Authentication 10

10.1 Definition

Authentication is the process of validating the identity of someone or something. It uses information provided to the authenticator to determine whether someone or something is in fact who or what it is declared to be. In private and public computing systems, for example, in computer networks, the process of authentication commonly involves someone, usually the user, using a password provided by the system administrator to *logon*. The user's possession of a password is meant to guarantee that the user is authentic. It means that at some previous time, the user requested, from the system administrator, and the administrator assigned and/or registered a self-selected password.

The user presents this password to the logon to prove that he or she knows something no one else could know.

Generally, authentication requires the presentation of credentials or items of value to really prove the claim of who you are. The items of value or credential are based on several unique factors that show something you know, something you have, or something you are [1]:

- *Something you know*: This may be something you mentally possess. This could be a password, a secret word known by the user and the authenticator. Although this is inexpensive administratively, it is prone to people's memory lapses and other weaknesses including secure storage of the password files by the system administrators. The user may use the same password on all system logons or may change it periodically, which is recommended. Examples using this factor include passwords, passphrases, and personal identification numbers (PINs).
- *Something you have*: This may be any form of issued or acquired self-identification such as:
  - SecurID
  - CryptoCard
  - ActivCard

J.M. Kizza, *Guide to Computer Network Security*, Computer Communications and Networks, DOI 10.1007/978-3-319-55606-2_10

- SafeWord
- Many other forms of cards and tags

This form is slightly safer than something you know because it is hard to abuse individual physical identifications. For example, it is harder to lose a smart card than to remember the card number.

- *Something you are*: This is a naturally acquired physical characteristic such as voice, fingerprint, iris pattern, and other biometrics discussed in Chap. 7. Although biometrics are very easy to use, this ease of use can be offset by the expenses of purchasing biometric readers. Examples of items used in this factor include fingerprints, retinal patterns, DNA patterns, and hand geometry.

In addition to the top three factors, another factor, though indirect, also plays a part in authentication:

- *Somewhere you are*: This usually is based on either physical or logical location of the user. The use, for example, may be on a terminal that can be used to access certain resources.

In general, authentication takes one of the following three forms [2]:

- *Basic authentication* involving a server. The server maintains a user file of either passwords and usernames or some other useful piece of authenticating information. This information is always examined before authorization is granted. This is the most common way computer network systems authenticate users. It has several weaknesses though, including forgetting and misplacing authenticating information such as passwords.
- *Challenge-response*, in which the server or any other authenticating system generates a challenge to the host requesting for authentication and expects a response. We will discuss challenge-response in Sect. 10.5.1.3.
- *Centralized authentication*, in which a central server authenticates users on the network and in addition also authorizes and audits them. These three processes are done based on server action. If the authentication process is successful, the client seeking authentication is then authorized to use the requested system resources. However, if the authentication process fails, the authorization is denied. The process of auditing is done by the server to record all information from these activities and store it for future use.

10.2 Multiple Factors and Effectiveness of Authentication

For an authentication mechanism to be considered effective, it must uniquely and in a forgery-proof manner identify an individual. The factors above do so in varying degrees depending on how they are combined. Each factor, if used alone to

authenticate users, is effective enough to authenticate a user; however, these systems' authentication may be more vulnerable to compromise of the authenticator. For example, both factors "authentication by knowledge" and "authentication by ownership" in factors 1 and 2 above require a person to be associated with something by knowledge or acquisition.

Notice that the user is not required to be physically attached to the authentication information. Possession of something that is not physically attached to the user can result in that authentication information getting lost, stolen, or otherwise compromised. For example, information by knowledge can be duplicated through user negligence or somebody else learning it without the user knowing. It can also be acquired through possible guessing, repeated attempts, or through brute force by using automated mathematical exhaustive search techniques.

Similarly "authentication by ownership" suffers from a set of problems that make it not so effective. For example, although items in this category have their major strength in the difficulty of duplication, such objects also require more effort to guard from theft; they can be made using special equipment or procedures [3].

Although the third factor, "authentication by characteristic," is much stronger than the first two, it suffers from high costs incurred to acquire and build effective peripherals that can obtain a complete enough sample of a characteristic to entirely distinguish one individual from another. It requires readers with more advanced features and functions to read, analyze, and completely discriminate and distinguish one's physical features. Readers with these functions are often very expensive and require highly trained personnel and other operating expenses.

As the Internet becomes widely used in everyday transactions including e-commerce, a stronger form of authentication that differs from the traditional username password authentication is needed to safeguard system resources from the potentially hostile environment of the "bad" Internet. The "bad" Internet consists of wide array of "untrusted" public and private clients, including civic networks and public kiosks and cafes. In addition to these, it also includes commonly available software that allows an intruder to easily sniff, snoop, and steal network logon passwords as they are exchanged in the traditional authentication schemes.

