Unit-9
Security

9.1 INTRODUCTION TO SECURITY

- We start our description of security in distributed systems by taking a look at some general security issues.
- First, it is necessary to define what a secure system is. We distinguish security *policies* from security *mechanisms.* and take a look at the Globus wide-area system for which a security policy has been explicitly formulated.
- Our second concern is to consider some general design issues for secure systems.
- Finally, we briefly discuss some cryptographic algorithms, which play a key role in the design of security protocols.

- **Security** in a computer system is strongly related to the notion of dependability.
- Informally, a dependable computer system is one that we justifiably trust to deliver its services (Laprie, 1995).
- As mentioned in Chap. 7, dependability includes availability, reliability, safety, and maintainability.
- However, if we are to put our trust in a computer system, then confidentiality and integrity should also be taken into account.
- **Confidentiality** refers to the property of a computer system whereby its information is disclosed only to authorized parties.
- **Integrity** is the characteristic that alterations to a system's assets can be made only in an authorized way.
- In other words, improper alterations in a secure computer system should be detectable and recoverable.
- Major assets of any computer system are its hardware, software, and data.

- Another way of looking at security in computer systems is that we attempt to protect the services and data it offers against security threats.
- There are four types of security threats to consider (Pfleeger, 2003):

1. Interception
2. Interruption
3. Modification
4. Fabrication

- The concept of **interception** refers to the situation that an unauthorized party has gained access to a service or data.
- A typical example of interception is where communication between two parties has been overheard by someone else.
- Interception also happens when data are illegally copied, for example, after breaking into a person's private directory in a file system.
- An example of interruption is when a file is corrupted or lost.
- More generally **interruption** refers to the situation in which services or data become unavailable, unusable, destroyed, and so on.
- In this sense, denial of service attacks by which someone maliciously attempts to make a service inaccessible to other parties is a security threat that classifies as interruption.

- **Modifications** involve unauthorized changing of data or tampering with a service so that it no longer adheres to its original specifications.
- Examples of modifications include intercepting and subsequently changing transmitted data, tampering with database entries, and changing a program so that it secretly logs the activities of its user.

Fabrication refers to the situation in which additional data or activity are generated that would normally not exist.

- For example, an intruder may attempt to add an entry into a password file or database.
- Likewise, it is sometimes possible to break into a system by replaying previously sent messages

Important security mechanisms are:

1. Encryption
2. Authentication
3. Authorization
4. Auditing

- **Encryption** is fundamental to computer security.
- Encryption transforms data into something an attacker cannot understand.
- In other words, encryption provides a means to implement data confidentiality.
- In addition, encryption allows us to check whether data have been modified.
- It thus also provides support for integrity checks.
- Authentication is used to verify the claimed identity of a user, client, server, host, or other entity.
- In the case of clients, the basic premise is that before a service starts to perform any work on behalf of a client, the service must learn the client's identity (unless the service is available to all).
- Typically, users are authenticated by means of passwords, but there are many other ways to authenticate clients.

- After a client has been authenticated, it is necessary to check whether that client is authorized to perform the action requested.
- Access to records in a medical database is a typical example.
- Depending on who accesses the database. Permission may be granted to read records, to modify certain fields in a record, or to add or remove a record.
- Auditing tools are used to trace which clients accessed what, and which way.
- Although auditing does not really provide any protection against security threats, audit logs can be extremely useful for the analysis of a security breach, and subsequently taking measures against intruders.
- For this reason, attackers are generally keen not to leave any traces that could eventually lead to exposing their identity.
- In this sense, logging accesses makes attacking sometimes a riskier business.

9.2 SECURE CHANNELS

- In the preceding chapters. we have frequently used the client-server model as a convenient way to organize a distributed system.
- In this model, servers may possibly be distributed and replicated, but also act as clients with respect to other servers.
- When considering security in distributed systems, it is once again useful to think in terms of clients and servers.
- In particular, making a distributed system secure essentially boils down to two predominant issues.
- The first issue is how to make the communication between clients and servers secure. Secure communication requires authentication of the communicating parties. In many cases it also requires ensuring message integrity and possibly confidentiality as well.
- As part of this problem, we also need to consider protecting the communication within a group of servers.

- The second issue is that of authorization: once a server has accepted a request from a client, how can it find out whether that client is authorized to have that request carried out?
- Authorization is related to the problem of controlling access to resources, which we discuss extensively in the next section.
- In this section, we concentrate on protecting the communication within a distributed system.
- The issue of protecting communication between clients and servers, can be thought of in terms of setting up a secure **channel** between communicating parties (Voydock and Kent, 1983).
- A secure channel protects senders and receivers against interception, modification, and fabrication of messages.
- It does not also necessarily protect against interruption.

- Protecting messages against interception is done by ensuring confidentiality: the secure channel ensures that its messages cannot be eavesdropped by intruders.
- Protecting against modification and fabrication by intruders is done through protocols for mutual authentication and message integrity.
- In the following pages, we first discuss various protocols that can be used for authentication, using symmetric as well as public-key cryptosystems.
- A detailed description of the logics underlying authentication can be found in Lampson et al. (1992). We discuss confidentiality and message integrity separately.

Cryptography

- Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it. The prefix "crypt-" means "hidden" or "vault" -- and the suffix "-graphy" stands for "writing.“
- In computer science, cryptography refers to secure information and communication techniques derived from mathematical concepts and a set of rule-based calculations called algorithms, to transform messages in ways that are hard to decipher.
- These deterministic algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on the internet, and confidential communications such as credit card transactions and email.

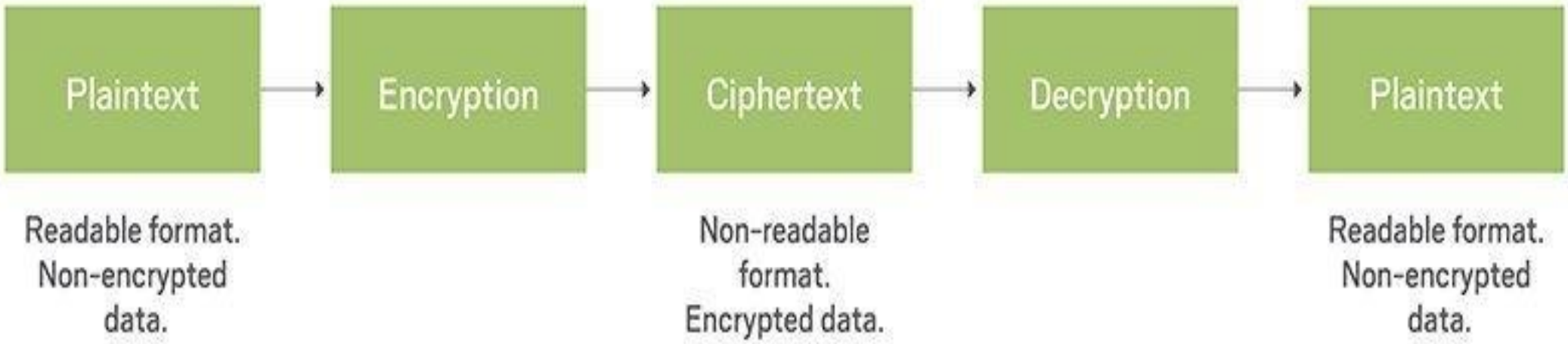
- Cryptography is closely related to the disciplines of cryptology and cryptanalysis.
- It includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit.
- However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext(ordinary text, sometimes referred to as cleartext) into ciphertext (a process called encryption), then back again (known as decryption).
- Individuals who practice this field are known as cryptographers.

