Authentication

Entity authentication is a technique designed to let one party prove the identity of another party. An entity can be a person, a process, a client, or a server. The entity whose identity needs to be proved is called the claimant; the party that tries to prove the identity of the claimant is called the verifier. When Bob tries to prove the identity of Alice, Alice is the claimant, and Bob is the verifier.

Verification Categories

In entity authentication, the claimant must identify herself to the verifier. This can be done with one of three kinds of witnesses: something known, something possessed, or something inherent.

❏ **Something known**. This is a secret known only by the claimant that can be checked by the verifier. Examples are a password, a PIN, a secret key, and a private key.

❏ **Something possessed**. This is something that can prove the claimant's identity. Examples are a passport, a driver's license, an identification card, a credit card, and a smart card.

❏ **Something inherent**. This is an inherent characteristic of the claimant. Examples are conventional signatures, fingerprints, voice, facial characteristics, retinal pattern, and handwriting.

PASSWORDS

The simplest and oldest method of entity authentication is the **password-based authentication**, where the password is something that the **claimant** knows. A password is used when a user needs to access a system to use the system's resources (login). Each user has a user identification that is public, and a password that is private. We can divide these authentication schemes into two groups: the **fixed password** and the **one-time password**.

Fixed Password A fixed password is a password that is used over and over again for every access.

One-Time Password A one-time password is a password that is used only once.

BIOMETRICS

Biometrics is the measurement of physiological or behavioral features that identify a person (authentication by something inherent). Biometrics measures features that cannot be guessed, stolen, or shared.

Components

Several components are needed for biometrics, including capturing devices, processors, and storage devices. Capturing devices such as readers (or sensors) measure biometrics features. Processors change the measured features to the type of data appropriate for saving. Storage devices save the result of processing for authentication.

Enrollment

Before using any biometric techniques for authentication, the corresponding feature of each person in the community should be available in the database. This is referred to as enrollment.

Authentication

Authentication is done by verification or identification.

Verification

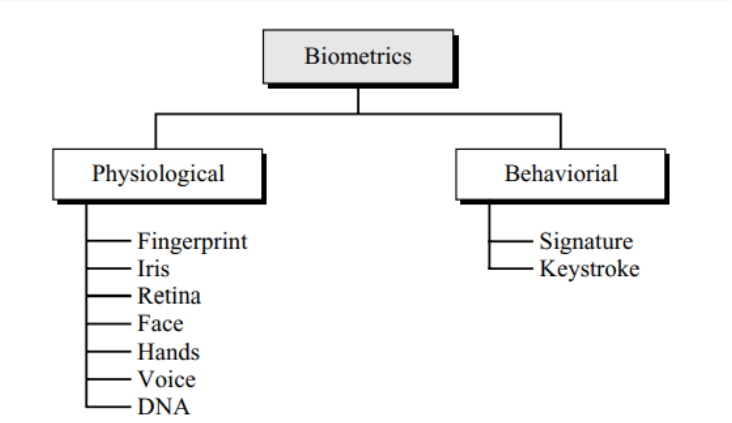
In verification, a person's feature is matched against a single record in the database (one-to-one matching) to find if she is who she is claiming to be. This is useful, for example, when a bank needs to verify a customer's signature on a check.

Identification

In identification, a person's feature is matched against all records in the database (one-to-many matching) to find if she has a record in the database. This is useful, for example, when a company needs to allow access to the building only to employees.

Techniques

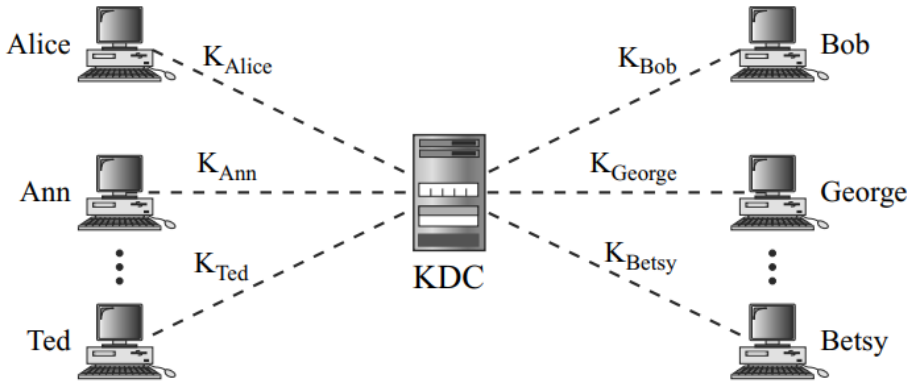
Biometrics techniques can be divided into two broad categories: physiological and behavioral.



