

Chapter 2. Security Infrastructure

Generally speaking, a security mechanism consists of protocols and algorithms to establish certain trust relationships in a communication system. However, a trust relationship cannot be established from the air. It relies on already existed trust relationships to establish the new trust. The already existed trust relationships are often used to provide services which are referred as *infrastructure support*. This chapter will introduce commonly employed infrastructure support for secure communication systems.

1 Infrastructure Support

As we mentioned before, infrastructure support is a set of services to establish trust relationships. If the service is to authenticate a user, then a successful authentication protocol execution implies an established trust on the user's identity. Based on such a trust, the user under the claimed identity can be authorized, for example, to access a network or certain information. If the service is to issue a certificate to a public key, then it is to establish a trust binding between a public key and its owner.

These services are provided based on a trust relationship, which may have been established through a business or organization relationship. For instance, an enterprise IT department may generate and distribute keys for the wireless devices used within the enterprise. On other hand, a service itself could be a business. For example, a certificate authority can be a commercial service provider employed by different applications to issue certificate.

A service can be an online service, that is, the infrastructure provides service in real time in each execution of the security mechanism. Some can also be an offline service, that is, the interaction with the infrastructure is not needed at the time a security mechanism is executed.

The security requirements and trust models are quite different for each service. In this chapter, since most of the security mechanisms have not been introduced formally, we may not be able to explain in details on how a security mechanism relies on a special service. But we feel that it is important to include a general introduction in the early stage so that the readers can see the role of infrastructure support in a secure communication system. We will look into the detailed interactions with each infrastructure support when the actual security mechanisms are introduced in the later chapters.

2 Authentication Server

An authentication server, as its name implies, is to provide authentication service. In order to explain the role of the authentication server, we will need to introduce entity authentication first.

⁰Copyright ©2008 L. Chen and G. Gong. All rights reserved. May be freely reproduced for educational or personal use.

2.1 Entity Authentication

In a communication system, entity authentication is a process for one party to assure another party's identity with whom it communicates. A communication party is a node, if it is a user device, together with its user. An entity authentication may be a user authentication or a device authentication. But in this section, an entity or a party is understood to be a user or a device. The authentication is to verify whether the specific authentication data is valid, which can be generated only by the party with the claimed identity. In other words, the party to be authenticated must provide some verifiable evidence to prove the entity is indeed identified by the identity as it claims. In some literature, the entity to be authenticated is called *claimant*.

For example, for a given identity like a user name, a password is widely used authentication evidence, which we probably have used every day. The password is generated by a user and stored in a place where it can be verified every time the authentication is conducted. However, it is obvious that the passwords can only provide very limited security for an entity authentication. Since human can only remember short passwords, they are vulnerable to exhausting search attacks. Furthermore, it is often the case that user passwords are not selected randomly such as the name of a pet or the title of a favorite book. As a result, the exhausting search can try the most likely passwords first to reduce the actual effort to succeed. Since entity authentication is often the step happening before key establishment in establishing trusted communications with a remote party, it may not be possible to protect the procedure of transmitting the password. A show stop limitation of the password based authentication in a communication system is that if the password is to be transmitted to a remote party, it is vulnerable to eavesdropping.

Nevertheless, password based authentication illustrates some basic ideas about entity authentication. In this section, we will introduce a cryptographic authentication method. This method is often called *challenge-response method*. Basically, the method can be either public key based or symmetric key based. We will further introduce this method in Chapter 4. Here we use symmetric key based challenge-response method to explain the idea.

With symmetric key based method, assume that both entities I and J share a key K_{IJ} . If party I is authenticated by party J , party J will generate a random string Ch , called a *challenge*, and send to party I . Upon receiving the string Ch , party I generates a response Res by computing

$$Res = MAC(K_{IJ}, Ch),$$

where MAC is a message authentication code as we introduced before. In order to verify the response, party J will calculate

$$Res' = MAC(K_{IJ}, Ch).$$

If $Res = Res'$, then party J can be assured that it is indeed party I , since no one else holds key K_{IJ} and can calculate a correct Res . Figure 1 illustrates an authentication by a challenge-response method.