- Modern cryptography concerns itself with the following four objectives:

1. **Confidentiality**: the information cannot be understood by anyone for whom it was unintended
2. **Integrity:** the information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected
3. **Non-repudiation**: the creator/sender of the information cannot deny at a later stage his or her intentions in the creation or transmission of the information
4. **Authentication**: the sender and receiver can confirm each other's identity and the origin/destination of the information

- While **cryptography** is the science of securing data, ***cryptanalysis*** is the science of analyzing and breaking secure communication.
- Classical **cryptanalysis** involves an interesting combination of analytical reasoning, application of mathematical tools, pattern finding, patience, determination, and luck.
- **Cryptanalysts** are also called ***attackers***.
- ***Cryptology*** embraces both **cryptography** and **cryptanalysis**.(https://youtu.be/5jpgMXt1Z9Y)

Cryptography

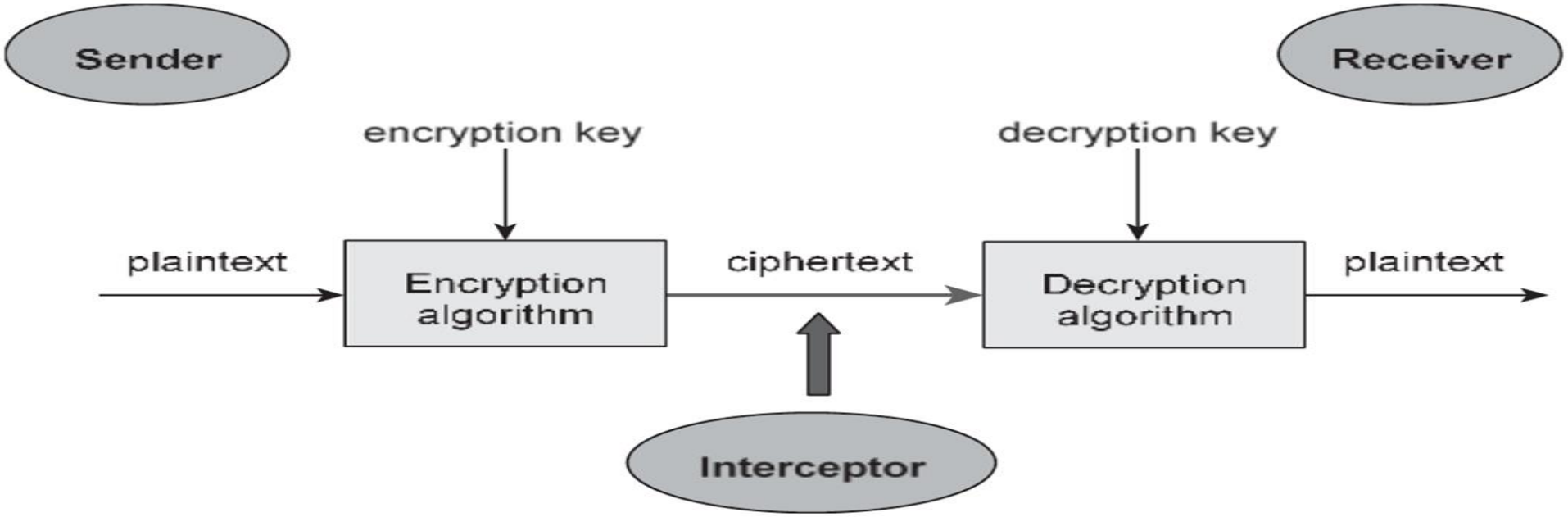


Components of a Cryptosystem

The various components of a basic cryptosystem are as follows:

- **Plaintext**. It is the data to be protected during transmission.
- **Encryption Algorithm**. It is a mathematical process that produces a ciphertext for any given plaintext and encryption key. It is a cryptographic algorithm that takes plaintext and an encryption key as input and produces a ciphertext.
- **Ciphertext**. It is the scrambled version of the plaintext produced by the encryption algorithm using a specific the encryption key. The ciphertext is not guarded. It flows on public channel. It can be intercepted or compromised by anyone who has access to the communication channel.
- **Decryption Algorithm**, It is a mathematical process, that produces a unique plaintext for any given ciphertext and decryption key. It is a cryptographic algorithm that takes a ciphertext and a decryption key as input, and outputs a plaintext. The decryption algorithm essentially reverses the encryption algorithm and is thus closely related to it.
- **Encryption Key**. It is a value that is known to the sender. The sender inputs the encryption key into the encryption algorithm along with the plaintext in order to compute the ciphertext.
- **Decryption Key**. It is a value that is known to the receiver. The decryption key is related to the encryption key, but is not always identical to it. The receiver inputs the decryption key into the decryption algorithm along with the ciphertext in order to compute the plaintext.

- An **interceptor** (an **attacker**) is an unauthorized entity who attempts to determine the plaintext.
- S/He can see the ciphertext and may know the decryption algorithm.
- S/He, however, must never know the decryption key.

