User Authentication

Tecnologie Internet

a.a. 2022/2023

Summary

- Principles of User Authentication
- User Authentication with Symmetric Encryption
- User Authentication with Asymmetric Encryption
- Federated Identity Management

Principles of User Authentication

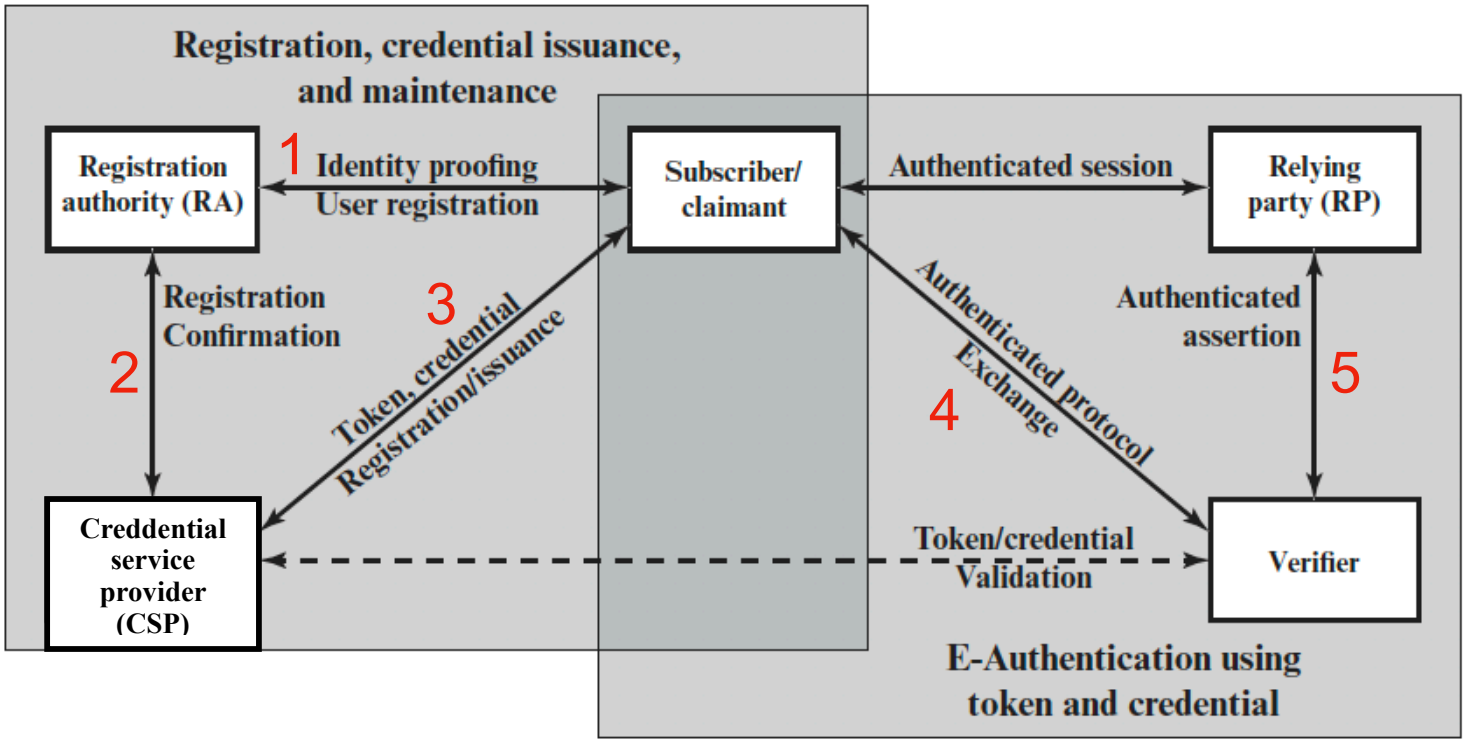
RFC 4949 (Internet Security Glossary) defines user authentication (equivalently, peer entity authentication) as the process of **verifying an identity** claimed by or for a system entity.

- **Identification step**: presenting an identifier to the security system
- **Verification step**: presenting or generating authentication information that corroborates the binding between the entity and the identifier

In essence, identification is the means by which a user provides a claimed identity to the system; user authentication is the means of establishing the validity of the claim. Note that user authentication is distinct from message authentication.

