Distributed Systems

Chapter-11

Security

Contents

Introduction

1. Security mechanism in distributed systems are required to provide privacy, integrity and availability of systems.
2. Security attack in the network are eavesdropping, masquerading , tampering and denial of services.
3. Cryptography techniques are the methods provided to apply encrypt and decrypt messages inorder to acheive secrecy and integrity in distributed systems.

4.The selection of cryptographic algorithms and keys are important to effectiveness , usability and performance of security.

5.Public keys are ideals for sharing information as distributing keys but are inadequate to bulk data. For this secret key cryotigraphy are better suited and hybrid protocols such as

6. Digital information can be signed producing digital certificates.Certificates anable trust among users and organizations.
7. This chapteer also gives an example on security mechanism based on Kerebros, SSL.TLS and 802.11 Wifi

Introduction

1. Since processes share resources among themselves they can be vulnerable to attack from outside network.
2. Some processes can be encasulated among source level and thus isolating them attacks can be avoided. Those processes can be organized by a system resource or program level authorization process.
3. But those processes which are interacting to processes at outside of network they need be kept under security policy.

Ovterview of security techniques

Worst case assumptions and design guidelines

1. Interfaces are exposed
2. Networks are secure
3. Limit the lifetime and scope of each secret
4. Algorithms and programming code are known to attackers
5. Attackers may have accesses to larger resources
6. Minimize the trusted base

Overview of security techniques

K_A	Alice's secret key
K_B	Bob's secret key
K_{AB}	Secret key shared between Alice and Bob
K_{APrvt}	Alice's private key (known only to Alice)
K_{Apub}	Alice's public key (published by Alice for all to read)
$\{M\}_k$	Message M encrypted with key K
$[M]_k$	Message M signed with key K

Table 1: Security keys as private and public keys

Cryptography

1. Encryption is the process of encoding a message in such a way as to hide its contents.
2. Modern cryptography includes **several secure algorithms for encrypting and decrypting messages.** They are all based on the use of secrets called keys.
3. A cryptographic key is a parameter used in an encryption algorithm in such a way that the encryption cannot be reversed without knowledge of the key.

Cryptography

Scenario 1:

Secret communication with a shared secret key: Alice wishes to send some information secretly to Bob. Alice and Bob share a secret key K_{AB} .

1. Alice uses K_{AB} and an agreed encryption function $E(K_{AB} , M)$ to encrypt and send any number of messages $\{M_i\} K_{AB}$ to Bob. (Alice can go on using K_{AB} as long as it is safe to assume that K_{AB} has not been compromised.)
2. Bob decrypts the encrypted messages using the corresponding decryption function $D(K_{AB} , M)$

Cryptography (Authentication)

Scenario 2:

Authenticated communication with a server-

1. Alice wishes to access files held by Bob, a file server on the local network of the organization where she works.
2. Sara is an authentication server that is securely managed.
3. Sara issues users with passwords and holds current secret keys for all of the principals in the system it serves (generated b applying some transformation to the user's password).

Cryptography (Authentication)

For example :

It knows Alice’s key K_A and Bob’s K_B . In this scenario this has been referred to a ticket. A ticket is an encrypted item issued by an authentication server, containing the identity of the principal to whom it is issued and a shared key that has been generated for the current communication session.

Cryptography

1. Alice sends an (unencrypted) message to Sara stating her identity and requestingg a ticket for access to Bob.

2. Sara sends a response to Alice encrypted in K_B consisting of a ticket (to be sent to Bob with each request for file access) encrypted in K_B and a new secret key K_{AB} for use when communicating with Bob. So the response that Alice receives looks like this: $\{\{Ticket\}\ K_B\ ,\ K_{AB}\ \}\ K_A$.

Cryptography

Alice decrypts the response using K A (which she generates from her password

using the same transformation; the password is not transmitted over the network,

and once it has been used it is deleted from local storage to avoid compromising

it). If Alice has the correct password-derived key K A , she obtains a valid ticket for

using Bob’s service and a new encryption key for use in communicating with Bob.

Alice can’t decrypt or tamper with the ticket,

Cryptography

5. The ticket, originally created by Sara, is actually:

   $\{K_{AB}$, Alice$\}$ K_B . Bob decrypts the ticket using his key K_B .

So Bob gets the authentic identity of Alice (based on the knowledge shared between Alice and Sara of Alice’s password) and a new shared secret key K AB for use when interacting with Alice. (This is called a session key because it can safely be used by Alice and Bob for a sequence of interactions.)