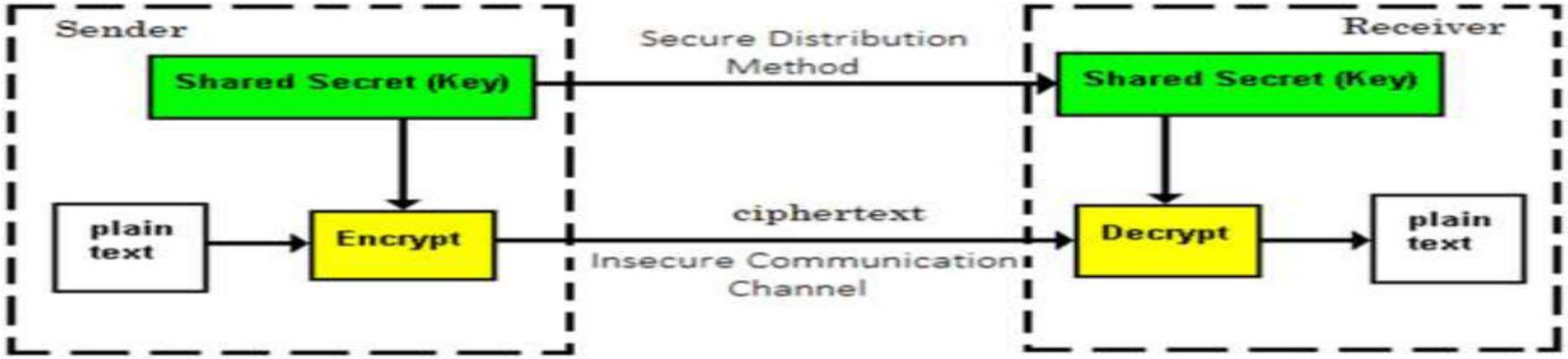


Types of Cryptosystems

- Fundamentally, there are two types of cryptosystems based on the manner in which encryption-decryption is carried out in the system:
  - Symmetric Key Encryption
  - Asymmetric Key Encryption
- The main difference between these cryptosystems is the relationship between the encryption and the decryption key. Logically, in any cryptosystem, both the keys are closely associated. It is practically impossible to decrypt the ciphertext with the key that is unrelated to the encryption key.

Symmetric Key Encryption / Secret key Cryptography

- The encryption process where **same keys are used for encrypting and decrypting** the information is known as Symmetric Key Encryption.
- The study of symmetric cryptosystems is referred to as **symmetric cryptography**. Symmetric cryptosystems are also sometimes referred to as **secret key cryptosystems**.
- A few well-known examples of symmetric key encryption methods are: Digital Encryption Standard (DES), Triple-DES (3DES), IDEA, and BLOWFISH



The salient features of cryptosystem based on symmetric key encryption are:

- Persons using symmetric key encryption must share a common key prior to exchange of information.
- Keys are recommended to be changed regularly to prevent any attack on the system.
- A robust mechanism needs to exist to exchange the key between the communicating parties. As keys are required to be changed regularly, this mechanism becomes expensive and cumbersome.
- In a group of **n** people, to enable two-party communication between any two persons, the number of keys required for group is **n × (n – 1)/2**.
- Length of Key (number of bits) in this encryption is smaller and hence, process of encryption-decryption is faster than asymmetric key encryption.
- Processing power of computer system required to run symmetric algorithm is less.

Challenge of Symmetric Key Cryptosystem

- There are two restrictive challenges of employing symmetric key cryptography.
  - **Key establishment** – Before any communication, both the sender and the receiver need to agree on a secret symmetric key. It requires a secure key establishment mechanism in place.
  - **Trust Issue** – Since the sender and the receiver use the same symmetric key, there is an implicit requirement that the sender and the receiver 'trust' each other. For example, it may happen that the receiver has lost the key to an attacker and the sender is not informed.
- These two challenges are highly restraining for modern day communication. Today, people need to exchange information with non-familiar and non-trusted parties. For example, a communication between online seller and customer. These limitations of symmetric key encryption gave rise to asymmetric key encryption schemes

Asymmetric Key Encryption / Public key Cryptography

- The encryption process where **different keys are used for encrypting and decrypting the information** is known as Asymmetric Key Encryption.
- Though the keys are different, they are mathematically related and hence, retrieving the plaintext by decrypting cipher text is feasible.
- Asymmetric Key Encryption was invented in the 20th century to come over the necessity of pre-shared secret key between communicating persons.

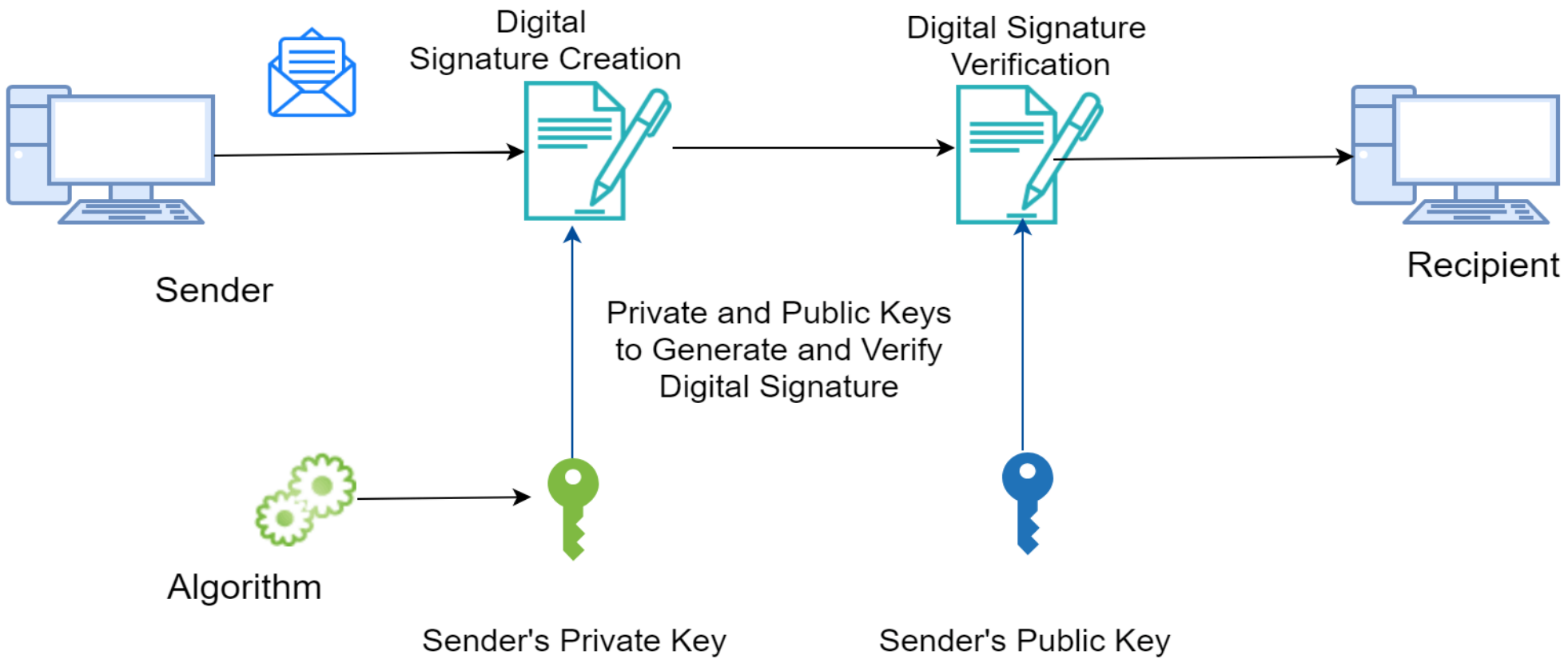
The salient features of this encryption scheme are as follows:

- Every user in this system needs to have a pair of dissimilar keys, **private key** and **public key**. These keys are mathematically related – when one key is used for encryption, the other can decrypt the ciphertext back to the original plaintext.
- It requires to put the public key in public repository and the private key as a well guarded secret. Hence, this scheme of encryption is also called **Public Key Encryption**.
- Though public and private keys of the user are related, it is computationally not feasible to find one from another. This is a strength of this scheme.
- When *Host1* needs to send data to *Host2*, he obtains the public key of *Host2* from repository, encrypts the data, and transmits.
- *Host2* uses his private key to extract the plaintext.
- Length of Keys (number of bits) in this encryption is large and hence, the process of encryption-decryption is slower than symmetric key encryption.
- Processing power of computer system required to run asymmetric algorithm is higher.