NIST Model for Electronic User Authentication

NIST SP 800-63-2 (Electronic Authentication Guideline, August 2013) defines electronic user authentication as the process of establishing confidence in **user identities that are presented electronically to an information system**.



NIST Model for Electronic User Authentication

The initial requirement for performing user authentication is that the user must be registered with the system.

An applicant applies to a **registration authority (RA)** to become a subscriber of a **credential service provider (CSP)**. The RA is a trusted entity that establishes and vouches for the identity of an applicant to a CSP.

The **credential** issued by the CSP to the subscriber is a data structure that authoritatively binds an identity and additional attributes to a **token** possessed by a subscriber, and can be verified when presented to the verifier in an authentication transaction.

Examples: encryption key, encrypted password, SPID, .. generated by the CSP, or by the subscriber, or by a third party

NIST Model for Electronic User Authentication

Once a user is registered as a subscriber, the actual **authentication process** can take place between the subscriber and one or more systems that perform authentication and, subsequently, authorization.

The party to be authenticated is called a **claimant** and the party verifying that identity is called a **verifier**.

When the claimant demonstrates possession of the credential, the verifier transmits an assertion about the identity of the subscriber to the **relying party (RP)**, which makes access control and authorization decisions.

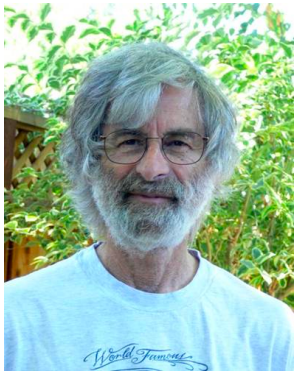
Means of authentication

There are four general means of authenticating a user's identity:

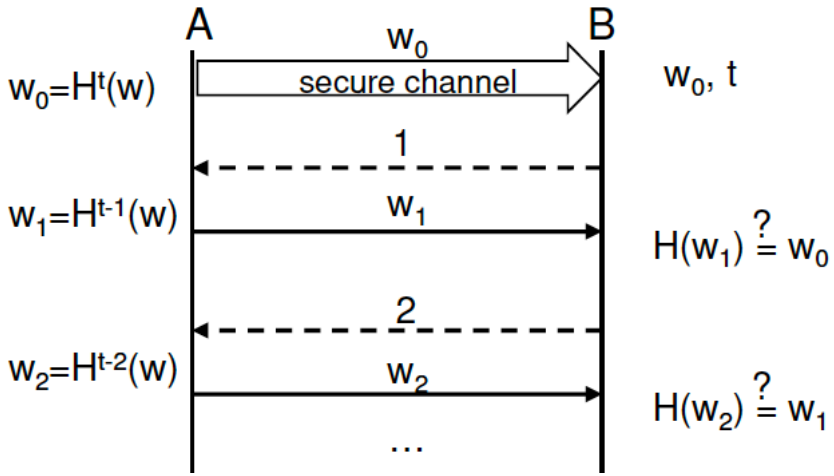
- **Something the individual knows**: a password, a personal identification number (PIN), answers to a prearranged set of questions, etc.
- **Something the individual possesses**: cryptographic keys, electronic keycards, smart cards, and physical keys (this type of authenticator is referred to as token)
- **Something the individual is (static biometrics)**: recognition by fingerprint, retina, face, etc.
- **Something the individual does (dynamic biometrics)**: recognition by voice pattern, handwriting characteristics, typing rhythm, etc.

Lamport's One-Time Password

- User A begins with a secret w and a constant t (e.g., t = 100 or 1000), defining the number of identifications to be allowed
- A computes $H(w)$, $H(H(w))$, ... , $H^t(w)$ using a one-way function H
- A sends $\{ID_A, w_0 = H^t(w)\}$ to B over a secure channel
- B initializes its counter i_A for A to 1
- For i=1 to t
  - A sends $\{ID_A, i, w_i = H^{t-i}(w)\}$ to B over a secure channel
  - B checks that $i = i_A$ and that $H(w_i) = w_{i-1}$



http://www.lamport.org/



Mutual authentication

Mutual authentication protocols enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys.

