Note: Khi làm việc trên Word, sử dụng những mẫu chữ là Heading 1, 2, … và Normal để dễ chỉnh sửa tự động về sau hơn.

Contents

[Contents 2](#_Toc161230325)

[I. Remote User-Authentication Principles 3](#_Toc161230326)

[II. Remote User-authentication using Symmetric encryption 4](#_Toc161230327)

[1. Mutual Authentication 4](#_Toc161230328)

[2. One-Way Authentication 7](#_Toc161230329)

[III. Kerberos 9](#_Toc161230330)

[IV. Remote User-Authentication using Asymmetric Encryption 10](#_Toc161230331)

[V. Real Life Applications 11](#_Toc161230332)

[VI. References 12](#_Toc161230333)

# Remote User-Authentication Principles

# Remote User-authentication using Symmetric encryption

## Mutual Authentication

Mutual authentication, as the name implies, involves both parties proving their identities to each other. This ensures that not only is the client confident in the server's identity, but the server is also confident in the client's identity. Mutual authentication is crucial in scenarios where trust needs to be established in both directions.

An example of mutual authentication is seen in some secure communication protocols like Kerberos. In Kerberos, the client proves its identity to the Key Distribution Center (KDC) and receives a ticket. When the client presents this ticket to the server, the server can verify the ticket's validity with the help of the KDC, thus authenticating the client. Meanwhile, the server also proves its identity to the client by showing that it can decrypt the information sent by the KDC, which could only be done if the server is legitimate.

**Key Distribution Center (KDC)**:

* + - Each party in the network shares a secret key, known as a master key, with the KDC. The KDC is responsible for generating keys to be used for a short time over a connection between two parties, known as session keys, and for distributing those keys using the master keys to protect the distribution.
    - [Needham–Schroeder protocol](https://en.wikipedia.org/wiki/Needham–Schroeder_protocol#See_also) [NEED78] for secret key distribution:
      * The protocol can be summarized as follows:
        1. f () is a generic function that modifies the values of nonce (N)

Diagram of a diagram of a key distribution center

Description automatically generated

A and B are identities of Alice and Bob respectively, Secret keys and are shared between A and the KDC and B and the KDC, respectively. The purpose of the protocol is to distribute securely a session key to A and B. Firstly A send a request to KDC include: A’s ID, B’s ID and a nonce, In step 2 A receive and an encrypted message from KDC that only B can decrypt, A then send this message to B in step 3. In step 4 B acknowledges and step 5 assures A know and assures B that this is a fresh message by the uses of nonce . The purpose of steps 4 and 5 is to prevent a certain type of replay attack. In particular, if an opponent is able to capture the message in step 3 and replay it, this might in some fashion disrupt operations at B.

Despite the handshake of steps 4 and 5, the protocol is still vulnerable to a form of replay attack. Suppose that an opponent, X, has been able to compromise an old session key. X can impersonate A and trick B into using the old key by simply replaying step 3. Unless B remembers indefinitely all previous session keys used with A, B will be unable to determine that this is a replay. If X can intercept the handshake message in step 4, then it can impersonate A’s response in step 5. From this point on, X can send bogus messages to B that appear to B to come from A using an authenticated session key.

* + - Denning [DENN81, DENN82] proposes to overcome this weakness by a modification to the Needham/Schroeder protocol that includes the addition of a timestamp to steps 2 and 3:
      * Her proposal assumes that the master keys, and , are secure, and it consists of the following steps:
      * T is a timestamp that assures A and B that the session key has only just been generated. Thus, both A and B know that the key distribution is a fresh exchange. A and B can verify timeliness by checking that where is the estimated normal discrepancy between the KDC’s clock and the local clock (at A or B) and is the expected network delay time. Each node can set its clock against some standard reference source. Because the timestamp T is encrypted using the secure master keys, an opponent, even with knowledge of an old session key, cannot succeed because a replay of step 3 will be detected by B as untimely.
      * The Denning protocol seems to provide an increased degree of security compared to the Needham/Schroeder protocol. However, a new concern is raised: namely, that this new scheme requires reliance on clocks that are synchronized throughout the network. [GONG92] points out a risk involved. The risk is based on the fact that the distributed clocks can become unsynchronized as a result of sabotage on or faults in the clocks or the synchronization mechanism. The problem occurs when a sender’s clock is ahead of the intended recipient’s clock. In this case, an opponent can intercept a message from the sender and replay it later when the timestamp in the message becomes current at the recipient’s site. This replay could cause unexpected results. Gong refers to such attacks as **suppress-replay attacks**.
      * One way to counter suppress-replay attacks is to enforce the requirement that parties regularly check their clocks against the KDC’s clock. The other alternative, which avoids the need for clock synchronization, is to rely on handshaking protocols using nonces. This latter alternative is not vulnerable to a suppress-replay attack, because the nonces the recipient will choose in the future are unpredictable to the sender.
    - In [KEHN92], an attempt is made to respond to the concerns about suppressing replay attacks and at the same time fix the problems in the Needham/Schroeder protocol.
      * The attempt was still inconsistent but there was a improved new strategy presented in [NEUM93a]:
      * Let us follow this exchange step by step:
        1. A initiate the authentication exchange by generate a nonce, , then sent to B in plaintext.
        2. B request KDC a session key by sending KDC: B’s ID and nonce (). B also send an encrypted block using the key B share between B and the KDC () which include A’s ID, Nonce and B’s timestamp. This block is used to instruct the KDC to issue credentials to A and specify A’s information and a suggested credentials expiration time.
        3. A receives B’s nonce and two block from KDC, one A can decrypt using the key A share with KDC and the other block A can’t decrypt. The block A can decrypt includes: to verify that B received A initial message, to ensure that this is not a replay, Session key , and time limit for on its use ().
        4. A then send B the block A can’t decrypt which work like a ticket to B alongside B’s nonce which encrypted with session key. B can then decrypt that block and receive session key, A’s ID to confirm sender. B can the use session key to decrypt the other block to recover B’s nonce then B can confirm that this is not a replay message.
      * This protocol provides an effective, secure means for A and B to establish a session with a secure session key. Furthermore, the protocol leaves A in possession of a key that can be used for subsequent authentication to B, avoiding the need to contact the authentication server repeatedly.
      * Suppose that A and B establish a session using the aforementioned protocol and then conclude that session. Subsequently, but within the time limit established by the protocol, A desires a new session with B. The following protocol ensues:
      * When B receives the message in step 1, it verifies that the ticket has not expired. The newly generated nonces and assure each party that there is no replay attack.
      * In all the foregoing, the time specified in is a time relative to B’s clock. Thus, this timestamp does not require synchronized clocks, because B checks only self-generated timestamps.