Cryptography

<u>Scenario 3:</u>

Authenticated communication with public keys: Assuming that Bob has generated a public/private key pair, the following dialogue enables Bob and Alice to establish a shared secret key, K_{AB} :

1. Alice accesses a key distribution service to obtain a public-key certificate giving Bob's public key. It's called a certificate because it is signed by a trusted authority a person or organization that is widely known to be reliable. After checking the signature, she reads Bob's public

Cryptography

2. Alice creates a new shared key, K_{AB} , and encrypts it using $K\ B_{pub}$ with a public- key algorithm. She sends the result to Bob, along with a name that uniquely identifies a public/private key pair (since Bob may have several of them) – that is,Alice sends keyname,$\{K_{AB}\}\ K\ B_{pub}$.

3. Bob selects the corresponding private key, $K\ B_{priv}$, from his private key store and uses it to decrypt K_{AB} . Note that Alice's message to Bob might have been corrupted or tampered with in transit. The consequence would simply be that Bob and Alice don't share the same key K_{AB} . If this is a problem, it can be circumvented by adding an agreed value or string to the message, such as Bob'sand Alice's names or email addresses, which Bob can check after decrypting.

Cryptography

Scenario 3:

Authenticated communication with public keys: Assuming that Bob has generated a public/private key pair, the following dialogue enables Bob and Alice to establish a shared secret key, K_{AB} :

1. Alice accesses a key distribution service to obtain a public-key certificate giving Bob's public key. It's called a certificate because it is signed by a trusted authority a person or organization that is widely known to be reliable. After checking the signature, she reads Bob's public

Uses of cryptography

Scenario 1:

Secret communication with a shared secret key: Alice wishes to send some information secretly to Bob. Alice and Bob share a secret key K_{AB}.

1. Alice uses K_{AB} and an agreed encryption function $E(K_{AB}, M)$ to encrypt and send any number of messages $\{M_i\} K_{AB}$ to Bob. (Alice can go on using K_{AB} as long as it is safe to assume that K_{AB} has not been compromised.)

2. Bob decrypts the encrypted messages using the corresponding decryption function $D(K_{AB}, M)$

Dligital Signatures

1. Authentic: It convinces the recipient that the signer deliberately signed the document and it has not been altered by anyone else.
2. Unforgeable: It provides proof that the signer, and no one else, deliberately signed the document. The signature cannot be copied and placed on another document.
3. Non-repudiable: The signer cannot credibly deny that the document was signed by them.

Certificates

A digital **certificate** is a document containing a statement (usually short) signed by a principal. Here is a scenario.

Scenario 5

The use of certificates:

Bob is a bank. When his customers establish contact with him they need to be sure that they are talking to Bob the bank, even if they have never contacted him before. Bob needs to authenticate his customers before he gives them access to their accounts.

Acess Control

1. Historically, the protection of resources in distributed systems has been largely service-specific. Servers receive request messages of the form <op, principal, resource>, where op is the requested operation, principal is an identity or a set of credentials for the principal making the request and resource identifies the resource to which the operation is to be applied.

2. The server must first authenticate the request message and the principal's credentials and then apply access control, refusing any request

Acess Control

Protection domains:

A protection domain is an execution environment shared by a collection of processes: it contains a set of <resource, rights> pairs, listing the resources that can be accessed by all processes executing within the domain and specifying the operations permitted on each resource.

Access control lists:

A list is stored with each resource, containing an entry of the form <domain, operations> for each domain that has access to the resource and giving the operations permitted to the domain. A domain may be specified by an identifier for a principal or it may be an expression that can be used to determine a principal’s membership of the domain. For example, the owner of this file is an expression that can be evaluated by comparing the requesting principal’s identity with the owner’s identity stored with a file.

Credentials

1. **Credentials** are a set of evidence provided by a principal when requesting access to a resource. In the simplest case, a certificate from a relevant authority stating the principal’s identity is sufficient, and this would be used to check the principal’s permissions in an access control list .
2. This is often all that is required or provided, but the concept can be generalized to deal with many more subtle requirements.

Credentials

Delegation

A particularly useful form of credential is one that entitles a principal, or a process acting for a principal, to perform an action with the authority of another principal. A need for delegation can arise in any situation where a service needs to access a protected resource in order to complete an action on behalf of its client.

Example : A print server that accepts requests to print files.

Firewalls

Firewalls protect intranets, performing filtering actions on incoming and outgoing communications. Here are a few advantages and drawbacks as security mechanisms.