We have noticed that in order to authenticate entity I , entity J must hold an authentication

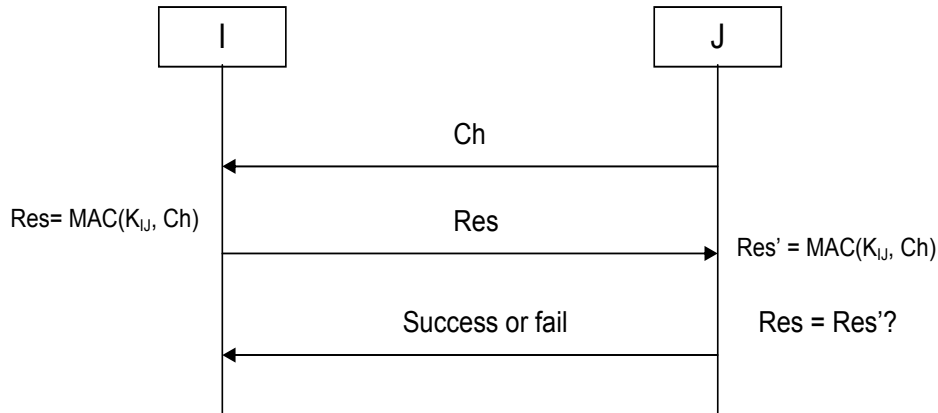


Figure 1: Entity Authentication by Challenge-Response

key and must have the capability to execute the cryptography operations. This leads to the need of an authentication server, which we will introduce in the next section.

2.2 Authentication Server

Entity authentication is often executed as an access authentication. That is, depending on the result of entity authentication, the entity can be authorized or rejected to access a network or certain information. For example, when a mobile phone is turned on, the first step is to register to the network for connections. The access permission depends on a subscriber authentication. If the authentication is successful, then the mobile phone gets connected. That is, it is authorized to access the cellular service.

In this section, the network entity which authorizes access is called a *network access server*. In the above example, a mobile phone, once authenticated, will be authorized to access cellular network. The entity in cellular network to authorize the access is considered as an access server. In IEEE 802.11 network, an access point plays the role of a network access server. Here the term server emphasizes the role in authorizing access for other entities. It does not indicate their processing capabilities nor physical security. As a matter of fact, it may not be physically safe to distribute the long-term keys used for authentication to these access servers when symmetric key based authentication methods are employed. Furthermore, since the access may be requested and authorized through any of these access servers, centralized database and authentication will help for billing and other purpose. Therefore, this leads to a demand of using a centralized server and database to execute authentication operations. As a result, an authentication server is often shared by many access servers. Not like an access server, an authentication server may be located remotely relative to the entity which requests for access. Figure 2 illustrates a situation that an entity I may request access from any of the access servers J_1, J_2, \dots, J_k , while each access server conducts authentication through the authentication server.

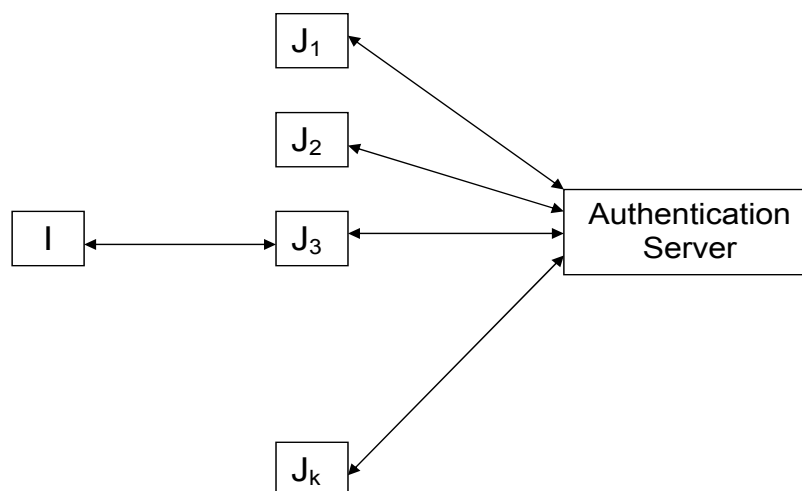


Figure 2: Authentication Server

The authentication server introduced in this section is a general concept. We will see how it works with each special authentication protocol in Chapter 4.

3 Certificate Authority

The invention of public key cryptography in 1976 makes “secret communications” possible without distributing “secret keys”. A public key, as its name implies, is public and will not need to be distributed “secretly”. In the very famous example with two parties Alice and Bob, Alice can send secret information to Bob by encrypting the information with Bob’s public key, which Bob can decrypt with his private key. Therefore, no secret key distribution is needed.

However, people quickly realized that the model with Alice and Bob can hardly be extended to a communication system with a large scale of users. The problem is how Alice can be assured that the key is Bob’s but not Eve’s public key. It may sound easy that Alice can get Bob’s public key through a conversation with Bob over the phone. But in a communication system, it is not realistic to assume the communication parties, even if they are human beings, know each other beforehand. Furthermore, such a key distribution method can hardly extend to a automated communication system.