Key-Distribution Center: KDC

It is an efficient way to maintain and distribute secret keys. A practical solution is the use of a trusted third party, referred to as a key-distribution center (KDC). To reduce the number of keys, each person establishes a shared secret key with the KDC.

Key-distribution center (KDC)



A secret key is established between the KDC and each member. Alice has a secret key with the KDC, which we refer to as KAlice; Bob has a secret key with the KDC, which we refer to as KBob; and so on. Now the question is how Alice can send a confidential message to Bob. The process is as follows:

1. Alice sends a request to the KDC stating that she needs a session (temporary) secret key between herself and Bob.
2. The KDC informs Bob about Alice's request.
3. If Bob agrees, a session key is created between the two.

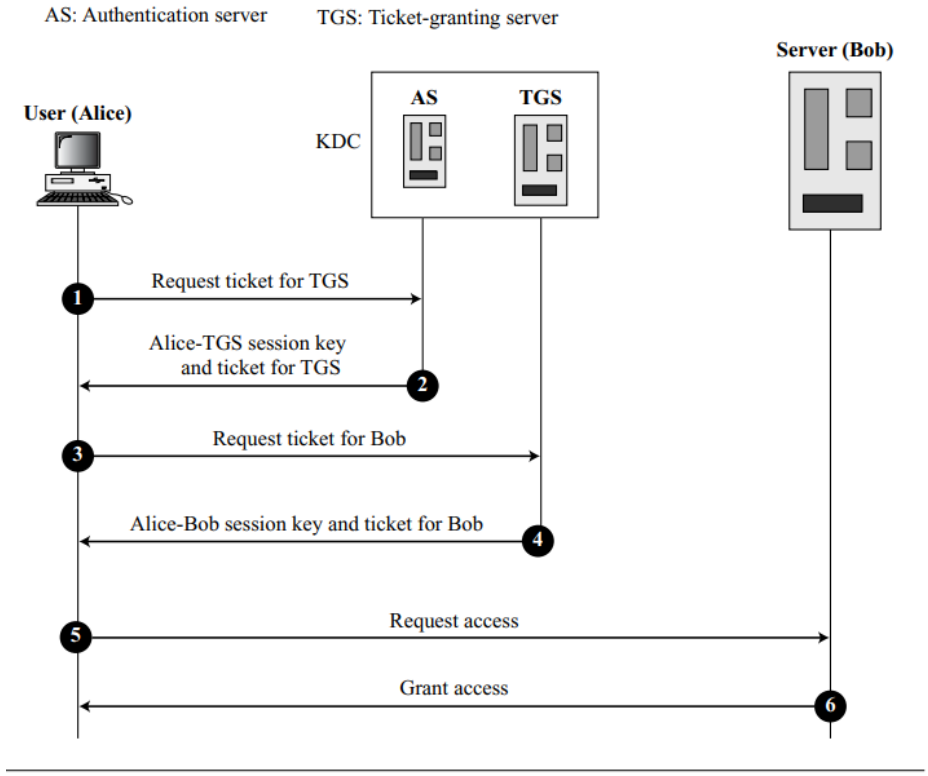
The secret key between Alice and Bob that is established with the KDC is used to authenticate Alice and Bob to the KDC and to prevent Eve from impersonating either of them.

KERBEROS

Kerberos is an authentication protocol, and at the same time a KDC, that has become very popular. Several systems, including Windows 2000, use Kerberos.

Servers

Three servers are involved in the Kerberos protocol: an authentication server (AS), a ticket-granting server (TGS), and a real (data) server that provides services to others.