Digital Signature

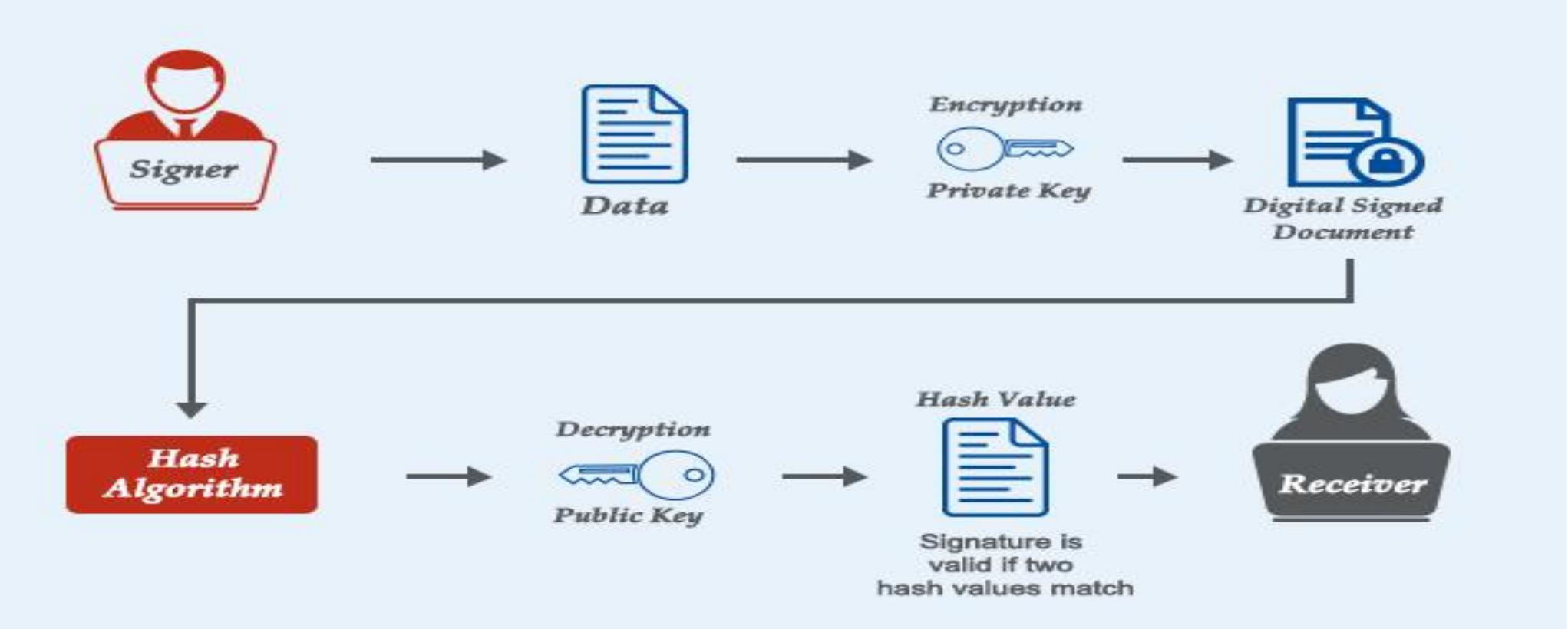
- A digital signature—a type of electronic signature—is a mathematical algorithm routinely used to validate the authenticity and integrity of a message (e.g., an email, a credit card transaction, or a digital document).
- Digital signatures create a virtual fingerprint that is unique to a person or entity and are used to identify users and protect information in digital messages or documents.
- In emails, the email content itself becomes part of the digital signature.
- Digital signatures are significantly more secure than other forms of electronic signatures.
- Digital signatures increase the transparency of online interactions and develop trust between customers, business partners, and vendors.(https://youtu.be/TmA2QWSLSPg)

Digital Signatures



How do digital signatures work?

- **Hash function** – A hash function (also called a “hash”) is a fixed-length string of numbers and letters generated from a mathematical algorithm and an arbitrarily sized file such as an email, document, picture, or other type of data.
- This generated string is unique to the file being hashed and is a one-way function— a computed hash cannot be reversed to find other files that may generate the same hash value.
- Some of the more popular hashing algorithms in use today are Secure Hash Algorithm-1 (SHA-1), the Secure Hashing Algorithm-2 family (SHA-2 and SHA-256), and Message Digest 5 (MD5).


Signer
Data
Encryption
Private Key
Digital Signed Document
Hash Algorithm
Decryption
Public Key
Hash Value
Signature is valid if two hash values match
Receiver

Public key cryptography

- Public key cryptography (also known as asymmetric encryption) is a cryptographic method that uses a key pair system.
- One key, called the public key, encrypts the data. The other key, called the private key, decrypts the data.
- Public key cryptography can be used several ways to ensure confidentiality, integrity, and authenticity.
- Public key cryptography can ensure integrity by creating a digital signature of the message using the sender's private key.
- This is done by hashing the message and encrypting the hash value with their private key. By doing this, any changes to the message will result in a different hash value.
- Ensure confidentiality by encrypting the entire message with the recipient's public key. This means that only the recipient, who is in possession of the corresponding private key, can read the message.
- Verify the user's identity using the public key and checking it against a certificate authority.

Digital certificates

- One issue with public key cryptosystems is that users must be constantly vigilant to ensure that they are encrypting to the correct person's key.
- In an environment where it is safe to freely exchange keys via public servers, *man-in-the-middle* attacks are a potential threat.
- In this type of attack, someone posts a phony key with the name and user ID of the user's intended recipient. Data encrypted to, and intercepted by, the true owner of this bogus key is now in the wrong hands.
- In a public key environment, it is vital that you are assured that the public key to which you are encrypting data is in fact the public key of the intended recipient and not a forgery.
- You could simply encrypt only to those keys which have been physically handed to you. But suppose you need to exchange information with people you have never met; how can you tell that you have the correct key?
- ***Digital certificates*** simplify the task of establishing whether a public key truly belongs to the purported owner.

- A digital certificate is data that functions much like a physical certificate.
- A digital certificate is information included with a person's public key that helps others verify that a key is genuine or *valid*.

A digital certificate consists of three things:

• A public key.
• Certificate information ("Identity" information about the user, such as name, user ID, and so on.)
• One or more digital signatures.

