the most secure computer systems available today. This comes from the fact that they can protect sensitive data, such as user credentials, physically and electronically. Protected data can only be accessed if the correct PIN-Code (Personal Identification Number) has been entered. Three false PIN attempts result in access denial. Additionally, depending on the type of the smart card, protected data gets immediately destroyed if the card is physically tackled. Furthermore, the simple fact that a SIM card is something that a customer physically owns makes it much more secure compared to other, software based (password, certificates) credentials, since a SIM card cannot be copied or compromised.

Moreover, EAP-SIM authentication can be completely automated. The user is not involved in the process, aside from obtaining and inserting the SIM card into the device and entering a PIN code when the device is started. This makes this approach very convenient for the customer.

However, EAP-SIM authentication requires the PWLAN operator to have access to the GSM Authentication Centre (AuC) in order to authenticate the customer. Naturally, telecom operators such as Swisscom that also want to offer PWLAN to their customers can easily integrate their existing AuC into the PWLAN network. To them, EAP-SIM is the perfect candidate for authenticating Wi-Fi devices with integrated SIM cards.

On the other hand, EAP-SIM's greatest advantage of being a hardware based approach to authentication can also be an issue. Relying on the presence of dedicated hardware (i.e. the SIM card and a card reader) drastically reduces the range of supported devices. Currently, only few devices do support EAP-SIM authentication. While this might change for so-called dual-mode phones with 3G and WLAN interfaces in the near future, most devices do not have a SIM card interface. How can laptop computers, netbooks, Internet enabled tablet PCs or PDAs be authenticated in an EAP-SIM enabled network? There are two options: either to allow these devices to use other EAP methods for authentication or to provide dedicated card readers that can be attached to the device. The latter

option is especially interesting for the customer when the SIM reader comes in the form of a USB dongle that further contains a separate UMTS/3G modem. The combination of 3G and PWLAN gives the customer the possibility to use the cheap and fast WLAN technology where available while guaranteeing good coverage when no access points are in range (using 3G). Such scenarios are discussed intensely under the term *3G/WLAN convergence*. EAP-SIM plays a very important role in this battle of technologies, allowing the two to co-exist while giving a seamless user experience to the customer. Swisscom (as well as many other telecom providers) already offer such 3G/PWLAN USB dongles with great success.

6.1.10 EAP: Special (EAP-TPM)

We have seen how hardware based authentication such as EAP-SIM makes the authentication process seamless without involving the user. The requirement of dedicated hardware and a separately sold SIM cards on the other hand, is limiting the range of supported devices. Integrating or attaching separate card readers or modems also introduces additional cost for the operator. This expenditure could be avoided, if the hardware used for authentication were already present in the user's device. This is exactly the idea of EAP-TPM, a new EAP method in the process of standardization (see Section 3.5.8).

Since EAP-TPM relies on dedicated hardware which is already integrated in most modern computer systems, it does not require any additional hardware or SIM-card to be installed. This makes EAP-TPM well suited for public WLAN authentication.

However, the method is not yet standardized. Swisscom is currently running a project on the subject and testing first prototype implementations [23]. While there are still technical issues to be solved, EAP-TPM is a very promising EAP method that would perfectly fit into the PWLAN environment.

Also with EAP-TPM, there is an issue that must be considered. Compared to EAP-SIM, where the provider hands out the SIM card to the customer, the TPM module must be somehow registered with the provider prior to first network access. The ID contained in the TPM must be linked to the real identity of the customer in order to allow billing for service usage. On the other hand, this is not different from approaches, be it username/password credentials that need to be communicated, certificates that must be distributed or SIM cards that need to be handed over to the customer.

6.2 Evaluation Table

The table on the next page summarizes the above findings.