To address this, an effective authentication scheme with multiple methods is preferred. Systems using two or more methods can result in greater system security.

This process of piggybacking authentication factors is one of the popular strategies now used widely for overcoming the limitations of a specific authentication factor by supplementing it with another factor. This technique of improving authentication assurance is referred to as *multifactor* authentication.

Although it is common to combine two or more authentication items from two or more factors as shown in Fig. 10.1, it is also possible to combine two or more items from the same authentication factor class. For example, one can combine an iris pattern and a fingerprint. There are generally two motives for taking this action [4]:

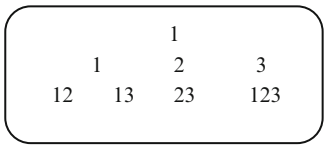

Fig. 10.1 Authentication factor combinations

- The need to improve usability and accuracy. Combining items from different authenticating factors improves the accuracy of the authentication process. It may also lead to reduction in the false rejection rate of legitimate users.
- To improve the authentication process' integrity by reducing the effect of certain items in some factors that are prone to vulnerabilities that weaken it. The combining technique, therefore, reduces the risk of false negatives where, for example, an impersonating user can succeed in accessing the system.

The discussion above provides one very important element of authentication: that different mechanisms provide different levels of authentication effectiveness. Choosing the most effective authentication, therefore, depends on the technology used and also on the degree of *trust* placed on that technology. Generally, trust is a firm belief or confidence one has in someone or something. Trust is manifested in attributes such as honesty, reliability, integrity, justice, and others. Since authorization comes after approval of identity, that is, after authentication, a organizational framework spelling out an authorization policy based on authentication is a *trust model*. Organizations use trust model to create authentication groups. For example, a group of company executives may be put in a different authentication process than a group consisting of parking attendants. These authentication and authorization groupings are based on the company's trust model.

10.3 Authentication Elements

An authentication process as described above is based on five different elements: the person or group of people seeking authentication, distinguishing characteristics from that person or group presented for authentication, the authenticator, the authenticating mechanism to verify the presence of the authenticating characteristics, and the access control mechanism to accept or deny authentication.

10.3.1 Person or Group Seeking Authentication

These are usually users who seek access to a system either individually or as a group. If individually, they must be prepared to present to the authenticator evidence to support the claim that they are actually authorized to use the requested system resource. They may present any one of the basic factors discussed in Sect. 10.1. Similarly as a group, the group again must present to the authenticator

evidence that any one member of the group is authorized to use the system based on a trust model.

10.3.2 Distinguishing Characteristics for Authentication

The second authentication element is the distinguishing characteristics from the user to the authenticator. In Sect. 10.1, we already discussed these characteristics and grouped them into four factors that include something you know, something you have, something you are, and a weaker one somewhere you are. In each of these factors, there are items that a user can present to the authenticator for authorization to use the system. Some of these items may not completely authenticate the user, and we have pointed out in Sect. 10.2 that a combination of items from different factors and trust may be used to strengthen the authentication and create better assurances.

10.3.3 The Authenticator

The job of the authenticator is to positively and sometimes automatically identify the user and indicate whether that user is authorized to access the requested system resource. The authenticator achieves application for authentication by prompting for user credentials when an authentication request is issued. The authenticator then collects the information and passes it over to the authentication mechanism.

The authenticator can be a user-designated server, a virtual private network (VPN), firewall, a local area network (LAN) server, an enterprise-wide dedicated server, independent authentication service, or some other form of global identity service. Whatever is being used as an authenticator must perform an authentication process that must result in some outcome value such as a token that is used in the authentication process to determine information about the authenticated user at a later time. A note of caution to the reader is that some authors call this token the authenticator. Because there is no standard on these tokens adhered to by all authenticating schemes, the format of the token varies from vendor to vendor.

10.3.4 The Authentication Mechanism

The authentication mechanism consists of three parts that work together to verify the presence of the authenticating characteristics provided by the user. The three parts are the input, the transportation system, and the verifier. They are linked with the appropriate technologies. An input component acts as the interface between the user and the authentication system. In a distributed environment, this could be a computer keyboard, card reader, video camera, telephone, or similar device. The captured user-identifying items need to be taken to a place where they are scrutinized, analyzed, and accepted or rejected. But in order for these items to

reach this point, they have to be transported. The transport portion of the system is, therefore, responsible for passing data between the input component and the element that can confirm a person's identity. In modern-day authenticating systems, this information is transported over a network, where it can be protected by protocols like Kerberos or sent in plaintext [4].

The last component of the authentication system is the verification component, which is actually the access control mechanism in the next section.