1. Firewalls produce a local communication environment in which all external communication is intercepted. Messages are forwarded to the intended local recipient only for communications that are explicitly authorized.
2. Access to internal networks may be controlled by firewalls, but access to public services on the Internet is unrestricted because their purpose is to offer services to a wide range of users.

Firewalls

3. Firewalls are **not** particularly effective against **denial-of-service** attacks such as the one based on IP spoofing. The problem is that the flood of messages generated by such attacks overwhelms any single point of defence such as a firewall. Any remedy for incoming floods of messages must be applied well upstream of the target.

Cryptographic Algorithms

1. A message is encrypted by the sender applying some rule to transform the plaintext message (any sequence of bits) to a ciphertext (a different sequence of bits).

2. The recipient must know the inverse rule in order to transform the ciphertext back into the original plaintext. Other principals are unable to decipher the message unless they also know the inverse rule.

3.The **encryption** transformation is defined with two parts, a

function E and a key K. The resulting encrypted message is written $\{M\}_k$.

$$E(K, M) = \{M\}_k$$

Cryptographic Algorithms

4. The **encryption** function E defines an algorithm that transforms data items in plaintext into encrypted data items by combining them with the key and transposing them in a manner that is heavily dependent on the value of the key.
5. **Decryption** is carried out using an inverse function D, which also takes a key as a parameter. For secret-key encryption, the key used for decryption is the same as that used for encryption:

$$D(K, E(K, M)) = M$$

Cryptographic Algorithms

Symmetric Algorithm:

If we remove the key parameter from consideration by defining

$F_K([M]) = E(K, M)$, then it is a property of strong encryption functions that $F_K([M])$ is relatively easy to compute, whereas the inverse, $F_K^{-1}([M])$, is so hard to compute that it is not feasible. Such functions are known as one-way functions. The effectiveness of any method for encrypting information depends upon the use of an encryption function F K that has this one-way property. It is this that protects against attacks designed to discover M given $\{M\}_K$.

Cryptographic Algorithms

<u>Asymmetric algorithms:</u>

1. When a public/private key pair is used, one-way functions are exploited in another way. The feasibility of a public-key scheme was first proposed by Diffie and Hellman [1976] as a cryptographic method that eliminates the need for trust between the communicating parties.

2. The basis for all public-key schemes is the existence of trap-door functions. A trap-door function is a one-way function with a secret exit – it is easy to compute in one direction but infeasible to compute the inverse unless a secret is known.

Cryptographic Algorithms

Block ciphers :

1. Most encryption algorithms operate on fixed-size blocks of data; 64 bits is a popular size for the blocks. A message is subdivided into blocks, the last block is padded to the standard length if necessary and each block is encrypted independently.
2. The first block is available for transmission as soon as it has been encrypted.

Cryptographic Algorithms

Stream Ciphers :

1. For some applications, such as the encryption of telephone conversations, encryption in blocks is inappropriate because the data streams are produced in real time in small chunks.
2. Data samples can be as small as 8 bits or even a single bit, and it would be wasteful to pad each of these to 64 bits before encrypting and transmitting them.
3. Stream ciphers are encryption algorithms that can perform encryption incrementally, converting plaintext to ciphertext one bit at a

Cryptographic Algorithms

Design of Cryptographic Algorithms:

1. There are many well-designed cryptographic algorithms such that E (K, M) = $\{M\}_K$ conceals the value of M and makes it practically impossible to retrieve K more quickly than by brute force.
2. All encryption algorithms rely on information-preserving manipulations of M using principles based on information theory [Shannon 1949]. Schneier [1996] describes Shannon's principles of confusion and diffusion to conceal the content of a ciphertext block M, combining it with a key K of sufficient size to render it proof against brute-force attacks.

Cryptographic Algorithms

Confusion:

1. Non-destructive operations such as XOR and circular shifting are used to combine each block of plaintext with the key, producing a new bit pattern that obscures the relationship between the blocks in M and $\{M\}_k$.
2. If the blocks are larger than a few characters this will defeat analysis based on a knowledge of character frequencies. (The WWII German Enigma machine used chained single-letter blocks, and was eventually defeated by statistical analysis.)

Cryptographic Algorithms

Diffusion:

There is usually repetition and redundancy in the plaintext. Diffusion dissipates the regular patterns that result by transposing portions of each plaintext block.

If CBC is used, the redundancy is also distributed throughout a longer text. Stream ciphers cannot use diffusion since there are no blocks.