	Universal Access Method (WISPr compliant)		Password Only			Certificates only	Server Certificate plus Client Authentication (username/password credentials)		SIM based	Special
	WISPr UAM CPP	WISPr UAM XML Smart Client	EAP-MD5	LEAP	EAP-FAST	EAP-TLS	EAP-TTLS	PEAP	EAP-SIM / EAP-AKA	EAP-TPM
Standard	Not standardized, WISPr recommendation document		RFC 2284 *Deprecated*	Cisco proprietary	Cisco proprietary (RFC4851)	RFC 5216	RFC 5281	Open standard, WPA/WPA2	RFC 4186 / 4187	draft-latze-emu-eap-tpm-00
Mutual Authentication	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Server Authentication	None	None	None	Password hash	PAC	Certificate	Certificate	Certificate		Special certificate
Client Authentication	username & password, Credit Card, Voucher, Mobile subscription	username & password	username & password	username & password	username & password	Certificate	username & password	username & password	SIM credentials (IMSI, Secret Key)	Special certificate
Key Management	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Security	LOW, Authentication process secured with HTTPS, no channel encryption	LOW, Authentication process secured with HTTPS, no channel encryption	Very low, Vulnerable to offline dictionary attacks if weak passwords are used	MEDIUM, Vulnerable to offline dictionary attack (ASLEEP)	MEDIUM, MITM attack possible during automated provisioning	STRONG, Best possible, if certificate stored on a secure hardware token.	STRONG	STRONG	STRONG	STRONG
Evil Twin vulnerable?	Yes	Yes	Yes	Limited	Limited	No[1]	No[1]	No[1]	No	No
Interoperability	Best possible, works with all devices with integrated browser	Depends on availability of the Smart Client	*Deprecated method*	Cisco devices only	Cisco licences products	Supported by most Wi-Fi certified products	Supported by most Wi-Fi certified products	Supported by most Wi-Fi certified products	Only devices with integrated SIM card interface	Devices with Trusted Platform Module
Usability (customer)	Cumbersome for frequent usage	One-click solution or automated logon possible	All EAP methods: Seamless / automated authentication for pre-configured devices makes 802.1X with EAP very comfortable to be used for frequent usage. First time setup often requires dedicated software clients to be installed (see deployment).							
Deployment	None	Smart Client download & install	*Deprecated method*	Supplicant download & install, Credentials have to be communicated.	Manual PAC distribution recommended (USB-Stick, CD-ROM, etc.)	Certificate distribution problematic	Network Profile (Supplicant Settings) required, dedicated Software Client recommended. Credentials have to be communicated to the customer (Web, email, postal service)		SIM Card distribution (if not already present)	Registration of TPM required
Advantages	Universally accessible, supports a variety of different credentials, possibility to place advertisements on CPP	Comfortable for the customer	*Deprecated method*	Lightweight, does not need certificates and PKI	Does not need certificates and PKI	Very secure in combination with secure tokens	Strong mutual authentication without client certificates, automated login once configured	Strong mutual authentication without client certificates, automated login once configured, widely available	Relies on physical presence of hardware (increases security), automated login, allows seamless 3G/PWLAN experience	TPM widely available in laptop computers, automated login
Disadvantages	Low security level (unilateral authentication, no channel encryption), cumbersome for frequent usage	Low security level (same as UAM CPP)	*Deprecated method*	Vulnerable to dictionary attacks, requires Cisco hardware	Vulnerable to MITM, provisioning of PAC problematic	Overhead of client side certificates (PKI, resources, expertise, costs), safekeeping of private key problematic	Needs careful configuration of the supplicant. Not as prominent as PEAP.	Needs careful configuration of the supplicant.	Range of supported devices limited.	Not yet standardized. Limited to devices with TPM.
Suitable for PWLAN	Yes, in combination with VPN software		No	No	No	No	Yes	Yes	Yes	Yes

[1] **Requires careful configuration of the supplicant.**

6.3 Conclusions

The traditional CPP approach for PWLAN authentication provides a universal mechanism that can be used by almost all mobile devices that have a browser interface. The hotspot operator does not have to deal with platform specific issues and does not need to provide client software. Furthermore, it can be seen as the de-facto standard for PWLAN roaming, since almost all PWLAN operators are using CPPs.