10.3.5 Access Control Mechanism

We discussed access control and the working of the access control mechanism in Chap. 8. Let us briefly review the role of the access control mechanism in the authentication process. User-identifying and authenticating information is passed to access control from the transport component. Here, this information must be validated against the information in its database. The database may reside on a dedicated authentication server or may be stored in a file on a local medium. The access control mechanism then cross-checks the two pieces of information for a match. If a match is detected, the access control system then issues temporary credentials authorizing the user to access the desired system resource.

10.4 Types of Authentication

In Sect. 10.1, we identified three factors that are used in the positive authentication of a user. We also pointed out in the previous section that while these factors are in themselves good, there are items in some that suffer from vulnerabilities. Table 10.1 illustrates the shortcomings of user identity characteristics from the factors that suffer from these vulnerabilities.

From Table 10.1, one can put the factors into two categories: nonrepudiable and repudiable authentication. Other types of authentication include user, client, and session authentication.

Table 10.1 Authentication factors and their vulnerabilities[a]

Number	Factor	Examples	Vulnerabilities
1	What you know	Password, PIN	Can be forgotten, guessed, duplicated
2	What you have	Token, ID card, keys	Can be lost, stolen, duplicated
3	What you are	Iris, voiceprint, fingerprint	Nonrepudiable

[a]Ratha, Nalini K., Jonathan H. Connell and Ruud M. Bolle. "Secure Fingerprint-based Authentication for Lotus Notes." https://faculty.unlv.edu/thatcher/is485/readings/biometrics.pdf

10.4.1 Nonrepudiable Authentication

Nonrepudiable authentication involves all items in factor 3. Recall that factor 3 consists of items that involve some type of characteristics and whose proof of origin cannot be denied. The biometrics used in factor 3, which include iris patterns, retinal images, and hand geometry, have these characteristics. Biometrics can positively verify the identity of the individual. In our discussion of biometrics in Chap. 8, we pointed out that biometric characteristics cannot be forgotten, lost, stolen, guessed, or modified by an intruder. They, therefore, present a very reliable form of access control and authorization. It is also important to note that contemporary applications of biometric authorization are automated, which further eliminates human errors in verification. As technology improves and our understanding of the human anatomy increases, newer and more sensitive and accurate biometrics will be developed.

Next to biometrics as nonrepudiable authentication items are *undeniable and confirmer digital signatures*. These signatures, developed by Chaum and van Antwerpen, are signatures that cannot be verified without the help of a signer and cannot with non-negligible probability be denied by the signer. Signer legitimacy is established through a confirmation or denial protocol [5]. Many undeniable digital signatures are based on Rivest, Shamir, and Adleman (RSA) structure and technology, which give them provable security that makes the forgery of undeniable signatures as hard as forging standard RSA signatures.

Confirmer signatures [6, 7] are a type of undeniable signatures, where signatures may also be further verified by an entity called the confirmer designated by the signer.

Lastly, there are *chameleon signatures*, a type of undeniable signatures in which the validity of the content is based on the trust of the signer's commitment to the contents of the signed document. But in addition, they do not allow the recipient of the signature to disclose the contents of the signed information to any third party without the signer's consent [5].

10.4.2 Repudiable Authentication

In our discussion of authentication factors in Sect. 10.2, we pointed out that the first two factors, "what you know" and "what you have," are factors that can present problems to the authenticator because the information presented can be unreliable. It can be unreliable because such factors suffer from several well-known problems including the fact that possessions can be lost, forged, or easily duplicated. Also knowledge can be forgotten and taken together, and knowledge and possessions can be shared or stolen. Repudiation is, therefore, easy. Before the development of items in factor 3, in particular the biometrics, authorization, and authentication methods relied only on possessions and knowledge.

10.5 Authentication Methods

Different authentication methods are used based on different authentication algorithms. These authentication methods can be combined or used separately, depending on the level of functionality and security needed. Among such methods are password authentication, public key authentication, anonymous authentication, and remote and certificate-based authentication.

10.5.1 Password Authentication

The password authentication methods are the oldest and the easiest to implement. They are usually set up by default in many systems. Sometimes, these methods can be interactive using the newer keyboard-interactive authentication. Password authentication includes reusable passwords, one-time passwords, challenge-response passwords, and combined approach passwords.

10.5.1.1 Reusable Passwords

There are two types of authentication in reusable password authentication (user and client authentication):

- *User authentication*. This is the most commonly used type of authentication, and it is probably the most familiar to most users. It is always initiated by the user, who sends a request to the server for authentication and authorization for the use of a specified system resource. On receipt of the request, the server prompts the user for a username and password. On submission of these, the server checks for a match against copies in its database. Based on the match, authorization is granted.
- *Client authentication*. Normally, the user requests for authentication and then authorization by the server to use a system or a specified number of system resources. Authenticating users does not mean the user is free to use any system resource the user wants. Authentication must establish user authorization to use the requested resources in the amount requested and no more. This type of authentication is called client authentication. It establishes users' identities and controlled access to system resources.