The purpose of the digital signature on a certificate is to state that the certificate information has been attested to by some other person or entity. The digital signature does not attest to the authenticity of the certificate as a whole; it vouches only that the signed identity information goes along with, or *is bound to*, the public key.

Thus, a certificate is basically a public key with one or two forms of ID attached, plus a hearty stamp of approval from some other trusted individual. (https://youtu.be/UbMlPIgzTxc)

9.3 ACCESS CONTROL

- In the client-server model, which we have used so far, once a client and a server have set up a secure channel, the client can issue requests that are to be carried out by the server.
- Requests involve carrying out operations on resources that are controlled by the server. A general situation is that of an object server that has a number of objects under its control.
- A request from a client generally involves invoking a method of a specific object.
- Such a request can be carried out only if the client has sufficient access rights for that invocation.

- Formally, verifying access rights is referred to as access control, whereas authorization is about granting access rights.
- The two terms are strongly related to each other and are often used in an interchangeable way. There are many ways to achieve access control.
- We start with discussing some of the general issues, concentrating on different models for handling access control.
- One important way of actually controlling access to resources is to build a firewall that protects applications or even an entire network.
- Firewalls are discussed separately.
- With the advent of code mobility, access control could no longer be done using only the traditional methods.

9.3.1 General Issues in Access Control

- In order to understand the various issues involved in access control, the simple model shown in Fig. 9-25 is generally adopted.
- It consists of subjects that issue a request to access an object. An object is very much like the objects we have been discussing so far.
- It can be thought of as encapsulating its own state and implementing the operations on that state.
- The operations of an object that subjects can request to be carried out are made available through interfaces. Subjects can best be thought of as being processes acting on behalf of users, but can also be objects that need the services of other objects in order to carry out their work

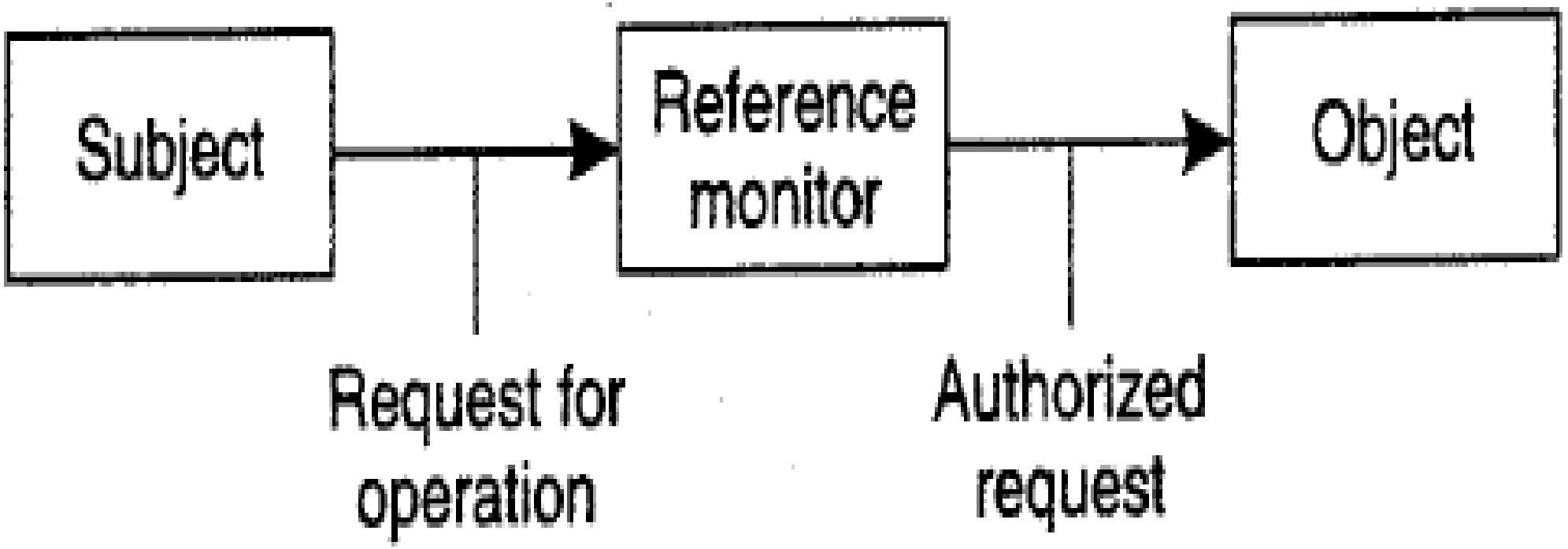


Figure 9-25. General model of controlling access to objects.

- Controlling the access to an object is all about protecting the object against invocations by subjects that are not allowed to have specific (or even any) of the methods carried out.
- Also, protection may include object management issues, such as creating, renaming, or deleting objects.
- Protection is often enforced by a program called a reference monitor.
- A reference monitor records which subject may do what, and decides whether a subject is allowed to have a specific operation carried out.
- This monitor is called (e.g., by the underlying trusted operating system) each time an object is invoked.
- Consequently, it is extremely important that the reference monitor is itself tamperproof: an attacker must not be able to fool around with it.

Access Control Matrix

- A common approach to modeling the access rights of subjects with respect to objects is to construct an access control matrix.
- Each subject is represented by a row in this matrix; each object is represented by a column.
- If the matrix is denoted *M,* then an entry *M [s,o]* lists precisely which operations subject *s* can request to be carried out on object *o.*
- In other words, whenever a subject *s* requests the invocation of method *m* of object *0,* the reference monitor should check whether *m* is listed in *M [s,o].* If *m* is not listed in 1\1*[s,o],* the invocation fails.

- Considering that a system may easily need to support thousands of users and millions of objects that require protection, implementing an access control matrix as a true matrix is not the way to go.
- Many entries in the matrix will be empty: a single subject will generally have access to relatively few objects.
- Therefore, other, more efficient ways are followed to implement an access control matrix.
- One widely-applied approach is to have each object maintain a list of the access rights of subjects that want to access the object.
- In essence, this means that the matrix is distributed column-wise across all objects, and that empty entries are left out.
- This type of implementation leads to what is called an Access Control List(ACL). Each object is assumed to have its own associated ACL.

- Another approach is to distribute the matrix row-wise by giving each subject a list of capabilities it has for each object. In other words, a capability corresponds to an entry in the access control matrix.
- Not having a capability for a specific object means that the subject has no access rights for that object.
- A capability can be compared to a ticket: its holder is given certain rights that are associated with that ticket. It is also clear that a ticket should be protected against modifications by its holder.
- One approach that is particularly suited in distributed systems and which has been applied extensively in Amoeba (Tanenbaum et aI., 1990), is to protect (a list of) capabilities with a signature.
- We return to these and other matters later when discussing security management.