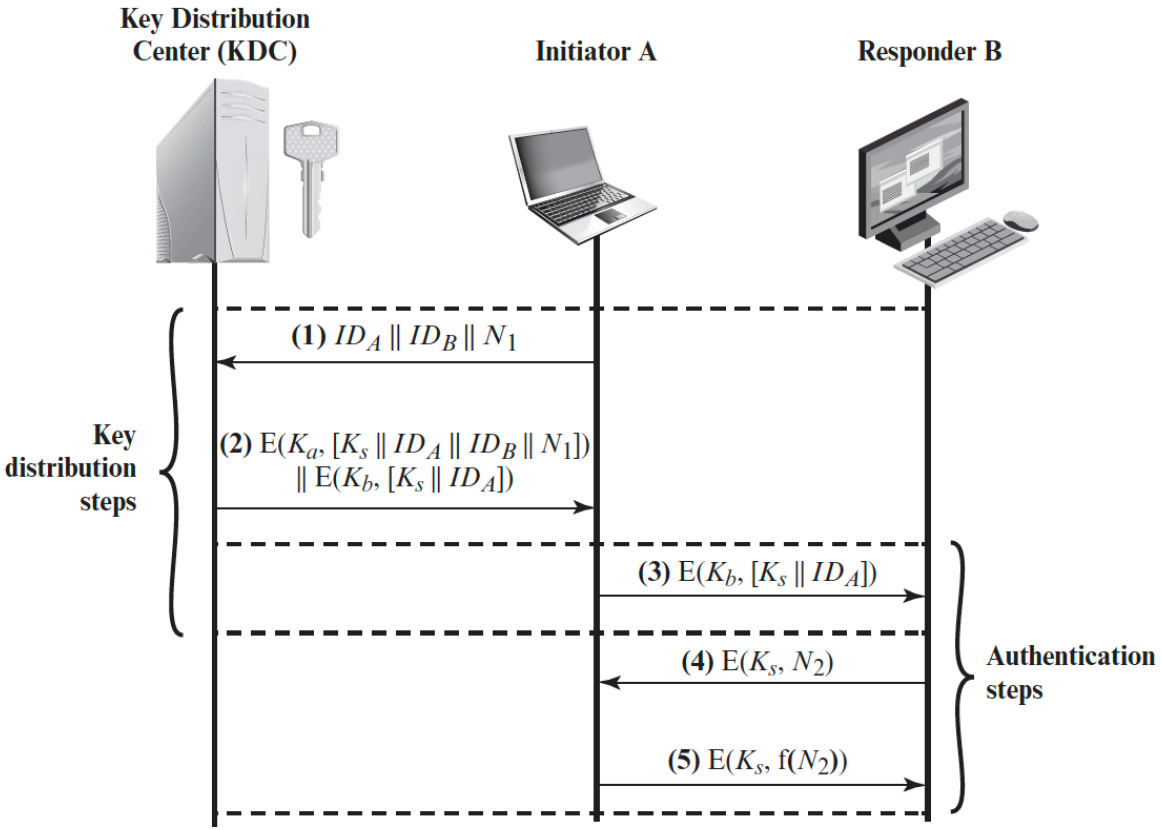
Major problem: **replay attacks**.

One of the following two general approaches is used:

- **Timestamps**: Party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time. This approach requires that clocks among the various participants be synchronized.
- **Challenge/response**: Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value (or a function of the nonce).

User Authentication with Symmetric Encryption

Recall the KDC-based **session key** distribution scheme.
Communications with the KDC are encrypted with a **master key** (a different one for each user).



User Authentication with Symmetric Encryption

Despite 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.

User Authentication with Symmetric Encryption

A better solution is:

1. $A \rightarrow B$: $ID_A \| N_a$
2. $B \rightarrow KDC$: $ID_B \| N_b \| E(K_b, [ID_A \| N_a \| T_b])$
3. $KDC \rightarrow A$: $E(K_a, [ID_B \| N_a \| K_s \| T_b]) \| E(K_b, [ID_A \| K_s \| T_b]) \| N_b$
4. $A \rightarrow B$: $E(K_b, [ID_A \| K_s \| T_b]) \| E(K_s, N_b)$

expiration time

The fact that **B's nonce is encrypted with the session key** authenticates that the message came from A and is not a replay.

Furthermore, the protocol leaves A in possession of a token that can be used for subsequent authentication to B, avoiding the need to contact the KDC repeatedly.