Some authentication protocols that support mutual authentication:

* + - Kerberos: Supports mutual authentication. Both the client and the server verify each other's identity using tickets provided by the Key Distribution Center (KDC). Kerberos was used in large networks of companies such as AWS, Google Cloud, and Microsoft Azure to facilitate SSO.
    - NT LAN Manager (NTLM): Provides mutual authentication. Both the client and the server verify each other's identity through a challenge-response mechanism, where the client proves its knowledge of the password through a calculation without sending the password to the server. SharePoint, a web-based collaborative platform that is part of the Microsoft Office suite, employs NTLM for authentication in environments where more modern protocols like OAuth are not configured. This is particularly common in intranet scenarios within organizations.
    - Challenge Handshake Authentication Protocol (CHAP): Used by Point-to-Point Protocol (PPP) servers to validate the identity of remote clients. CHAP periodically verifies the identity of the client by using a three-way handshake, where the server sends a challenge to the client, the client responds with a value calculated using a one-way hash function, and the server checks this response against its own calculation of the expected hash value. Internet Service Providers (ISPs) like AT&T and Verizon have used CHAP in the past for dial-up and DSL services to authenticate subscribers' Internet connections.
    - Secure Remote Password (SRP): A password-authenticated key exchange protocol that allows the establishment of a secure communication channel without transmitting the password over the network. It's designed to work around the vulnerabilities of transmitting passwords over insecure networks. 1Password, a popular password management service, implements a version of the Secure Remote Password protocol to authenticate users to their servers securely. This ensures that even during the login process, users' passwords are never transmitted or at risk of interception, enhancing the security of user data.

## One-Way Authentication

* + In one-way authentication, only one party proves its identity to the other, while the identity of the second party remains unverified. This is commonly seen in scenarios where a client needs to prove its identity to a server to gain access to resources, but the server does not need to prove its identity to the client.
  + A typical example of one-way authentication is when you log into a website. You, as the user, provide your username and password to the server to prove who you are. However, unless a separate mechanism is used (like certificates in SSL/TLS), the server does not prove its identity to you in the same way.
  + The Key Distribution Center (KDC) strategy for single-side authentication using symmetric encryption simplifies secure communications by managing the distribution of secret keys within a network. This approach is particularly useful in environments where multiple clients need to securely authenticate various services or servers.
    - With some refinement, the KDC strategy illustrated above is a candidate for encrypted electronic mail. Because we wish to avoid requiring that the recipient (B) be online at the same time as the sender (A), steps 4 and 5 must be eliminated. For a message with content M, the sequence is as follows:
    - This approach guarantees that only the intended recipient of a message will be able to read it. It also provides a level of authentication that the sender is A. As specified, the protocol does not protect against replays. Some measure of defense could be provided by including a timestamp with the message. However, because of the potential delays in the email process, such timestamps may have limited usefulness.
  + Some authentication protocols that support one-way authentication:
    - Password Authentication Protocol (PAP): A simple, and less secure, authentication protocol where passwords are sent over the network as plain text. It's often used in some contexts where security is not a major concern or is provided by other means.
    - LAN Manager (LM) Hash: An outdated and insecure authentication protocol used by Microsoft for network LAN Manager sessions. It uses a hashed version of the user's password for authentication, which has been known to be susceptible to various attacks and is generally not recommended for use.

# Kerberos

# Remote User-Authentication using Asymmetric Encryption

# Real Life Applications

# References

<https://en.wikipedia.org/wiki/Needham–Schroeder_protocol>

<https://en.wikipedia.org/wiki/Kerberos_(protocol)>

<https://en.wikipedia.org/wiki/Authentication_protocol>

<https://en.wikipedia.org/wiki/Mutual_authentication>

<https://en.wikipedia.org/wiki/Kerberos_(protocol)>

<https://en.wikipedia.org/wiki/Challenge-Handshake_Authentication_Protocol>

<https://en.wikipedia.org/wiki/NTLM>

<https://en.wikipedia.org/wiki/Secure_Remote_Password_protocol>

<https://secgroup.dais.unive.it/teaching/cryptography/strong-authentication/#:~:text=We%20discuss%20strong%20authentication%20protocols,a%20message%20encrypted%20under%20K>.

<https://www.brainkart.com/article/Remote-User-Authentication-Using-Symmetric-Encryption_8474/#:~:text=Secret%20keys%20Ka%20and%20Kb,Ks%20to%20A%20and%20B>.

<https://www.net.in.tum.de/pub/netsec2016/10_cryptoprot.pdf>