However, for frequent usage, the CPP solution is much too cumbersome. Customers want to use PWLAN without having to enter their credentials each time they want to use a public hotspot. Here, a Smart Client can ease the process for the user, without having the operator change anything in the network architecture. The only, yet really important issue with this solution is security.

UAM, independent of whether it is performed by the user through the CPP or by a Smart Client, does not provide link layer encryption. Although the authentication process itself is secured through HTTPS, all traffic before and after the authentication is sent in the clear. Even worse is the fact that UAM does not authenticate the network (i.e. the RADIUS server). This allows the dangerous evil twin attack to be performed.

If the UAM authentication process is automated using a Smart Client, the customer does not even see a CPP. This makes it even easier for the evil twin to perform his malicious attack, since he does not have to fake the CPP. All he needs to do is to download and install software onto his laptop computer, enter the SSID of the PWLAN operator and sit there and wait. PWLAN customers will connect to the rogue access point (simulated by the hacker's computer) without even recognizing what is happening. The Smart Client will automatically present the username and the password to the attacker.

It becomes even more dangerous when the customer enters his Credit Card number in the CPP. If the evil twin provides an authentic looking CPP, most users would not see that the page is not secured and authenticated through HTTPS. The next thing that the customer sees is a massive credit card bill, and there is no way of telling that an evil twin attack has been performed and who has stolen the credit card information.

802.1X port based authentication with EAP overcomes the security issues of UAM. EAP provides a framework for specific EAP authentication methods to be performed. However, in wireless networks, it is a strong requirement that the EAP method provides mechanisms for key management in order to derive encryption keys that are used for encrypting the data traffic after the authentication.

For that reason, simple password based EAP methods such as EAP-MD5 cannot be used for wireless LANs. Ciscos' LEAP and EAP-FAST are also not suited in public WLANs since they too are vulnerable to dictionary attacks or, in case of EAP-FAST, a MITM attack can be performed (which similar to the evil twin attack).

Authentication that is based on client and server certificates provide very strong, mutual authentication. However, client certificates simply cannot be effectively managed by a PWLAN operator, since there is no way of automatically distributing the certificates to the clients in a secure and seamless manner. Therefore, EAP methods that use server side certificate to prove the identity of the network and let the client authenticate using any other method are best suited for authentication in public hotspot environments. The two EAP methods that provide this flexibility are EAP-TTLS and PEAP.

The problem with this approach is that the clients must be carefully configured. If default settings are used or the customer has to configure the device manually, the provided security mechanisms cannot guarantee to be working as expected. For this reason, the PLWAN operator has to provide network configuration profiles, which could be included in a dedicated software client. Such a client can further provide a user friendly GUI that allows the operator to interact directly with its customers. On the other hand, client software has to be developed individually for each operating system and most popular Smartphones and PDAs. This introduces some cost in the beginning, but in the long term, customers will feel safe and comfortable using PWLAN, hence attract new customers.

Currently, SIM based authentication is probably the safest and most comfortable way for the customer to be authenticated at a hotspot. However, in order to check the customer's identity, the PWLAN operator has to have access to the AuC, which should be no problem if the operator is also running a GSM/UMTS network. Unfortunately, by far not all mobile devices do have integrated SIM card readers. In order to also support the wide range of mobile devices without a SIM card interface, the operator has to provide an alternative EAP method, which is based e.g. on username/password credentials.

Furthermore, EAP-SIM allows the operator to offer 3G/PWLAN USB dongles with integrated SIM cards to be attached to standard laptop computers. Such a dongle (e.g. Swisscom's Mobile Unlimited products) give the customer the very good Wi-Fi coverage using the 3G network when abroad and cheap and fast PWLAN access in cities.

Once the operator is running 802.1X authentication with whatever specific EAP methods available, he can quickly adapt to new EAP methods, such as EAP-TPM. EAP-TPM overcomes the drawbacks of client side certificates and the problem of supporting mobile devices without SIM card readers.