1. $A \rightarrow B$: $E(K_b, [ID_A \| K_s \| T_b]) \| N'_a$
2. $B \rightarrow A$: $N'_b \| E(K_s, N'_a)$
3. $A \rightarrow B$: $E(K_s, N'_b)$

User Authentication with Asymmetric Encryption

A protocol using **timestamps**, with a central Authentication Server (AS):

1. $A \rightarrow AS: \quad ID_A \| ID_B$
2. $AS \rightarrow A: \quad E(PR_{as}, [ID_A \| PU_a \| T]) \| E(PR_{as}, [ID_B \| PU_b \| T])$
3. $A \rightarrow B: \quad E(PR_{as}, [ID_A \| PU_a \| T]) \| E(PR_{as}, [ID_B \| PU_b \| T]) \| E(PU_b, E(PR_a, [K_s \| T]))$

Here, K_s is created by A. The AS provides public-key certificates.

The timestamps protect against replays of compromised keys, but require the synchronization of clocks.

User Authentication with Asymmetric Encryption

Another protocol using **nonces** instead of timestamps, and involving a Key Distribution Center (KDC) that creates k_s:

1. A → KDC: $ID_A \| ID_B$
2. KDC → A: $E(PR_{auth}, [ID_B \| PU_b])$
3. A → B: $E(PU_b, [N_a \| ID_A])$
4. B → KDC: $ID_A \| ID_B \| E(PU_{auth}, N_a)$
5. KDC → B: $E(PR_{auth}, [ID_A \| PU_a]) \| E(PU_b, E(PR_{auth}, [N_a \| K_s \| ID_A \| ID_B]))$
6. B → A: $E(PU_a, [N_b \| E(PR_{auth}, [N_a \| K_s \| ID_A \| ID_B])])$
7. A → B: $E(K_s, N_b)$

PR_{auth} is the private key of the KDC

Federated Identity Management

Federated identity management is a relatively new concept dealing with the use of **a common identity management scheme across multiple enterprises and numerous applications** and supporting many thousands, even millions, of users.

Identity management = defining an **identity** for each user (human or process), associating **attributes** with the identity, and enforcing a means for identity **verification**.

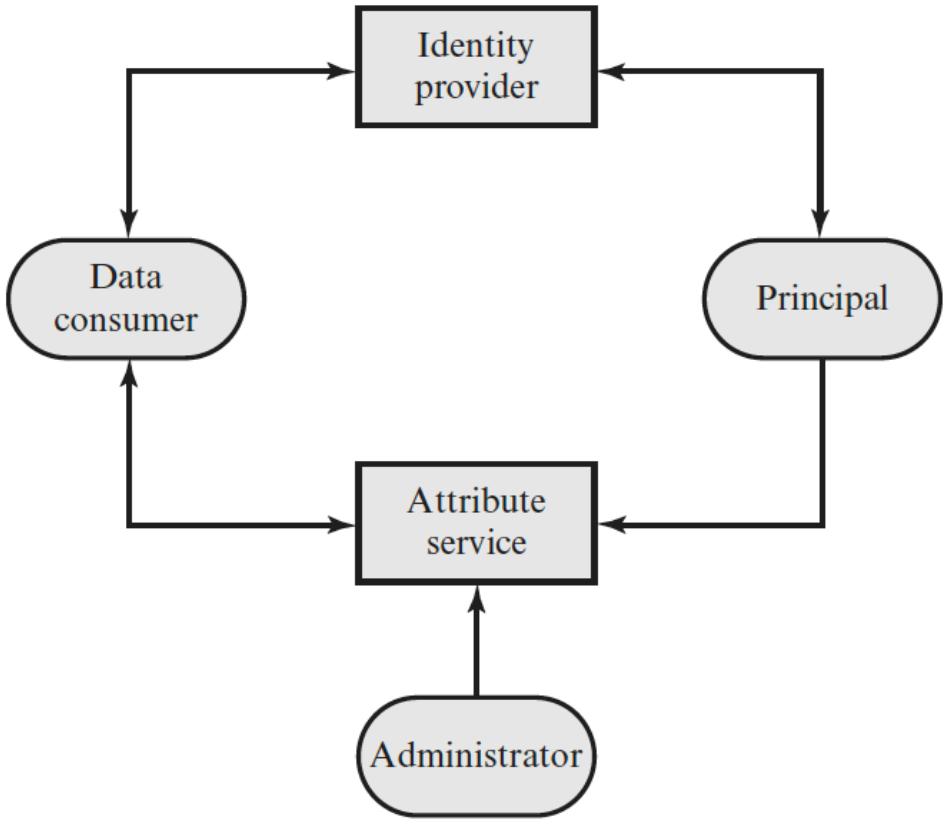
Single sign-on (SSO) enables a user to access all network resources after a single authentication.