The main problem with the above example of Alice and Bob is that, in a public key cryptography scheme, it is feasible for any party to generate a pair of mathematically valid public and private keys. As a result, any one, for example, Eve, can generate such a key pair to fool others by claiming that it is Bob’s public key. The information sent to Bob will be encrypted by the public key Eve generated, for which Eve knows the corresponding private key. As a result, instead of Bob, Eve can decrypt the information supposed to be accessed only by Bob. If this key pair is generated for digital signatures, then Eve can sign everything she wishes on the behalf of Bob.

Therefore, in order to provide the supposed security features with public key cryptography, it is crucial to bind a public key with its owner in a trusted way. For this reason, a trusted third party, called certificate authority (CA) is introduced to issue certificate.

3.1 Public Key Certificate

For a given public key and its owner, a certificate is a binding between the public key and the owner's identity. Digital signature is used to assure such a binding. The CA will sign the public key together with its owner's identity in order to generate a verifiable signature. Since a certificate must be verified by CA's public key, it seems we still have the original problem, that is, how it can be assured that it is CA's but not Eve's public key. Indeed, we have to make sure that a CA's public key is able to be recognized as authentic. Otherwise, it has no help by introducing a CA. By relying on a CA, it converts a peer-to-peer trust assumption to a multiple-to-one trust assumption. Intuitively, such a centralized trust is more manageable. It is easier to remember one phone number than a list of extensions. Practically, each party can obtain the CA's public key by downloading it from a trusted server. It can also install the CA's public key in the communication devices, like Internet routers and personal computers.

Besides the public key and the identity, a certificate may include some other attributes of the public key, for example, the expiration date, the scope of usage, etc..

3.2 Certificate Chain and Revocation

Introducing the certificate authority makes the trust relationship more scalable, especially when it is impossible to rely on a single certificate authority to establish all the peer-to-peer trust relationships. For example, a system may consist of multiple administrations. Each administration has its own certificate authority, CA_1 and CA_2 , respectively. Two entities, I and J , belonging to two different administrations, cannot establish trust unless two administrations have a commonly trusted third party. This party can issue certificates for CA_1 and CA_2 . Sometimes, it can have a hierarchical trust relationship with multiple levels of CAs. The highest CA is called a root CA. We show the two level hierarchy in Figure 3.

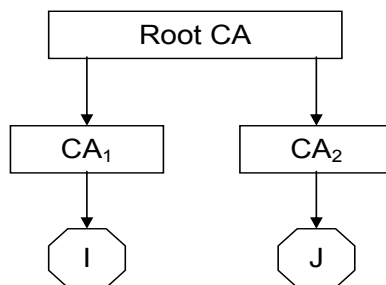


Figure 3: Certificate Hierarchy

In a certificate hierarchy, a higher level certificate authority will sign certificates for the public keys of the lower level certificate authorities. In the hierarchy shown in Figure 3, the root CA will sign certificates for public keys of CA₁ and CA₂, while CA₁ and CA₂ will sign certificates for public keys of entity *I* and entity *J* respectively.

If entity *I* needs to verify a public key certificate of the entity *J*, the verification will include the following two steps.

1. Verify the certificate of CA₂'s public key. That is, verify the signature of root CA with root CA's public key. If it is valid, then go to the next step.
2. Verify the certificate of *J*'s public key. That is, verify the signature of CA₂ with CA₂'s public key. If it is valid, then *J*'s public key is valid.

Notice that the entity *I* does not have to have any trust relationship directly with CA₂, who signs *J*'s public key certificate. As long as entity *I* trusts the root CA's public key, then it can verify whether CA₂'s public key is trustable. Based on the verified trust on CA₂'s public key, it can verify whether *J*'s public key is trustable. This trust relationship can be described as a trust chain. The corresponding cascade certificates form a *certificate chain*, which is illustrated in Figure 4. The arrow implies the action of using one public key to verify certificate of another public key as pointed.

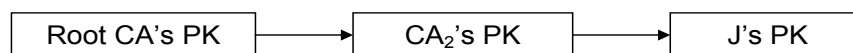


Figure 4: Certificate Chain

A certificate may have to be revoked before its expiration date. For certificate verification, it may have to verify whether it is included in a certificate revocation list (CRL). The certificate authority is the entity to maintain and update the CRL.

4 Key Generation and Distribution Server

Compared with the services provided by an authentication server or a certificate authority, the service a key generation server can provide must be restricted to the scenario that a secure key distribution is available. Therefore, we assume that a key generation server is also a key distribution server. In this section, we will see that key generation and distribution service are also a necessary infrastructure support in establishing protected communications.