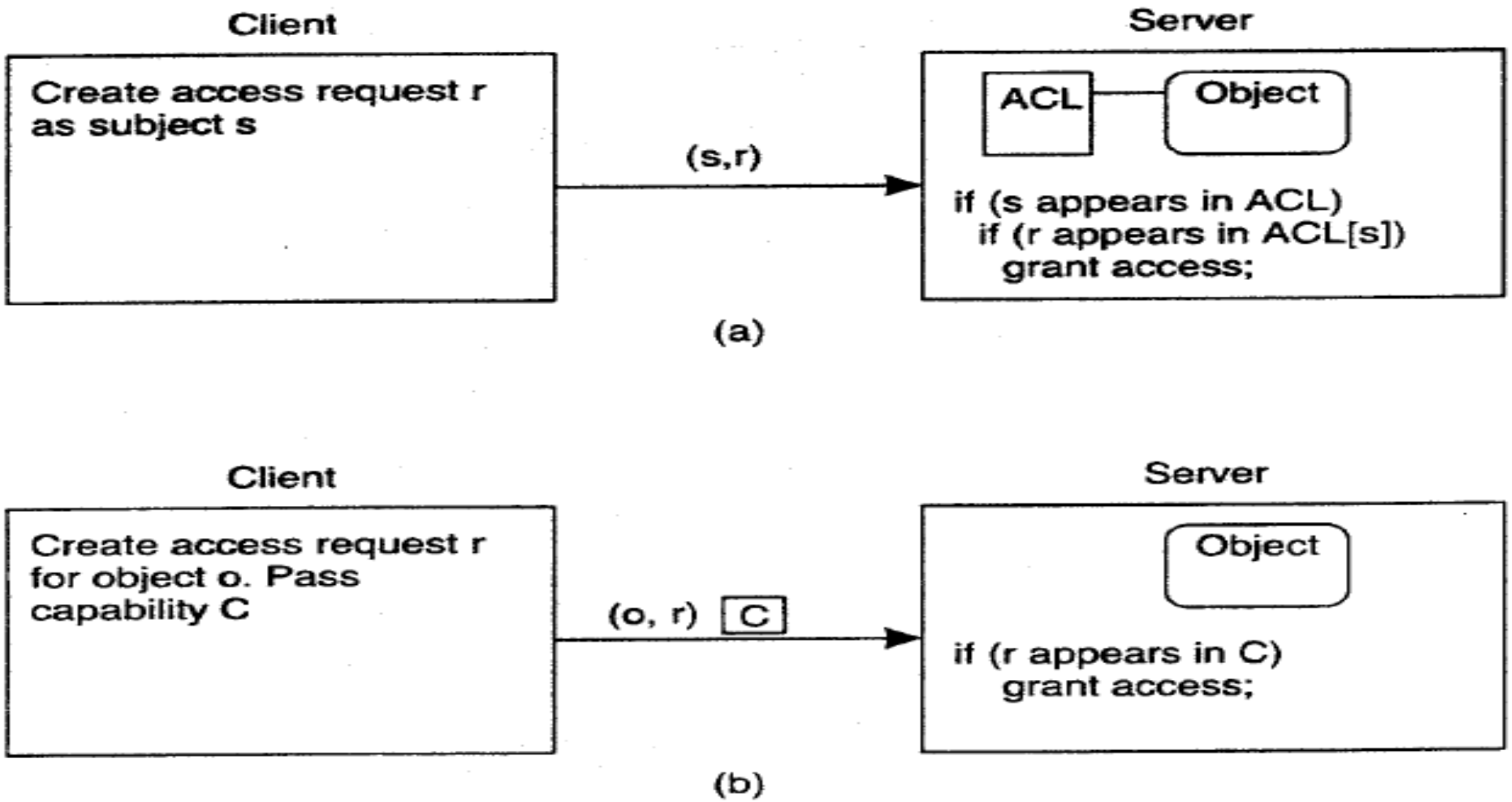


Figure 9-26. Comparison between ACLs and capabilities for protecting objects. (a) Using an ACL. (b) Using capabilities.

- The difference between how ACLs and capabilities are used to protect the access to an object is shown in Fig. 9-26.
- Using ACLs, when a client sends a request to a server, the server's reference monitor will check whether it knows the client and if that client is known and allowed to have the requested operation carried out, as shown in Fig. 9-26(a).
- However, when using capabilities, a client simply sends its request to the server. The server is not interested in whether it knows the client; the capability says enough.
- Therefore, the server peed only check whether the capability is valid and whether the requested operation is listed in the capability. This approach to protecting objects by means of capabilities is shown in Fig. 9-26(b).

9.3.2 Firewalls

- So far, we have shown how protection can be established using cryptographic techniques, combined with some implementation of an access control matrix.
- These approaches work fine as long as all communicating parties play according to the same set of rules.
- Such rules may be enforced when developing a standalone distributed system that is isolated from the rest of the world.
- However, matters become more complicated when outsiders are allowed to access the resources controlled by a distributed system.
- Examples of such accesses including sending mail, downloading files, uploading tax forms, and so on.

- To protect resources under these circumstances, a different approach is needed.
- In practice, what happens is that external access to any part of a distributed system is controlled by a special kind of reference monitor known as a firewall (Cheswick and Bellovin, 2000; and Zwicky et aI., 2000).
- Essentially, a firewall - disconnects any part of a distributed system from the outside world, as shown in
- Fig. 9-28. All outgoing, but especially all incoming packets are routed through a special computer and inspected before they are passed.
- Unauthorized traffic is discarded and not allowed to continue.
- An important issue is that the firewall itself should be heavily protected against any kind of security threat: it should never fail.