Cryptographic Algorithms

Secret key (Symmetric Algorithms):

Many cryptographic algorithms have been developed and published in recent years. Schneier [1996] describes more than 25 symmetric algorithms, many of which he identifies as secure against known attacks. Here only three of them have been described vividly.

1. TEA (Tiny Encryption Algorithm) 2. **DES** (Data Encryption Standard) 3. IDEA(International Data Encryption Algorithms) 4. RC4 5. **AES** (Advanced Encryptio n Standard)

Cryptographic Algorithms

TEA() :

1. The design principles for symmetric algorithms outlined above are illustrated well in the Tiny Encryption Algorithm (TEA) developed at Cambridge University [Wheeler and Needham 1994].

2. The TEA algorithm uses rounds of integer addition, XOR (the ^ operator) and bitwise logical shifts (<< and >>) to achieve diffusion and confusion of the bit patterns in the plaintext. The plaintext is a 64-bit block represented as two 32-bit integers in the vector text[]. The key is

Cryptographic Algorithms

IDEA():

The International Data Encryption Algorithm (IDEA) was developed in the early 1990s [Lai and Massey 1990, Lai 1992] as a successor to DES. Like TEA, it uses a 128-bit key to encrypt 64-bit blocks. Its algorithm is based on the algebra of groups and has eight rounds of XOR, addition modulo 2 16 and multiplication. For both DES and IDEA, the same function is used for encryption and decryption: a useful property for algorithms that are to be implemented in hardware.

Cryptographic Algorithms

<u>RC4 :</u>

1. RC4 is a stream cipher developed by Ronald Rivest [Rivest 1992b]. Keys can be of any length up to 256 bytes. RC4 is easy to implement [Schneier 1996, pp. 397–8] and performs encryption and decryption about 10 times as fast as DES.
2. It was therefore widely adopted in applications including IEEE 802.11 WiFi networks, but a weakness was subsequently discovered by Fluhrer et al. [2001] that enabled attackers to

Cryptographic Algorithms

AES():

1. The Rijndael algorithm selected to become the Advanced Encryption Standard algorithm by NIST was developed by Joan Daemen and Vincent Rijmen [Daemen and Rijmen 2000, 2002].
2. The cipher has a variable block length and key length, with specifications for keys with a length of 128, 192 or 256 bits to encrypt blocks with a length of 128, 192 or 256 bits.

Cryptographic Algorithms

Public-key (asymmetric) algorithms:

1. Only a few practical public-key schemes have been developed to date. They depend upon the use of trap-door functions of large numbers to produce the keys.
2. The keys Ke and Kd are a pair of very large numbers, and the encryption function performs an operation, such as exponentiation on M, using one of them.
3. Decryption is a similar function using the other key. If the exponentiation uses modular arithmetic, it can be shown that the result is the

Cryptographic Algorithms

Rivest, Shamir and Adelman (RSA):

1. The Rivest, Shamir and Adelman (RSA) design for a public-key cipher [Rivest et al. 1978] is based on the use of the product of two very large prime numbers (greater than 10 100), relying on the fact that the determination of the prime factor.

Cryptographic Algorithms

Elliptic curve algorithms :

1. A method for generating public/private key pairs based on the properties of elliptic curves has been developed and tested. Full details can be found in the book by Menezes devoted to the subject [Menezes 1993].
2. The keys are derived from a different branch of mathematics, and unlike RSA their security does not depend upon the difficulty of factoring large numbers.

Cryptographic Algorithms

Hybrid cryptographic protocols:

1. Public-key cryptography is convenient for electronic commerce because there is no need for a secure key-distribution mechanism. (There is a need to authenticate public keys,but this is much less onerous, requiring only a public-key certificate to be sent with the key.)
2. But the processing costs of public-key cryptography are too high for the encryption of even the medium-sized messages normally encountered in electronic commerce.
3. The solution adopted in most large-scale

Cryptographic Algorithms

Public-key (asymmetric) algorithms:

1. Only a few practical public-key schemes have been developed to date. They depend upon the use of trap-door functions of large numbers to produce the keys.

2. The keys Ke and Kd are a pair of very large numbers, and the encryption function performs an operation, such as exponentiation on M, using one of them.

3. Decryption is a similar function using the other key. If the exponentiation uses modular arithmetic, it can be shown that the result is the

Digital Signatures

2. Strong digital signatures are an essential requirement for secure systems. They are needed in order to certify certain pieces of information – for example, to provide trustworthy statements binding users' identities to their public keys or binding some access rights or roles to users' identities.