4.1 Public/Private Key Pair Generation

When public key cryptography is introduced, the most attractive advantage as we have understood is that it does not need to have a secure channel to distribute the key since a public key is public

and no one besides the owner needs to know the private key. However, practically, it is often that a public/private key pair is generated by a third party, for example, a certificate authority. In this case, the private key will need to be transported to the owner in a protected manner.

It may sound a little disappointing and inconvenient. But generating a public and private key pair often involves complicated mathematics operations. For example, it may need to search for primes among integers in a required range or an elliptic curve over a finite field with required properties. These operations may be very challenging to a low power device.

Another reason to rely on a third party to generate key pairs is to assure that the generated key satisfies the designated security requirements for the devoted cryptographic operations. Some attacks discovered years ago are based on maliciously generated key pairs (see [1]). Whether employing a third party to generate key pairs is a decision made on multiple factors. However, if a third party is used, then it must facilitate a secure key distribution system.

4.2 Key Escrow

Despite the historical debating on key escrow, it has been a service to provide back up for encrypted contents. A key escrow server is often a key generator. Sometimes, a digital content provider will escrow the keys so that the protected digital products can be retrieved in case an eligible customer lost its key. However, a key generator may not be a key escrow server. For a key generator, it may not provide any back up service. Oppositely, it is required that the keys should be deleted from the server immediately after delivered to their owners for the obvious security reasons.

In the previous section, we discussed that a public key pair can be generated by either the owner or a key generator. However, some cryptography schemes have to rely on a trusted third party to generate a key pair for each entity, because a piece of common secret information is needed in generating every key pair. For example, identity based encryption (IBE) is such a scheme (see [2]). In order to embed an identity to a public key, it relies on a trusted third party to generate the corresponding private key. Otherwise, it cannot embed the identity to a public key.

Notice that when an identity is embedded in a public key, they are bound together. Therefore, certificate authority is not needed for IBE schemes. However, the IBE schemes need a key generator. In either case, it relies on a trusted third party even though for different services.

4.3 Symmetric Key Generation and Distribution

It is often that the strength of a symmetric key based cryptographic algorithm is measured by its key length. Since for an ideal cryptographic algorithm, the complexity of recovering the key should not be significantly lower than exhausting search. Therefore, the basic secure requirement for a symmetric key is its randomness. Obviously, it is not practical to generate a key by flipping a coin. A key should be generated by a qualified pseudorandom generator.

A symmetric key may be generated and shared among different parties through a public key method without relying on a key generation and distribution server. However, a system may have

only implemented symmetric key based cryptography mechanisms. In that case, without a key distribution server it is impossible to launch any security protection.

For symmetric keys, the key distribution may be accomplished in the manufacture by provisioning the keys to each device. It can also be distributed through service providers with a smart card. However, if the keys must be distributed to a remote party, then it requires a pre-established cryptographically and/or physically protected channel between the party and the key distribution server.

5 Signing Server

Since any one with the certified public key can verify a digital signature, it brings a great convenience in authorizing software installation and execution. Digital signature can also be used to protect copyrights on digital products. In these applications, the signing procedure may be conducted by a server, called a *signing server*. The signing server is usually controlled by the authorities to issue the rights. This section will introduce a general description on signing servers. In Chapter 7, we will see the dependencies on signing servers in establishing a trusted platform.

5.1 Signature for Authorized Software

Signing server is a kind of infrastructure support needed to build trusted platforms such that only authorized software can be executed on the platform. The software authorization is also used to make access control to certain data. For software authorization, a signing server may be managed by a device manufacture or by an IT department of an enterprise. For the software running in subscriber devices, like cellular phones, operators may manage a signing server.

The interface with a signing server is often well controlled. For example, in a manufacture, the software must pass standard tests and approved by an authority before sending to the signing server.

For a given platform, software packages may be signed by different servers. They must be verified based on a trust hierarchy. We will get this into the details in Chapter 7.

5.2 Signature for Copyrights

Today everything can be digitalized, from music to movies. Protection of digital copyrights has never been such challenging. Thank the invention of public key cryptography, digital signature provides a undisputable way to verify the digital copyrights.

The digital products can be signed by a signing server before being distributed to retailers. In Chapter 7, we will discuss how to enforce protection of digital copyrights in a trusted platform.

References

- [1] R. Anderson and R. Needham, *Robustness Principle for Public Key Protocols*, *Advances in Cryptology – Crypto '95*, pp. 236–247, Lecture Notes in Computer Science, Volume 963, Springer-Verlag, 1995.
- [2] D. Boneh and M. Franklin. *Identity-Based Encryption from Weil Pairing*. *Advances in Cryptology- Crypto 2001* pp. 213–229. Lecture Notes in Computer Science, Volume 2139, Springer-Verlag. 2001.