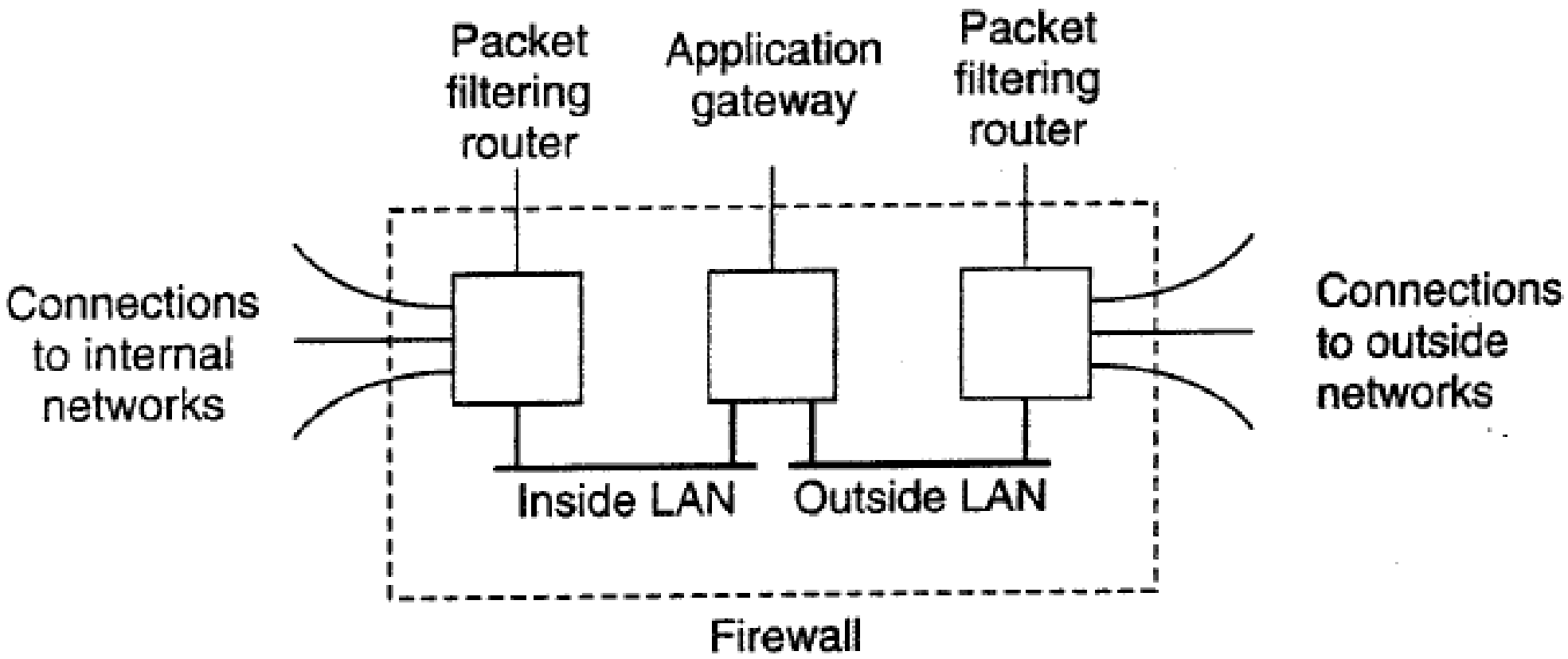


Figure 9-28. A common implementation of a firewall.

- Firewalls essentially come in two different flavors that are often combined.
- An important type of firewall is a **packet-filtering gateway**.
- This type of firewall operates as a router and makes decisions as to whether or not to pass a network packet based on the source and destination address as contained in the packet's header.
- Typically, the packet-filtering gateway shown on the outside LAN in Fig. 9-28 would protect against incoming packets, whereas the one on the inside LAN would filter outgoing packets.
- For example, to protect an internal Web server against requests from hosts that are not on the internal network, a packet-filtering gateway could decide to drop all incoming packets addressed to the Web server.

- The other type of firewall is an **application-level gateway**.
- In contrast to a packet-filtering gateway, which inspects only the header of network packets, this type of firewall actually inspects the content of an incoming or outgoing message.
- A typical example is a mail gateway that discards incoming or outgoing mail exceeding a certain size.
- More sophisticated mail gateways exist that are, for example, capable of filtering spam e-mail.
- Another example of an application-level gateway is one that allows external access to a digital library server, but will supply only abstracts of documents.
- If an external user wants more, an electronic payment protocol is started. Users inside the firewall have direct access to the library service.

- A special kind of application-level gateway is what is known as a **proxy gateway.**
- This type of firewall works as a front end to a specific kind of application, and ensures that only those messages are passed that meet certain criteria. Consider, for example, surfing the Web.
- As we discuss in the next section, many Web pages contain scripts or applets that are to be executed in a user's browser.
- To prevent such code to be downloaded to the inside LAN, all Web traffic could be directed through a Web proxy gateway.
- This gateway accepts regular HTTP requests, either from inside or outside the firewall.
- In other words, it appears to its users as a normal Web server.
- However, it filters all incoming and outgoing traffic, either by discarding certain requests and pages, or modifying pages when they contain executable code.

9.3.3 Secure Mobile Code

- As we discussed in Chap. 3, an important development in modem distributed systems is the ability to migrate code between hosts instead of just migrating passive data.
- However, mobile code introduces a number of serious security threats.
- For one thing, when sending an agent across the Internet, its owner will want to protect it against malicious hosts that try to steal or modify information carried by the agent.
- Another issue is that hosts need to be protected against malicious agents. Most users of distributed systems will not be experts in systems technology and have no way of telling whether the program they are fetching from another host can be trusted not to corrupt their computer.
- In many cases it may be difficult even for an expert to detect that a program is actually being downloaded at all.

- Unless security measures are taken, once a malicious program has settled itself in a computer, it can easily corrupt its host.
- We are faced with an access control problem: the program should not be allowed unauthorized access to the host's resources.
- As we shall see protecting a host against downloaded malicious programs is not always easy.
- The problem is not so much as to avoid downloading of programs.
- Instead, what we are looking for is supporting mobile code that we can allow access to local resources in a flexible, yet fully controlled manner.

Protecting an Agent

- Before we take a look at protecting a computer system against downloaded malicious code, let us first take a look at the opposite situation.
- Consider a mobile agent that is roaming a distributed system on behalf of a user. Such an agent may be searching for the cheapest airplane ticket from Nairobi to Malindi, and has been authorized by its owner to make a reservation as soon as it found a flight.
- For this purpose, the agent may carry an electronic credit card.
- Obviously, we need protection here. Whenever the agent moves to a host, that host should not be allowed to steal the agent's credit card information.

- Also, the agent should be protected against modifications that make the owner pay much more than actually is needed.
- For example, if Chuck's Cheaper Charters can see that the agent has not yet visited its cheaper competitor Alice Airlines, Chuck should be prevented from changing the agent so that it will not visit Alice Airlines‘ host.
- Other examples that require protection of an agent against attacks from a hostile host include maliciously destroying an agent, or tampering with an agent such that it will attack or steal from its owner when it returns.

Protecting the Target

- Although protecting mobile code against a malicious host is important, more attention has been paid to protecting hosts against malicious mobile code.
- If sending an agent into the outside world is considered too dangerous, a user will generally have alternatives to get the job done for which the agent was intended.
- However, there are often no alternatives to letting an agent into your system, other than locking it out completely.
- Therefore, if it is once decided that the agent can come in, the user needs full control over what the agent can do.

9.3.4 Denial of Service

- Access control is generally about carefully ensuring that resources are accessed only by authorized processes.
- A particularly annoying type of attack that is related to access control is maliciously preventing authorized processes from accessing resources.
- Defenses against such denial-of-service (DoS) attacks are becoming increasingly important as distributed systems are opened up through the Internet.
- Where DoS attacks that come from one or a few sources can often be handled quite effectively, matters become much more difficult when having to deal with distributed denial of service (DDoS).

- In DDoS attacks, a huge collection of processes jointly attempt to bring down a networked service.
- In these cases, we often see that the attackers have succeeded in hijacking a large group of machines which unknowingly participate in the attack.
- Specht and Lee (2004) distinguish two types of attacks: those aimed at bandwidth depletion and those aimed at resource depletion.