3. The need for signatures in many kinds of business and personal transaction is beyond dispute. Handwritten signatures have been used as a means of verifying documents for as long

Digital Signatures

1. Authentic: It convinces the recipient that the signer deliberately signed the document and it has not been altered by anyone else.
2. Unforgeable: It provides proof that the signer, and no one else, deliberately signed the document. The signature cannot be copied and placed on another document.
3. Non-repudiable: The signer cannot credibly deny that the document was signed by them.

Digital Signatures

<u>Digital signing</u>

1. An electronic document or message M can be signed by a principal A by encrypting a py of M with a key K A and attaching it to a plaintext copy of M and A’s identifier.
2. The signed document then consists of: M, A, [M] K A .
3. The signature can be verified by a principal that subsequently receives the document to check that it was originated by A and that its contents, M, have not subsequently been altered.

Digital Signatures

Digest functions :

1. Digest functions are also called secure hash functions and denoted H(M).
2. They must be carefully designed to ensure that H(M) is different from H(M') for all likely pairs of messages M and M'.
3. If there are any pairs of different messages M and M' such that H(M) = H(M'), then a duplicitous principal could send a signed copy of M, but when confronted with it claim that M' was originally sent and that it must have been altered in transit

Digital Signatures

Digital signatures with public keys :

Public-key cryptography is particularly well adapted for the generation of digital signatures because it is relatively simple and does not require any communication between the recipient of a signed document and the signer or any third party.

The method for A to sign a message M and B to verify it is as follows:

1. A generates a key pair Kpub and Kpriv and publishes the public key K_{pub} by placing it in a well known location

Digital Signatures

2. A computes the digest of M, H(M) using an agreed secure hash function H and

encrypts it using the private key Kpriv to produce the signature S = {H(M)} K_{priv} .

3. A sends the signed message $[M]_K$ = M,S to B.

4. B decrypts S using Kpub and computes the digest of M, H(M). If they match, the signature is valid.

The RSA algorithm is quite suitable for use in constructing digital signatures. Here the private key of the signer is used to encrypt the

Digital Signatures

Digital signatures with secret keys – MACs

1. There is no technical reason why a secret-key encryption algorithm should not be used to encrypt a digital signature, but in order to verify such signatures the key must be disclosed, and this causes some problems:

- The signer must arrange for the verifier to receive the secret key used for signing securely.

Digital Signatures

- It may be necessary to verify a signature in several contexts and at different times
  - at the time of signing, the signer may not know the identities of the verifiers. To resolve this, verification could be delegated to a trusted third party who holds secret keys for all signers, but this adds complexity to the security model and requires secure communication with the trusted third party.
- The disclosure of a secret key used for signing is undesirable because it weakens the security of signatures made with that key – a signature

Case studies : Needham-Shroeder

Header	Message	Notes
1. A -> S: A, B, N A		A requests S to supply a key for communication with B.
2. S -> A:	{N A , B, K AB , {K AB , A} K B } K A	S returns a message encrypted in A’s secret key, containing a newly generated key K AB , and a‘ticket’ encrypted in B’s secret key. The nonce N A demonstrates that the message was sent in responseto the preceding one. A believes that S sent the message because only S knows A’s secret key.

Needham-Shroeder’s secret key authetication protocol

Case studies : Needham-Shroeder

Header	Message	Notes
1. A -> S: A, B, N A		A requests S to supply a key for communication with B.
2. S -> A:	{N A , B, K AB , {K AB , A} K B } K A	S returns a message encrypted in A's secret key, containing a newly generated key K AB , and a 'ticket' encrypted in B's secret key. The nonce N A demonstrates that the message was sent in response to the preceding one. A believes that S sent the message because only S knows A's secret key.
3. A o B: {K AB , A} K B		A sends the ticket to B.