Federated Identity Management

Services provided by a federated identity management system:

- **Point of contact**: authentication, user/server session management
- **SSO protocol services**: vendor-neutral security token
- **Trust services**: enforce trust relationships between partners
- **Key services**: keys and certificates
- **Identity services**: interfaces to local user registries and databases
- **Authorization**: granting access to specific resources based on authentication
- **Provisioning**: for creating accounts and defining access rights
- **Management**: for runtime configuration and deployment

Generic identity management architecture



Principal = identity holder: human user, device, agent process, server system

Identity provider: associates authentication information (such as passwords) with a principal, as well as attributes (e.g., shipping address)

Attribute service: manages the creation and maintenance of attributes

Administrator: assigns roles and access permissions to users

Data consumers: (e.g., e-commerce platform) obtain and employ data maintained by identity providers and attribute services

Identity Federation

Identity federation is identity management over multiple security domains.

The goal is to provide the sharing of digital identities so that a user can be authenticated a single time and then access applications and resources across multiple domains.

Because these domains are relatively autonomous or independent, no centralized control is possible. Rather, the cooperating organizations must form a federation based on **agreed standards** and **mutual levels of trust** to securely share digital identities and perform **identity mapping** (different security domains may represent identities and attributes differently).

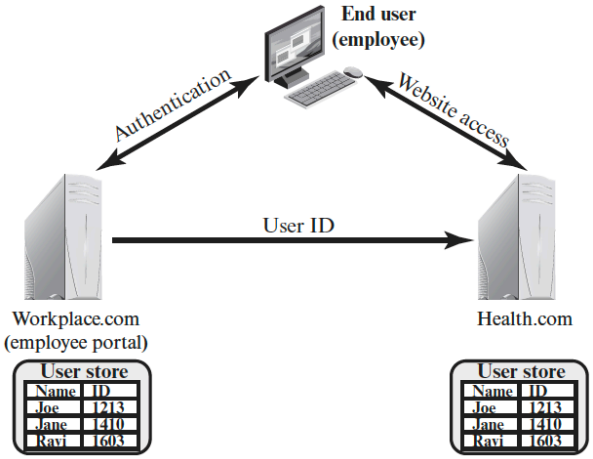
Identity Federation

When multiple organizations implement interoperable federated identity schemes, an employee in one organization can use SSO to access services across the federation with trust relationships associated with the identity.

Example

An employee may log onto her corporate intranet and be authenticated to perform authorized functions and access authorized services on that intranet.

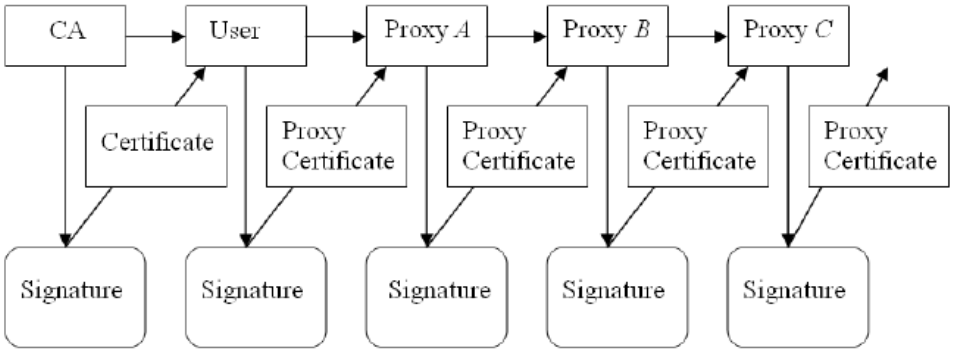
The employee could then access her health benefits from an outside health-care provider without having to re-authenticate.



X.509 Proxy Certificates

X.509 proxy certificates [RFC3820] solve two problems at the same time: **SSO** and **delegation**.

At each stage, there is a requester that generates a proxy certificate, signs it, and sends it to a service provider. The user can be always identified, by following backward the chain of signatures.



The longer the chain, the greater the risk that credentials are acquired and used by unauthorized third parties. For this reason, delegation operations must be carefully managed, for example by limiting delegated privileges and their duration.

References

William Stallings, ***Cryptography and Network Security - Principles and Practice***, 7th edition, Pearson 2017