PC & Network Security

CNET – 250 Section 01

David L. Sylvester, Sr., Assistant Professor

Chapter 1

Introduction

Fundamental Concepts

An important aspect of computer security is the identification of vulnerabilities in computer systems, which can, for instance allow a malicious user to gain access to private data and even assume full control of a machine. Vulnerabilities enable a variety of attacks.

Analysis of these attacks can determine the severity of damage that can be inflicted and the likelihood that the attack can be further replicated. Actions that need to be taken to defend against attacks include:

- Identifying compromised machines,
- Removing the malicious code, and
- Patching systems to eliminate the vulnerability.

In order to have a secure computer system, sound models are a first step. This involves:

- Defining the security properties that must be assured,
- Anticipate the types of attacks that could be launched, and
- Develop specific defenses.

The design should also take into account usability issues (*Security measures that are difficult to understand and inconvenient are likely not to be adopted.)*

Next, the hardware and software implementation of a system needs to be vigorously tested to detect programming errors that introduce vulnerabilities.

Once the system is put into operation, procedures should be put into place to monitor the system for security breaches.

Finally, security patches must be applied as soon as they become available.

Confidentiality, Integrity, Availability

(The CIA Concepts)

Computers and networks are being misused at a growing rate. Spam, phishing, and computer viruses are becoming multibillion-dollar problems, as is identity theft, which poses a serious threat to the personal finances and credit ratings of users, and create liabilities for corporations.

Spam is flooding the Internet with many copies of the same message, in an attempt to force the message on people who would not otherwise choose to receive it.

Phishing is the act of sending an e-mail to a user falsely claiming to be an established legitimate enterprise in an attempt to scam the user into surrendering private information that will be used for identity theft.

Computer viruses are computer programs that can copy itself and infect a computer.

Information security has been defined in terms of the acronym C.I.A., which in this case stands for confidentiality, integrity and availability.

Confidentiality

Confidentiality is the avoidance of the unauthorized disclosure of information. This involves the protection of data (keeping the data secret), providing access for those who are allowed to see that data while disallowing others from learning anything about its content.

With the various threats to the confidentiality of data today, computer security researchers and system designers have come up with a number of tools for protecting sensitive information.

These tools incorporate the following concepts:

- **Encryption** which is the transformation of information using a secret, called an encryption key, so that the transformed information can only be read using another secret, called the decryption key.
- **Access Controls** are rules and policies that limit access to confidential information to those people and/or systems with a "need to know." This need to know may be determined by identity, (name, computer's serial number, or the role of that person)

- **Authentication** which is the determination of the identity or role that someone has. (smart card, radio key for storing secret keys, password, or fingerprint)
- **Authorization** which involves determining if a person or system is allowed access to resources, based on an access control policy. (These authorization policies should prevent an attacker from tricking the system into letting them have access to protected resources.
- **Physical Security** involves the establishment of physical barriers to limit access to protected computational resources. (Barriers may include locks on cabinets and doors, the placement of computers in windowless rooms, the use of sound dampening materials and even the construction of buildings or rooms with walls incorporating copper mesh so that electromagnetic signals cannot enter or exit the enclosure.

Example

When we visit a web page that asks for your credit card number and the internet browser shows a little lock icon in the corner, several things happen in the background to help ensure the confidentiality of your credit card number.

- The browser begins by performing an authentication procedure to verity that the web site connected to is indeed who it say it is.
- The web site checks to see if the browser is authentic and have the appropriate authorizations to access the web page according to its access control policy.
- Your browser may then asks the web site for an encryption key to encrypt the credit card, which it then uses to send the credit card information in encrypted form.
- Finally, once the credit card number reaches the server that is providing the web site, the data center where the server is located should have appropriate levels of physical security, access policies, and authorization and authentication mechanisms to keep your credit card number safe.

Integrity

Integrity which is the property that information has not been altered in an unauthorized way.

The importance of integrity is often demonstrated to school children in the Telephone game. (*A message is whispered onto others and the last person says the message aloud.*)

Typically, the message has been so mangled by this point that it is a great joke for all the children. This game is done to show that as the statement is passed from person to person it may and almost always lose it's integrity.

There are a number of ways that data integrity can be compromised in computer systems and networks, and these compromises can be benign (i.e. disk drive crash), or malicious (i.e. virus effecting the system, deliberately changing files of the operating system, and the system replicates the virus and sends it to other computers).

Tools That Support Integrity

- **Backups**: periodically archiving your data. Archiving is done for restoration purposes, in case data is altered in an unauthorized or unintended way.
- **Checksums**: computation of a function that maps the contents of a file to a numerical value. This technique is used to detect when a breach of data integrity has occurred.
- **Data Correcting Codes**: methods that involves storing data in such a way that small changes can be easily detected and automatically corrected.