Because these types of authentication are the most widely used authentication methods, they are the most abused. They are also very unreliable because users forget them, they write them down, they let others use them, and, most importantly, they are easy to guess because users choose simple passwords. They are also susceptible to cracking and snooping. In addition, they fall prey to today's powerful computers, which can crack them with brute force through exhaustive search.

10.5.1.2 One-Time Passwords

One-time password authentication is also known as session authentication. Unlike reusable passwords that can be used over extended periods of time, one-time passwords are used once and disposed of. They are randomly generated using powerful random number generators. This reduces the chances of their being guessed. In many cases they are encrypted and then issued to reduce their being intercepted if they are sent in the clear. There are several schemes of one-time passwords. The most common of these schemes are S/Key and token:

- *S/Key password* is a one-time password generation scheme defined in RFC 1760 and is based on MD4 and MD5 encryption algorithms. It was designed to fight against replay attacks where, for example, in a log-in session, an intruder eavesdrops on the network log-in session and gets the password and user ID for the legitimate user. Its protocol is based on a client-server model in which the client initiates the S/Key exchange by sending the first packet to which the server responds with an ACK and a sequence number. Refer to Chap. 1 for this. The client then responds to the server by generating a one-time password and passes it to the server for verification. The server verifies the password by passing it through a hash function and compares the hash digest to the stored value for a match.
- *Token password* is a password generation scheme that requires the use of a special card such as a smart card. According to Kaeo, the scheme is based on two schemes: challenge-response and time-synchronous [8]. We are going to discuss challenge-response in Sect. 10.5.1.3. In a time-synchronous scheme, an algorithm executes both in the token and on the server, and outputs are compared for a match. These numbers, however, change with time.

Although they are generally safer, one-time passwords have several difficulties including synchronization problems that may be caused by lapsed time between the time stamp in the password and the system time. Once these two times are out of phase, the password cannot be used. Also synchronization problems may arise when the one-time password is issued based on either a system or user. If it is based on the user, the user must be contacted before use to activate the password.

10.5.1.3 Challenge-Response Passwords

In Sect. 10.1, we briefly talked about challenge-response authentication as another form of relatively common form of authentication. Challenge-response, as a password authentication process, is a handshake authentication process in which the authenticator issues a challenge to the user seeking authentication. The user must provide a correct response in order to be authenticated. The challenge may take many forms depending on the system. In some systems, it is in the form of a message indicating "unauthorized access" and requesting a password. In other systems, it may be a simple request for a password, a number, a digest, or a nonce (a server-specified data string that may be uniquely generated each time a server generates a 401 server error). The person seeking authentication must

respond to the system challenge. Nowadays, responses are by a one-way function using a password token, commonly referred to as *asynchronous tokens*. When the server receives the user response, it checks to be sure the password is correct. If so, the user is authenticated. If not or if for any other reason the network does not want to accept the password, the request is denied.

Challenge-response authentication is used mostly in distributed systems. Though becoming popular, challenge-response authentication is facing challenges as a result of weaknesses that include user interaction and trial-and-error attacks. The problem with user interaction involves the ability of the user to locate the challenge over usually clattered screens. The user then must quickly type in a response. If a longer than anticipated time elapses, the request may be denied. Based on the degree of security needed, sometimes the user has to remember the long response or sometimes is forced to write it down, and finally the user must transcribe the response and type it in. This is potentially error prone. Some vendors have tried to cushion the user from remembering and typing long strings by automating most of the process either by cut and paste of the challenge and responses or through a low-level automated process where the user response is limited to minimum yes/no responses.

In trial-and-error attacks, the intruders may respond to the challenge with a spirited barrage of trial responses hoping to hit the correct response. With powerful computers set to automatically generate responses in a given time frame, it is potentially possible for the intruder to hit on a correct response within the given time frame.

Also of interest is to remember that in its simplest form, challenge-responses that use passwords can be abused because passwords are comparatively easy to steal. And if transmitted in the clear, passwords can also be intercepted. However, this situation is slightly better in the nonce or digest authentication, the more sophisticated of the two forms of scheme, because the password is not sent in the clear over the network. It is encrypted which enhances security, although not fully hack-proof protection.

10.5.1.4 Combined Approach Authentication

Although basic authentication which uses either names or names and passwords is the most widely used authentication scheme, it is prudent not to rely on just basic authentication. Passwords are often transmitted in the clear from the user to the authentication agent, which leaves the passwords open to interception by hackers. To enhance the security of authentication, it is better sometimes to combine several schemes. One of the most secure authentication methods is to use a random challenge-response exchange using digital signatures. When the user attempts to make a connection, the authentication system, a server or a firewall, sends a random string back as a challenge. The random string is signed using the user's private key and sent back as a response. The authenticating server or firewall can then use the user's public key to verify that the user is indeed the holder of the associated private key [9].