Needham-Shroeder's secret key authetication protocol

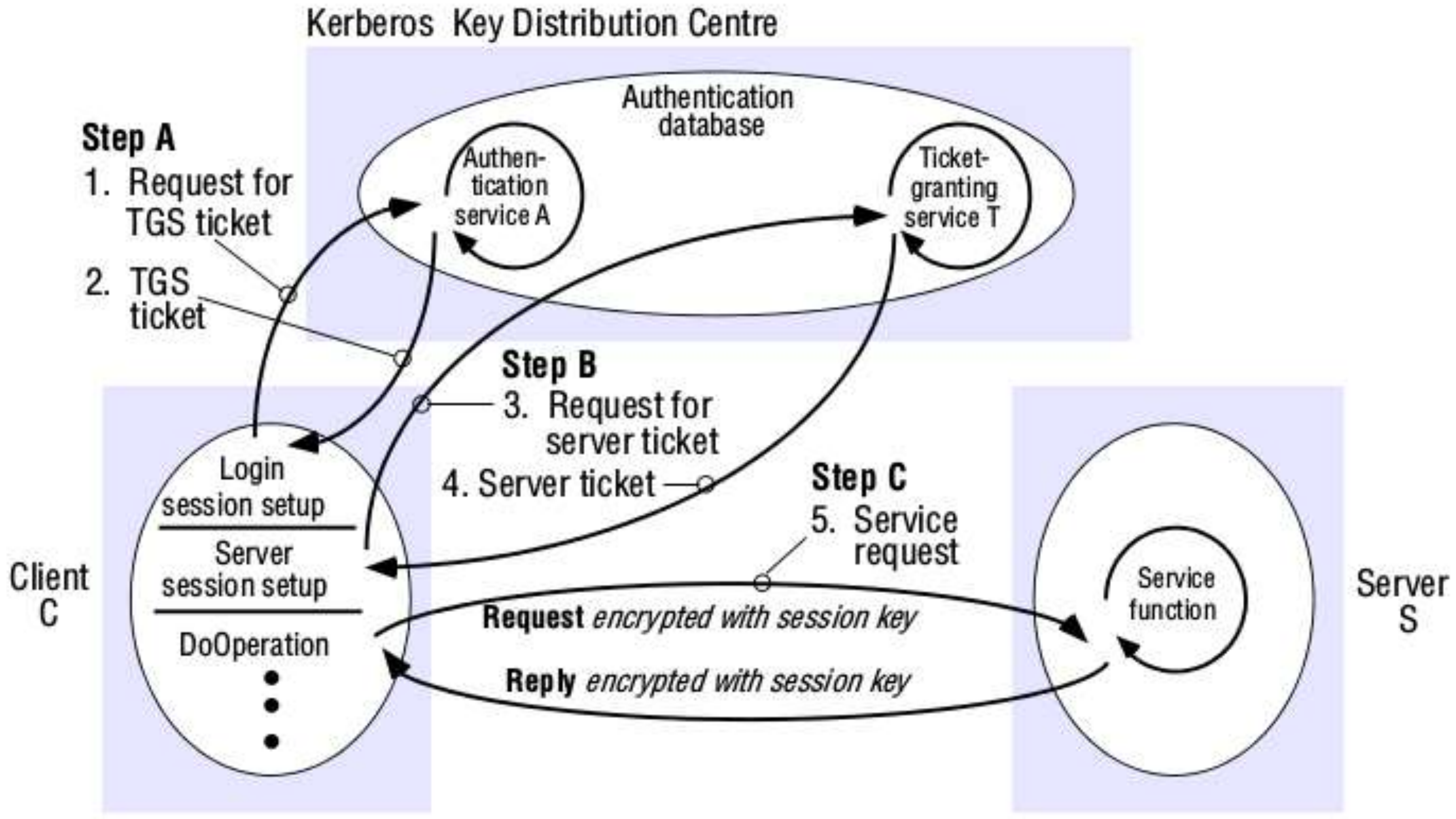
Case studies: Kerebros

1. public-key certficates providers

Kerberos was developed at MIT in the 1980s [Steiner et al. 1988] to provide a range of authentication and security facilities for use in the campus computing network at MIT and other intranets.

It has undergone several revisions and enhancements in the light of experience and feedback from user organizations. Kerberos version 5 [Neuman and Ts’o 1994], which we describe here, is an Internet standard (see RFC 4120 [Neuman et al. 2005]) and is used by

Kerebros



Kerberos Key Distribution Centre
Authentication database
Authen-tication service A
Ticket-granting service T
Step A
1. Request for TGS ticket
2. TGS ticket
Step B
3. Request for server ticket
4. Server ticket
Step C
5. Service request
Client C
Login session setup
Server session setup
DoOperation
Request encrypted with session key
Reply encrypted with session key
Service function
Server S

Figure 2 : Diagram for Kerebros system architecture

References

1. Coulouris, George, Jean Dollimore, and Tim Kindberg. "Distributed Systems: Concepts and Design Edition 3." (2001).

Questions ?