6.4 Recommendations

At this point, it should be clear to the reader why the traditional WISPr UAM method is no longer fulfilling the requirements of secure, easy to use public WLANs. The only alternative is to use 802.1X with EAP authentication.

EAP-SIM can be considered the optimal method for mobile devices with integrated SIM card readers. Such devices will get more and more popular in the next years. Offering EAP-SIM authentication to PWLAN customers is recommended in case the PWLAN operator is also running an AuC, i.e. has an existing GSM/UMTS customer base.

However, EAP-SIM is supported by a only small range of devices. All others should have the option to authenticate based on username/password credentials. PEAP with MS-CHAPv2 fulfils all the requirements for authentication in wireless networks and is also well suited for public hotspots. The PWLAN operator should first find out what devices he wants to support and consider to develop or buy and brand individual client solutions (i.e. client software) for the most important devices. This should include the most popular OSs such as Microsoft Windows and MAC OS X as well as common Smartphone OSs, such as Windows Mobile, the iPhone OS X and maybe Android.

Such client software allows the configuration of the supplicant as described in Section 5.7.1 and provides the option for a convenient GUI for user interaction. A list of issues to be considered for software clients was given in Section 5.7.2.2.

Nevertheless, the traditional UAM method should be running in parallel to 802.1X. This is mainly because existing customers cannot adapt fast enough. However, once they see the advantage of secured public networks, they will switch and even be willing to pay more for it.

Since a roaming is also a key aspect for PWLAN operators, a dedicated software client that authenticates using 802.1X with EAP should also support the WISPr XML Smart Client authentication. The client should first try to authenticate using the secure 802.1X approach, and then, if not available, authenticate using the Smart Client interface which the roaming partners will support.

Last but not least, it should be noted that seamless PWLAN authentication only makes sense if current price plans are also adapted. Seamless authentication implies that customers do no longer have to explicitly log in to public networks, thus are not always aware of the fact that they are connected to a commercial hotspot. Pricing of PWLAN services should be as transparent for the customer as the authentication method is. Flat-rate based subscriptions would provide the required level of transparency. If offered as an option to the existing mobile phone subscription for a few additional francs per month for e.g. two hours of secure PWLAN usage would be a welcomed by many customers.

Apple's iPhone and Acer's EeePC[3] opened a new chapter in mobile technology. Youtube, Facebook, Google Maps, Skype and Twitter are only a few actors which are changing the way urban people communicate and live. If public wireless networks want to play a role in this game, then PWLAN operators must act now.

6.5 Future Work

This thesis discusses the concepts and different approaches for authentication at public hotspots. We concluded that 802.1X and EAP should be used for secure authentication of customers in PWLAN environments. We found that EAP-SIM in combination with PEAP would be the preferred choice with respect to usability and security. However, we only briefly discussed the process of adapting the provider's network architecture to the EAP framework. Aside from upgrading the system components, other aspects need to be considered. One important consideration is how 802.1X and EAP authentication can be offered in parallel to the traditional WISPr mechanism, to give the customer the choice between the two. However, since Swisscom has already implemented the EAP framework in its PWLAN network (offering EAP-SIM), most of these issues are already solved. Enabling another EAP method (such as PEAP) should not require far reaching changes in the architecture, since all future methods run within the existing EAP framework.

In the evaluation of the different EAP methods we did not make any measurements of connection establishment times and did not consider the hands-off capabilities of the different approaches. Such issues can be of importance if more than just one hotspot of the same operator is in the same

[3] Acer introduced the EeePC in late 2007. It is a small and relatively cheap laptop computer, a so-called netbook, intended to be primarily used for Internet browsing and emailing on the go. The EeePC was so successful that nowadays almost every manufacturer of mobile devices is offering a netbook.

location and customers walk out of the range of one hotspot into the range of the next one. Mechanisms for fast re-authentication or hand-off features should then also be evaluated.