Authentication Server (AS)

The authentication server (AS) is the KDC in the Kerberos protocol. Each user registers with the AS and is granted a user identity and a password. The AS has a database with these identities and the corresponding passwords. The AS verifies the user, issues a session key to be used between Alice and the TGS, and sends a ticket for the TGS.

Ticket-Granting Server (TGS)

The ticket-granting server (TGS) issues a ticket for the real server (Bob). It also provides the session key (KAB) between Alice and Bob. Kerberos has separated user verification from the issuing of tickets. In this way, though Alice verifies her ID just once with the AS, she can contact the TGS multiple times to obtain tickets for different real servers.

Real Server

The real server (Bob) provides services for the user (Alice). Kerberos is designed for a client-server program, such as FTP, in which a user uses the client process to access the server process. Kerberos is not used for person-to-person authentication.

Certificate-Based Authentication

Certificate-based authentication (CBA) uses a digital certificate derived from cryptography to identify a user, device or machine, before granting access to an application, network or other resource.

A digital identity certificate is an electronic document used to prove private key ownership. Certificate-based authentication uses the information within said document to verify the user, device or machine, in contrast to the classic username and password combination which is strictly limited to verifying only those who are in possession, i.e. potentially not just the user who should have access. The increased sophistication of attacks, coupled with the exploding number of attack surfaces as more and more devices access the network, make this additional capability more necessary than ever.

A digital certificate contains a number of important details, depending on the standard in use. For example, the popular X.509 certificate includes the below elements:

- The public key
- The user or device's name
- The name of the Certificate Authority (CA) that issued the certificate
- The date from which the certificate is valid
- The expiry date of the certificate
- The version number of the certificate data
- A serial number

When implemented correctly, digital certificate-based authentication makes it possible for security administrators to issue digital certificates to new users, renew certificates that have expired for existing users, and finally, revoke certificates from users who no longer should have access.

A certificate-based authentication server usually follows some variation of the below process in order to validate a client request:

- The server checks that the current date is valid, and the certificate has not expired
- The server sends random bits of data, also known as a nonce, to be signed by the requesting device
- The nonce is signed by the requesting device using its private key, and is returned to the server alongside its certificate

- The server receives both the certificate and the signed nonce, and then validates the user’s identity by signing the original nonce itself using the public key stated within the certificate, and comparing that result with the received signed nonce
- If the results match, the user is authenticated and granted access

Single sign-on (SSO)

Single sign-on (SSO) is a technology which combines several different application login screens into one. With SSO, a user only has to enter their login credentials (username, password, etc.) one time on a single page to access all of their SaaS applications.

SSO is often used in a business context, when user applications are assigned and managed by an internal IT team. Remote workers who use SaaS applications also benefit from using SSO.

Imagine if customers who had already been admitted to a bar were asked to show their identification card to prove their age each time they attempted to purchase additional alcoholic beverages. Some customers would quickly become frustrated with the continual checks and might even attempt to circumvent these measures by sneaking in their own beverages.

However, most establishments will only check a customer's identification once, and then serve the customer several drinks over the course of an evening. This is somewhat like an SSO system: instead of establishing their identity over and over, a user establishes their identity once and can then access several different services.

How does an SSO login work?

Whenever a user signs in to an SSO service, the service creates an authentication token that remembers that the user is verified. An authentication token is a piece of digital information stored either in the user's browser or within the SSO service's servers, like a temporary ID card issued to the user. Any app the user accesses will check with the SSO service. The SSO service passes the user's authentication token to the app and the user is allowed in. If, however, the user has not yet signed in, they will be prompted to do so through the SSO service.

An SSO service does not necessarily remember who a user is, since it does not store user identities. Most SSO services work by checking user credentials against a separate identity management service.

Think of SSO as a go-between that can confirm whether a user's login credentials match with their identity in the database, without managing the database themselves — somewhat like when a librarian looks up a book on someone else's behalf based on the title of the book. The librarian does not have the entire library card catalog memorized, but they can access it easily.

Security Handshake Pitfalls

http://gauss.ececs.uc.edu/Courses/c653/lectures/PDF/pitfalls.pdf