10.5.2 Public Key Authentication

As we discussed in Sect. 2.3.2 and we will later see in the next chapter, the process of public key authentication requires each user of the scheme to first generate a pair of keys and store each in a file. Each key is usually between 1024 and 2048 bits in length. Public-private key pairs are typically created using a key generation utility. As we will discuss in the next chapter, the pair will consist of a user's public and private key pair. The server knows the user's public key because it is published widely. However, only the user has the private key.

Public key systems are used by authentication systems to enhance system security. The centralized authentication server, commonly known as the *access control server* (ACS), is in charge of authentication that uses public key systems. When a user tries to access an ACS, it looks up the user's public keys and uses it to send a challenge to the user. The server expects a response to the challenge where the user must use his or her private key. If the user then signs the response using his or her private key, he or she is authenticated as legitimate.

To enhance public key security, the private key never leaves the user's machine and, therefore, cannot be stolen or guessed like a password can. In addition, the private key has a *passphrase* associated with it; so even if the private key is stolen, the attacker must still guess the passphrase in order to gain access. The ACS is used in several authentication schemes including SSL, Kerberos, and MD5 authentication.

10.5.2.1 Secure Sockets Layer (SSL) Authentication

Secure Sockets Layer (SSL) is an industry standard protocol designed by Netscape Communications Corporation for securing network connections. SSL provides authentication, encryption, and data integrity using public key infrastructure (PKI). SSL authentication, being cryptographic-based, uses a public/private key pair that must be generated before the process can begin. Communicating elements acquire verification certificates from a *certificate authority* (CA).

> A certificate authority is a trusted third party, between any two communicating elements such as network servers, that certifies that the other two or more entities involved in the intercommunication, including individual users, databases, administrators, clients, servers, are who they say they are. The certificate authority certifies each user by verifying each user's identity and grants a certificate, signing it with the certificate authority's private key. Upon verification, the certificate authority then publishes its own certificate which includes its public key. Each network entity, server, database, and others gets a list of certificates from all the trusted CAs and it consults this list every time there is a communicating user entity that needs authentication. With the CA's issued certificate, the CA guarantees that anything digitally signed using that certificate is legal. As we will see in the next chapter, sometimes it is possible to also get a private key along with the certificate, if the user does not want to generate the corresponding private key from the certificate. As e-commerce picks up momentum, there is an increasing need for a number of creditable companies to sign up as CAs. And indeed many are signing up. If the trend continues, it is likely that the use of digital certificates issued and verified by a CA as part of a public key infrastructure (PKI) is likely to become a standard for future e-commerce.

These certificates are signed by calculating a checksum over the certificate and encrypting the checksum and other information using the private key of a signing certificate. User certificates can be created and signed by a signing certificate which can be used in the SSL protocol for authentication purposes. The following steps are needed for an SSL authentication [10]:

- The user initiates a connection to the server by using SSL.
- SSL performs the handshake between client and server.
- If the handshake is successful, the server verifies that the user has the appropriate authorization to access the resource.

The SSL handshake consists of the following steps [10]:

- The client and server establish which authenticating algorithm to use.
- The server sends its certificate to the client. The client verifies that the server's certificate was signed by a trusted CA. Similarly, if client authentication is required, the client sends its own certificate to the server. The server verifies that the client's certificate was signed by a trusted CA.
- The client and server exchange key material using public key cryptography (see more of this in the next chapter), and from this material, they each generate a session key. All subsequent communication between client and server is encrypted and decrypted by using this set of session keys and the negotiated cipher suite.

It is also possible to authenticate using a two-way SSL authentication, a form of mutual authentication. In two-way SSL authentication, both the client and server must present a certificate before the connection is established between them.

10.5.2.2 Kerberos Authentication

Kerberos is a network authentication protocol developed at the Massachusetts Institute of Technology (MIT) and designed to provide strong authentication for client/ server applications by using PKI technology. See RFC 1510 for more details on Kerberos. It was designed to authenticate users' requests to the server.

In his paper "The Moron's Guide to Kerberos," Brian Tung, using satire, compares the authentication by Kerberos to that of an individual using a driver's license issued by the Department of Motor Vehicles (DMV). He observes that in each case, personal identity consists of a name and an address and some other information, such as a birth date. In addition, there may be some restrictions on what the named person can do; for instance, he or she may be required to wear corrective lenses while driving. Finally, the identification has a limited lifetime, represented by the expiration date on the card.