In a next step, however, one should focus on the client software. The properties of such client software are essential for secure networking as well as for the usability of the whole solution. A first prototype should be built for a specific device or operating system (e.g. MS Windows XP) and the user experience should be tested and optimized. Once the client is running as expected, it should serve as a reference for possible porting to other platforms.

Another aspect that is important for PWLAN operators is roaming. Since most operators do not (yet) offer 802.1X EAP authentication, a solution has to be found for roaming customers connecting from a foreign hotspot. Again, this issue can be addressed with carefully designed client software, which is able to also authenticate using the WISPr XML Smart Client interface and possibly automatically establishing a VPN connection. While this is not as secure as 802.1X with EAP, it would allow customers to still use the existing hotspots all over the world.

Last but not least, current price plans are not appropriate for seamless authentication. As mentioned in the recommendations Section above, flat-rate pricing should be offered as soon as automated login mechanisms are available. This however is not a technical issue, but more an aspect of product management.

Bibliography

[1] Internet Draft published by the IETF., *PEAPv0 draft-kamath-pppext-peapv0-00.txt and PEAPv1 draft-josefsson-pppext-eap-tls-eap-10.txt.*

[2] Internet Architecture Board (IAB)., [Online] http://www.ietf.org/proceedings/06mar/IDs/draft-iab-auth-mech-05.txt.

[3] RSA Security., [Online] http://www.rsa.com/node.aspx?id=1156.

[4] Diffie, Whitfield and Hellman, Martin., *New Directions in Cryptography, IEEE Transactions on Information Theory, Vol. IT-22,6, pp. 644-654.* 1976.

[5] Gerardo I. Simari - Universidad Nacional del Sur, Argentina., *A Primer on Zero Knowledge Protocols.* 2002.

[6] Hannu A. Aronsson, Helsinki UT., Zero Knowledge Protocols and Small Systems. [Online] http://www.tml.tkk.fi/Opinnot/Tik-110.501/1995/zeroknowledge.html.

[7] S. M. Bellovin and M. Merritt., *Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks.* Oakland : Proceedings of the I.E.E.E. Symposium on Research in Security and Privacy, 1992.

[8] Madjid Nakhjiri and Mahsa Nakhjiri., *AAA and Network Security for Mobile Access.* s.l. : WILEY, 2005.

[9] Ron Rivest, Adi Shamir and Len Adleman., *A method for obtaining Digital Signatures and Public Key Cryptosystems.* ACM : Communications of the ACM. pages 120-126, 1978.

[10] VeriSign., *White Paper - Total Cost of Ownership for PKI.*

[11] International Organisation for Standardization (ISO)., *ISO/IEC 13335-1:2004 - Information technology - Security techniques.*

[12] —. *ISO 17799-2005 - Information technology - Security techniques.*

[13] N. Borisov, I. Goldberg und D. Wagner., *Intercepting mobile communications: the insecurity of 802.11.* Italy : International Conference on Mobile Computing and Networking, Proceedings of the 7th annual international conference, 2001.

[14] The Wi-Fi Alliance., http://www.wi-fi.org/. [Online]

[15] Internet Assigned Numbers Authority (IANA)., [Online] http://www.iana.org/.

[16] Krishna Sankar, Sri Sundaralingam, Darrin Miller, Andrew Balinsky., *Cisco Wireless LAN Security.* s.l. : Cisco Systems (Cisco Press), 2004.

[17] Microsoft TechNET., MS-CHAP v1. [Online] http://technet.microsoft.com/en-us/library/cc957985.aspx.

[18] Joshua Wright - ASLEAP, Exploiting Cisco LEAP., [Online] http://www.willhackforsushi.com/Asleap.html.

[19] The Internet Engineering Task Force (IETF)., *Internet Draft EAP-FAST - draft-cam-winget-eap-fast-06.txt.*

[20] EAP-TPM - IETF Internet Draft (Experimental, draft-latze-emu-eap-tpm-00)., [Online] http://tools.ietf.org/html/draft-latze-emu-eap-tpm-00.