9.4 SECURITY MANAGEMENT

- So far, we have considered secure channels and access control, but have hardly touched upon the issue how, for example, keys are obtained.
- In this section, we take a closer look at security management.
- In particular, we distinguish three different subjects. First, we need to consider the general management of cryptographic keys, and especially the means by which public keys are distributed.
- As it turns out, certificates play an important role here.
- Second, we discuss the problem of securely managing a group of servers by concentrating on the problem of adding a new group member that is trusted by the current members.
- Clearly, in the face of distributed and replicated services, it is important that security is not compromised by admitting a malicious process to a group.

- Third, we pay attention to authorization management by looking at capabilities and what are known as attribute certificates.
- An important issue in distributed systems with respect to authorization management is that one process can delegate some or all of its access rights to another process.
- Delegating rights in a secure way has its own subtleties as we also discuss in this section.

9.4.1 Key Management

- So far, we have described various cryptographic protocols in which we (implicitly) assumed that various keys were readily available.
- For example, in the case of public-key cryptosystems, we assumed that a sender of a message had the public key of the receiver at its disposal so that it could encrypt the message to ensure confidentiality.
- Likewise, in the case of authentication using a key distribution center (KDC), we assumed each party already shared a secret key with the KDC.
- However, establishing and distributing keys is not a trivial matter.
- Also, mechanisms are needed to revoke keys, that is, prevent a key from being used after it has been compromised or invalidated.
- For example, revocation is necessary when a key has been compromised.

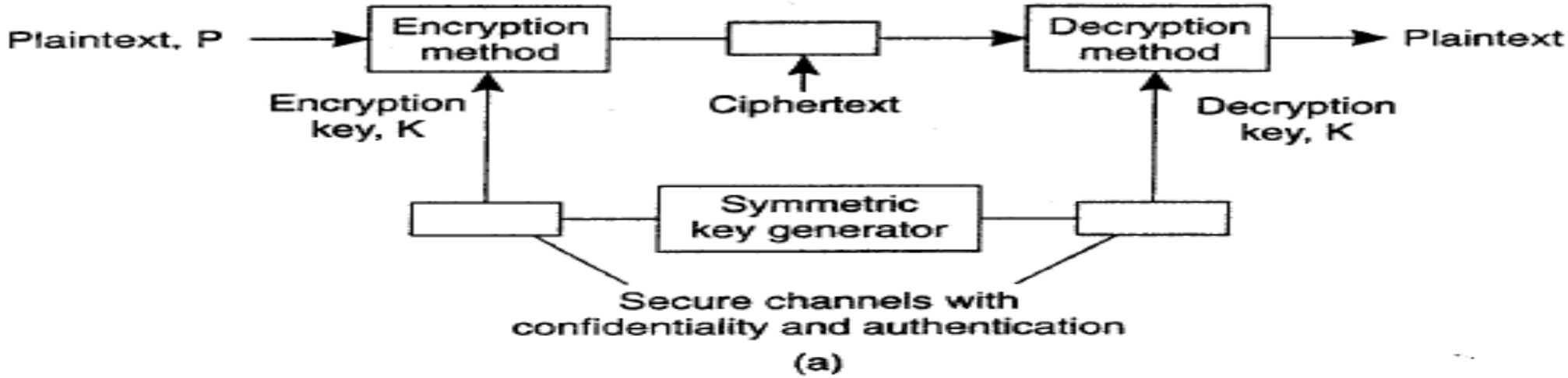


Figure 9-34. (a) Secret-key distribution. (b) Public-key distribution [see also Menezes et al. (1996)].

Lifetime of Certificates

- An important issue concerning certificates is their longevity.
- First let us consider the situation in which a certification authority hands out lifelong certificates.
- Essentially, what the certificate then states is that the public key will always be valid for the entity identified by the certificate.
- Clearly, such a statement is not what we want.
- If the private key of the identified entity is ever compromised, no unsuspecting client should ever be able to use the public key (let alone malicious clients).
- In that case, we need a mechanism to revoke the certificate by making it publicly-known that the certificate is no longer valid,

- There are several ways to revoke a certificate. One common approach is with a Certificate Revocation List (CRL) published regularly by the certification authority.
- Whenever a client checks a certificate, it will have to check the CRL to see whether the certificate has been revoked or not.
- This means that the client will at least have to contact the certification authority each time a new CRL is published.
- Note that if a CRL is published daily, it also takes a day to revoke a certificate.
- Meanwhile, a compromised certificate can be falsely used until it is published on the next CRL.
- Consequently, the time between publishing CRLs cannot be too long.
- In addition, getting a CRL incurs some overhead.

- An alternative approach is to restrict the lifetime of each certificate.
- The validity of a certificate automatically expires after some time.
- If for whatever reason the certificate should be revoked before it expires, the certification authority can still publish it on a CRL.
- However, this approach will still force clients to check the latest CRL whenever verifies a certificate.
- In other words, they will need to contact the certification authority or a trusted database containing the latest CRL.

Secure Group Management

- Many security systems make use of special services such as Key Distribution Centers (KDCs) or Certification Authorities (CAs).
- These services demonstrate a difficult problem in distributed systems.
- In the first place, they must be trusted. To enhance the trust in security services, it is necessary to provide a high degree of protection against all kinds of security threats.
- For example, as soon as a CA has been compromised, it becomes impossible to verify the validity of a public key, making the entire security system completely worthless.

- On the other hand, it is also necessary that many security services offer high availability.
- For example, in the case of a KDC, each time two processes want to set up a secure channel, at least one of them will need to contact the KDC for a shared secret key.
- If the KDC is not available, secure communication cannot be established unless an alternative technique for key establishment is available, such as the Diffie-Hellman key exchange.
- The solution to high availability is replication. On the other hand, replication makes a server more vulnerable to security attacks.
- We already discussed how secure group communication can take place by sharing a secret among the group members.
- In effect, no single group member is capable of compromising certificates, making the group itself highly secure.

Authorization Management

- Managing security in distributed systems is also concerned with managing access rights.
- So far, we have hardly touched upon the issue of how access rights are initially granted to users or groups of users, and how they are subsequently maintained in an unforgeable way.
- It is time to correct this omission.
- In nondistributed systems, managing access rights is relatively easy.
- When a new user is added to the system, that user is given initial rights, for example, to create files and subdirectories in a specific directory. create processes, use CPU time, and so on.
- In other words, a complete account for a user is set up for one specific machine in which all rights have been specified in advance by the system administrators.

Capabilities and Attribute Certificates

- A much better approach that has been widely applied in distributed systems is the use of capabilities.
- As we explained briefly above, a capability is an unforgeable data structure for a specific resource, specifying exactly the access rights that the holder of the capability has with respect to that resource.