He compares this real-life case to the working of Kerberos. Kerberos typically is used when a user on a network is attempting to make use of a network service and the service wants assurance that the user is who he says he is. To that end, just like a merchant would want you to present your driver's license issued by the DMV

before he or she issues you with a ticket for the needed service, the Kerberos user gets a ticket that is issued by the Kerberos *authentication server* (AS). The service then examines the ticket to verify the identity of the user. If all checks out, then the user is issued an access ticket [11].

According to Barkley [12], there are five players involved in the Kerberos authentication process: the user, the client who acts on behalf of the user, the key distribution center, the ticket-granting service, and the server providing the requested service. The role of the key distribution center, as we will see in the coming chapter and also Chap. 16, is to play a trusted third party between the two communicating elements, the client and the server. The server, commonly known as the "Kerberos server," is actually the *key distribution center* or the KDC for short. The KDC implements the authentication service (AS) and the ticket-granting service (TGS).

When a user wants a service, the user provides the client with a password. The client then talks to the authentication service to get a *ticket-granting ticket*. This ticket is encrypted with the user's password or with a session key provided by the AS. The client then uses this ticket to talk to the ticket-granting service to verify the user's identity using the ticket-granting ticket. The TGS then issues a ticket for the desired service.

The ticket consists of the:

- Requested server name
- Client name
- Address of the client
- Time the ticket was issued
- Lifetime of the ticket
- Session key to be used between the client and the server
- Some other fields

The ticket is encrypted using the server's secret key and thus cannot be correctly decrypted by the user.

In addition to the ticket, the user must also present to the server an authenticator which consists of the:

- Client name
- Address
- Current time
- Some other fields

The authenticator is encrypted by the client using the session key shared with the server. The authenticator provides a time validation for the credentials.

A user seeking server authentication must then present to the server both the ticket and the authenticator. If the server can properly decrypt both the ticket, when it is presented by the client, and the client's authenticator encrypted using the session key contained in the ticket, the server can have confidence that the user is who he claims to be [12].

The KDC has a copy of every password and/or secret key associated with every user and server, and it issues ticket-granting tickets so users do not have to enter in their passwords every time they wish to connect to a kerberized service or keep a copy of their password around. If the ticket-granting ticket is compromised, an attacker can only masquerade as a user until the ticket expires [13].

Since the KDC stores all user and server secret keys and passwords, it must be well secured and must have stringent access control mechanism. If the secret key database is penetrated, a great deal of damage can occur.

10.5.2.3 MD5 for Authentication

In the previous chapter, we discussed MD5 as one of the standard encryption algorithms in use today. Beyond encryption, MD5 can be used in authentication. In fact, the authentication process using MD5 is very simple. Each user has a file containing a set of keys that are used as input into an MD5 hash. The information being supplied to the authenticating server, such as passwords, has its MD5 checksum calculated using these keys and is then transferred to the authenticating server along with the MD5 hash result. The authenticating server then gets user identity information such as password, obtains the user's set of keys from a key file, and then calculates the MD5 hash value. If the two are in agreement, authentication is successful [11].

10.5.3 Remote Authentication

Remote authentication is used to authenticate users who dial into the ACS from a remote host. This can be done in several ways, including using Secure Remote Procedure Call (RPC), dial-up, and Remote Authentication Dial-In User Service (RADIUS) authentication.

10.5.3.1 Secure RPC Authentication

There are many services, especially Internet services, in which the client may not want to identify itself to the server, and the server may not require any identification from the client. Services falling in this category, like the Network File System (NFS), require stronger security than the other services. Remote Procedure Call (RPC) authentication provides that degree of security. Since the RPC authentication subsystem package is open-ended, different forms and multiple types of authentication can be used by RPC including:

- NULL authentication
- Unix authentication
- Data Encryption Standard (DES) authentication
- DES authentication protocol
- Diffie-Hellman encryption

Servers providing the call services require that users be authenticated for every RPC call keys to servers and clients using any encryption standard.

10.5.3.2 Dial-In Authentication

As in remote calls, passwords are required in dial-in connections. Point-to-point protocol (PPP) is the most common of all dial-in connections, usually over serial lines or ISDN. An authentication process must precede any successful log-in. Dial-in authentication services authenticate the peer device, not the user of the device. There are several dial-in authentication mechanisms. PPP authentication mechanisms, for example, include the Password Authentication Protocol (PAP), the Challenge-Handshake Authentication Protocol (CHAP), and the Extensible Authentication Protocol (EAP) [8]:

- The PAP authentication protocol allows the peer to establish identity to the authenticator in a two-way handshake to establish the link. The link is used to send to the authenticator an initial packet containing the peer name and password. The authenticator responds with authenticate-ACK if everything checks out and the authentication process is complete. PAP is a simple authentication process that sends the peer authentication credentials to the authenticator in the clear, where they can be intercepted by the eavesdropper.
- The CHAP authentication protocol is employed periodically to verify any user who uses a three-way handshake. Like PAP, it uses the handshake to initialize a link. After establishing the link, CHAP requires the peer seeking authentication and the authenticator share a secret text that is never actually sent over the links. The secret is established through a challenge-response. The authenticator first sends a challenge consisting of an identifier, a random number, and a host name of the peer or user. The peer responds to the challenge by using a one-way hash to calculate a value; the secret is the input to the hash. The peer then sends to the authenticator an encrypted identification, the output of the hash, the random number, and the peer name or username. The authenticator verifies these by performing the same encryption and authenticates the peer, if everything checks out. It is possible for a relay attack on a CHAP authentication. So steps must be taken to safeguard the passing of the passwords.
- Extensible protocol supports multiple authentication mechanisms. Like all other PPP authentication mechanisms, a link is first established. The authenticator then first sends a request or requests, with a type field to indicate what is being requested, to the peer seeking authentication. The peer responds with a packet, with a type field, as requested. The authenticator then verifies the content of the packet and grants or denies authentication. EAP is more flexible as it provides a capability for new technologies to be tried.

10.5.3.3 Radius

Remote authentication Dial-in User Service (RADIUS) is a common user protocol that provides user dial-up to the ACS which does the user authentication. Because all information from the remote host travels in the clear, RADIUS is considered to be vulnerable to attacks and therefore not secure. We will discuss RADIUS in detail in Chap. 17.

10.5.4 Anonymous Authentication

Not all users who seek authentication to use system resources always want to use operations that modify entries or access protected attributes or entries that generally require client authentication. Clients who do not intend to perform any of these operations typically use anonymous authentication. Mostly these users are not indigenous users in a sense that they do not have membership to the system they want access to. In order to give them access to some system resources, for example, to a company Web site, these users, usually customers, are given access to the resources via a special "anonymous" account. System services that are used by many users who are not indigenous, such as the World Wide Web service or the FTP service, must include an anonymous account to process anonymous requests. For example, Windows Internet Information Services (IIS) creates the anonymous account for Web services, IUSR_ *machinename,* during its setup. By default, all Web client requests use this account, and clients are given access to Web content when they use it. You can enable both anonymous logon access and authenticated access at the same time [14].

10.5.5 Digital Signature-Based Authentication

Digital signature-based authentication is yet another authentication technique that does not require passwords and usernames. A *digital signature* is a cryptographic scheme used by the message recipient and any third party to verify the sender's identity and/or message on authenticity. It consists of an electronic signature that uses public key infrastructure (PKI) to verify the identity of the sender of a message or of the signer of a document. The scheme may include a number of algorithms and functions including the Digital Signature Algorithm (DSA), Elliptic Curve Digital Signature and Algorithm (ECDSA), account authority digital signature, authentication function, and signing function [6, 7].

The idea of a digital signature is basically the same as that of a handwritten signature, to authenticate the signer. It is used to authenticate the fact that what has been promised by a signature can't be taken back later. Like a paper signature, the digital signature creates a legal and psychological link between the signer of the message and the message.

As we will discuss in detail in the next chapter, since digital signatures use PKI, both a public key and a private key must be acquired in order to use the scheme. The private key is kept and used by the signer to sign documents. The person who verifies the document then uses the signer's corresponding public key to make sure the signer is who he or she claims to be. With keys, the user sends the authentication request to the ACS. Upon receipt of the request, the server uses its private key to decrypt the request. Again, as we will discuss in Chap. 10, both these keys are only mathematically related, so knowing the public key to verify the signer's signature does not require knowledge of the signer's private key. Many times, it is very difficult to compute one key from the knowledge of the other.

10.5.6 Wireless Authentication

Because of the growing use of wireless technology and its current low security, there is a growing need for wireless network authentication for mobile devices as they connect to fixed network as well as mobile networks. The IEEE 802.1X, through its Extensible Authentication Protocol (WEP), has built in authentication for mobile unit users. This authentication requires Wi-Fi mobile units to authenticate with network operating systems such as Windows XP.

10.6 Developing an Authentication Policy

Although in many organizations, the type of authentication used is not part of the security policy, which means that the rank and file of the users in the organization do not have a say in what authentication policy is used, it is becoming increasingly popular nowadays to involve as wide a spectrum of users as possible in as much detail of security as possible.