[21] Carolin Latze., Towards a Zero Configuration Authentication Scheme for 802.11 Based Networks. [Online] http://diuf.unifr.ch/people/latzec/lcn2008.pdf.

[22] Trusted Computing Group., [Online] https://www.trustedcomputinggroup.org.

[23] EAP-TPM - Carolin Latze., Very First EAP-TPM Prototype. [Online] http://diuf.unifr.ch/people/latzec/prototyping/first/.

[24] The Internet Engineering Task Force (IETF)., AAA Working Group. [Online] http://www.ietf.org/html.charters/HISTORY/aaa-charter.2006-12-06.15.html.

[25] FON - The biggest WLAN-sharing community., [Online] http://www.fon.com.

[26] Wi-Fi Alliance., *WISPr Recommendation Document.* No longer avaliable - try Google.

[27] Trustive., http://corporate.trustive.com/research/. [Online]

[28] International Roaming Access Protocols (IRAP)., [Online] http://www.irap.nl/.

[29] European Telecommunications Standards Institute (ETSI)., [Online] http://www.etsi.org.

[30] ETSI TISPAN Project., [Online] http://www.etsi.org/tispan/.

[31] Black Hat - Briefings and Trainings., [Online] http://www.blackhat.com/.

[32] George Ou - ZD Net., Hamster plus Hotspot equals Web 2.0 meltdown! [Online] 2007. http://blogs.zdnet.com/Ou/?p=651.

[33] Sarah Granger., Social engineering reloaded. [Online] http://www.securityfocus.com/infocus/1860/2.

[34] Ethereal - Network Protocol Analyzer., http://ethereal.com/. [Online]

[35] Wireshark - The successor of Ethereal., http://www.wireshark.org/. [Online]

[36] AirSnarf - A rogue AP setup utility by The Shmoo Group., http://airsnarf.shmoo.com/. [Online]

[37] George Ou - ZD Net: "What exactly is this new DNS hack you keep hearing about?"., [Online] 5. 8 2004. http://blogs.zdnet.com/BTL/?p=280.

[38] Microsoft MSDN., Securing Public Wi-Fi Hotspots. [Online] http://msdn.microsoft.com/en-us/library/ms818947.aspx.

[39] George Ou - ZD Net., You use my hotspot, I'll use your credit card. [Online] http://blogs.zdnet.com/Ou/?p=30.

[40] Wi-Fi Planet - News Oct 2004., T-Mobile's Hotspots Are Secured. [Online] http://www.wi-fiplanet.com/news/article.php/3417281.

[41] Boingo - Global Wi-Fi Roaming., [Online] http://www.boingo.com/.

[42] Apple iPhone., [Online] http://www.apple.com/iphone/.

[43] AutoWi-Fi Lite., [Online] http://subzero.eu/autowifi/AutoWiFi/Welcome.html.

[44] Devicescape Software Inc., Easy Wi-Fi. [Online] http://www.devicescape.com/.

[45] ShmooCon - The Hacker Convention (2008)., [Online] http://www.shmoocon.org/2008/.

[46] Microsoft Corporation., "WPA2 Update for Windows XP (KB893357)."

[47] —. Wireless Provisioning Services (WPS) Authoring Tool. [Online] Microsoft Download Center.

[48] Mircosoft Native Wi-Fi API (Microsoft Developer Network)., [Online] http://msdn.microsoft.com/en-us/library/ms706556(VS.85).aspx.

[49] Apple., iPhone Configuration Utility. [Online]
http://www.apple.com/support/iphone/enterprise/.

[50] Apple iPhone Enterprise Deployment Guide (Fourth Edition)., [Online]
http://manuals.info.apple.com/en_US/Enterprise_Deployment_Guide.pdf.

[51] Whisher wifi.com Client., [Online] http://www.wifi.com.

Request for Comments

All [RFCXXXX] references can be accessed at *http://www.faqs.org/faqs/faqsearch.html*