Mechanisms for Security and Authentication of Wi-Fi devices

Gajraj Singh
Research Scholar, NIMS University,
Jaipur(Rajasthan)

Dr. P.K. Yadav,
Associate Professor, Govt. College
Kanwali(Rewari) Haryana

ABSTRACT
This paper presents an overview of the different approaches for Wi-Fi device security 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 concepts, protocols and standards in the context of wireless LAN security are presented. The paper concludes with an evaluation of the different approaches for hotspot authentication.

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

1. INTORDUCTION
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 claims 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.

1.2 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.

1.2.1 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.

1.2.2 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.

1.2.3 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 paper. Therefore, for the rest of the text, the term authentication refers to either user- or device authentication, or client authentication in general.

1.3 **Authentication Mechanisms**

Since the intention of this paper 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:

A. 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.

B. 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).

C. 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 attacks1 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.

D. 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.

E. 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.

F. 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 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).

G. 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.

1.4 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.

1.4.1 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.

1.4.2 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

Fig: 1.1 Three-party authentication model

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 million.

1.5 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 1.1, 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 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.

1.6 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.

1.6.1 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.

1.6.2 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 paper.

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. Let us now discuss the mechanisms behind digital signatures.

1.6.2.1 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:

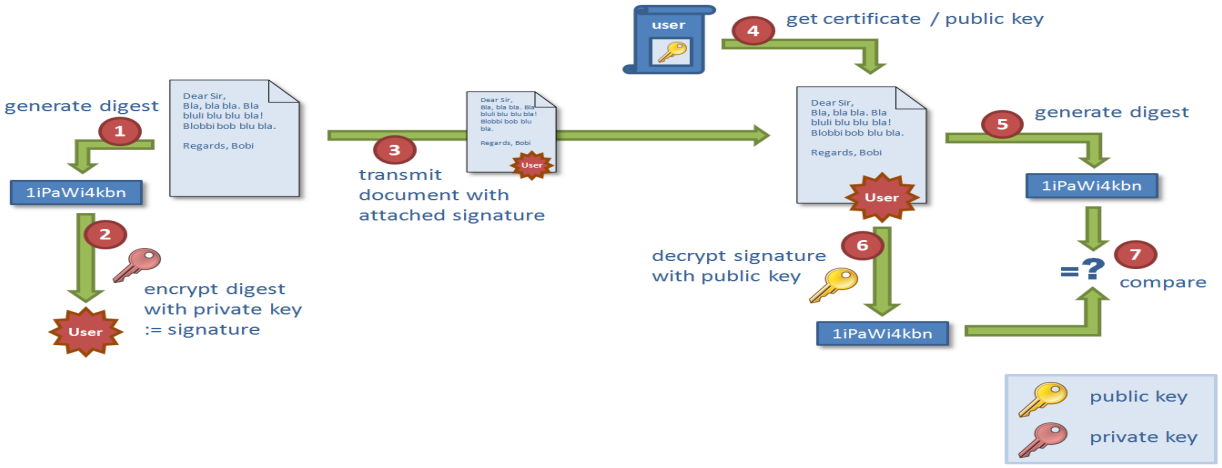

Fig 1.2: Digital Signature Scheme

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.

1.6.2.2 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.

1.6.3 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. Gajraj). 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:

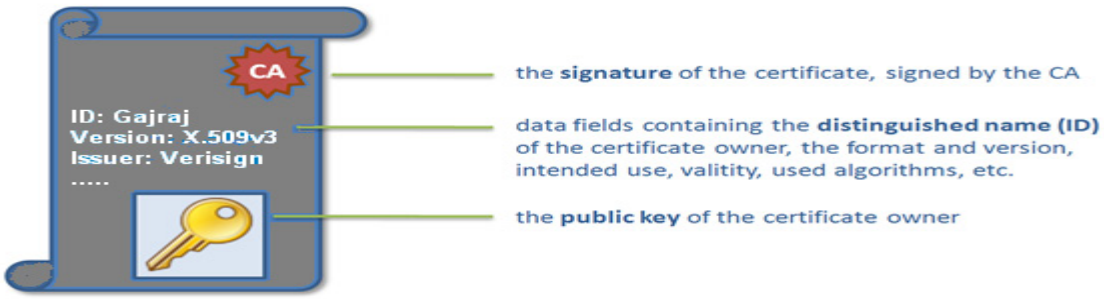

Fig. 1.3: Abstract view of a digital certificate

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.

1.6.4 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. 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.

3. CONCLUSION

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.

3.1 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.

3.2 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.

3.3 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.

3.4 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.

3.5 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.

4. REFERENCES

[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/Tik110.501/1995/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.