This means an early involvement of most users in the development of the authentication policy. Sometimes it even requires input from business and IT representative communities that do business with the organization. This is sometimes key to ensuring acceptance and compliance by those communities. Paul Brooke lists the following steps as necessary for a good authentication policy [15]:

- List and categorize the resources that need to be accessed, whether these resources are data or systems. Categorize them by their business sensitivity and criticality.
- Define the requirements for access to each of the above categories taking into account both the value of the resource in the category and the method of access (such as LAN, Internet, or dial-up). For example, as Brooke notes, common internal resources, such as e-mail or file and print systems, might require that the single-factor authentication included in the operating system is sufficient as long as the access is via the internal LAN.
- Set requirements for passwords and IDs. Every authentication policy should clearly state requirements for the following:
  - *ID format*: Authentication policies should strive to employ as universal an ID format as possible to make the management of IDs and passwords much easier.
  - *Complexity*: Whether to require nonalphabetic characters or not in the passwords.
  - *Length*: Stating the minimum and maximum password lengths.
  - *Aging*: Stating the frequency in changing passwords.
  - *Reuse*: How frequently a password can be reused.
  - *Administrative access*: Whether there will be special requirements for superuser passwords.

  - *Defaults*: To allow default passwords for vendors and other special interest users.
  - *Guest and shared accounts*: To decide if guest accounts will be used. If so, are there any special administration, password, or authentication requirements?
  - *Storage*: Required storage for passwords. This is important for the storage of encrypted or hashed passwords.
  - *Transmission*: To decide on the requirements for transmission of passwords; is clear-text transmission of passwords during authentication or is encryption required?
  - *Replication*: To decide on the requirements for replication of password databases; how often must it occur, and are there any special requirements for transmission?
- Create and implement processes for the management of authentication systems.
- Communicate policies and procedures to all concerned in the organizations and outside it. The creation of policies and procedures has no value unless the community regulated by them is made aware. Compliance cannot be expected if people are not conscious of the requirements.

Exercises

1. Authentication is based on three factors. List the factors and discuss why each one determines which type of authentication to use.
2. Making an authentication policy must be a well-kept secret to ensure the security of the intended system. Why then is it so important that a security policy include an authentication policy that involves as many as possible? What kind of people must be left out?
3. In RPC authentication, why it is necessary that each client request that server services be authenticated by the authentication server?
4. The Kerberos authentication process actually involves two tickets. Explain the need for each ticket and why only one ticket cannot be used.
5. Discuss in detail the role played by each one of the five players in a Kerberos authentication process.
6. There are many compelling reasons why a system that must implement security to the maximum must give anonymous authentication to a class of users. Detail five of these reasons.
7. Does anonymous authentication compromise the security of systems for the advantages of a few services?
8. Discuss the role of certificate authentication in e-commerce.
9. Many predict that the future of e-commerce is pegged on the successful implementation of authentication. Discuss.
10. Discuss the role of public key authentication in the growth of e-commerce.

Advanced Exercises

1. Research and discuss the much talked about role of public key authentication in the future of e-commerce. Is the role of PKI in authentication exaggerated?
2. Study the dial-in authentication mechanisms. What mechanisms (discuss five) can be used in EAP?
3. Discuss the benefits of enhancement of basic authentication with a cryptographic scheme such as Kerberos, SSL, and others. Give specific examples.
4. Authentication using certificates, although considered safe, suffers from weaknesses. Discuss these weaknesses using specific examples.
5. Kerberos and SSL are additional layers to enhance authentication. Detail how these enhancements are achieved in both cases.

References

1. Pipkin DL (2000) Information security: protecting the global enterprise. Prentice Hall, Upper Saddle River
2. Holden G (2004) Guide to firewalls and network security: intrusion detection and VPNs. Thomason Learning, Boston
3. The Rainbow Books. National Computer Security Center, http://fas.org/irp/nsa/rainbow.htm
4. Marshall B. Consider your options for authentication. http://www.passwordresearch.com/files/TipsforAvoidingBadQuestions.pdf
5. Cryptography Research Group – Projects. http://www.research.ibm.com/security/projects.html
6. Galbraith S, Mao W Invisibility and anonymity of undeniable and consumer signatures. http://www-uk.hpl.hp.com/people/wm/papers/InAnRSA.pdf
7. Glossary of Terms. http://www.asuretee.com/developers/authentication-terms.shtm
8. Kaeo M (1999) Designing network security: a practical guide to creating a secure network infrastructure. Cisco Press, Indianapolis
9. Digital Signature Authentication. https://www.google.com/patents/US20080222049
10. Configuring SSL Authentication. Oracle advance security administrator's guide release 8.1.5. A677–01. http://www.csee.umbc.edu/help/oracle8/network.815/a67766/09_ssl.htm
11. Tung B. The Moron's Guide to Kerberos. http://www.isi.edu/~brian/security/kerberos.html
12. Barkley J. Robust authentication procedures. http://csrc.nist.gov/publications/nistpubs/800–7/node166.html
13. General Information on Kerberos. http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html#tgttgs
14. Winkelmeyer R. Explaining Certificate Authentication – 101 Style July 14. https://blog.winkelmeyer.com/2014/07/explaining-certificate-authentication-101-style/
15. Brooke P. Setting the stage for authentication network computing. http://www.networkcomputing.com/1211/1211ws22.html