These tools for achieving data integrity all possess a common trait- they use redundancy; which involves the replication of some information content or functions of the data so that we can detect and sometimes even correct breaches in data integrity.

In addition to maintaining the integrity of the actual data, the integrity of the metadata for each file must also be maintained.

metadata - *data about other data (access attributes, owner of file, last user to modify file, last user to read the file, dates and times file was created modified and accessed, name and location of the file and the list of users or groups who can read or write the file.*

Example

A computer intruder might not actually modify the content of any user files in a system he has infiltrated, but he may be modifying metadata, such as access time stamps, by looking at our files (and thereby compromising their confidentiality if they are not encrypted). If the system has integrity checks in place for this type of metadata, it may be able to detect and intrusion that would have otherwise gone unnoticed.

Availability

Besides confidentiality and integrity, another important property of information security is availability, which is the property that information is accessible and modifiable in a timely fashion by those authorized to do so.

Example

If someone stole your credit card and it took weeks before the credit card company could notify anyone, because its list of stolen numbers was unavailable to merchants. Thus, as with confidentiality and integrity, computer security researchers and system designers have developed a number of tools for providing availability.

Two tools used for providing availability

- **Physical protections**: infrastructure meant to keep information available even in the event of physical challenges (i.e. storms, earthquakes, and bomb blasts). These structures are outfitted with generators and other electronic equipment to be able to cope with power outages and surges.
- **Computational redundancies:** computers and storage devices that serve as fallbacks in the case of failures.
  - RAID (redundant array of inexpensive disks), use storage redundancies to keep data available to their clients.
  - Web servers are often organized in multiples called “farms” so that the failure of any single computer can be dealt with without degrading the availability of the web site.

Example

An attacker who otherwise doesn't care about the confidentiality or integrity of data may choose to attack its availability. A thief who steals lots of credit cards might wish to attack the availability of the list of stolen credit cards that is maintained and broadcasted by a major credit card company. Thus, availability forms the third leg of support for the vital C.I.A. triad of information security.

Assurance, Authenticity, and Anonymity A.A.A.

In addition to the C.I.A. concepts, there are a number of additional concepts that are also important in modern computer security applications. These concepts are categorized by the acronym A.A.A, which in this refers to **A**ssurance, **A**uthenticity, and **A**nonymity.

Unlike the C.I.A. concepts, the A.A.A concepts are independent of each other.

Assurance

Assurance refers to how trust is provided and managed in computer systems. Trust involves the following.

Policies – specifies behavior expectations that people or systems have for themselves and others. (Ex: the designer of an online music system may specify policies that describe how user can access and copy songs.

Permission – describes the behavior that are allowed by the agents that interact with a person or system. (i.e. online music store provide permissions for limited access and copying to people who have purchased certain songs.

Protections – describes mechanisms that are put in place to enforce permissions and policies. (i.e. online music store build in protections to prevent people from unauthorized access and copying of its songs.)

The designers of a computer systems want to protect more than just confidentiality, integrity, and availability of information. They also want to protect and manage the resources of these systems and they want to make sure users don't misuse these resources.

This could involve keeping unauthorized people from:

- using CPU memory
- Accessing networks

Trust management deals with the design of effective, enforceable policies, methods for granting permission to trusted users, and the components that can enforce those policies and permissions for protecting and managing the resources in the system.

Another important part of system assurance involves software engineering, where the software on the system be designed to conform to the engineer's system design.

Authenticity

Authenticity is the ability to determine that statements, policies, and permissions issued by persons or systems are genuine. If such things can be faked, there is no way to enforce the implied contracts that people and systems engage in when buying and selling items online.

Protocols that achieve such type of authenticity demonstrates nonrepudiation, which is the property that authentic statements issued by some person or system cannot be denied.

This is done through the use of digital signatures, which are cryptographic computations that allow a person or system to commit to the authenticity of their documents in a unique way that achieves nonrepudiation.

Digital signatures typically have some additional benefits over blue-ink signatures, in that digital signatures also check the integrity of signed documents. If document is modified, it becomes invalid.

Anonymity

When we interact with systems in ways that involves our real-world identities, we end up spreading our identity across a host of digital records, which ties our identity to:

- Our medical history,
- Purchase history,
- Legal records,
- Email communications,
- Employment records, etc.

Therefore, we have a need for anonymity, which is the property that certain records or transactions not be attributable to any individual.

To publish data in a privacy-preserving fashion, it can be done using the following tools:

- **Aggregation** – combining data from many individuals so that disclosed sums or averages cannot be tied to any individual.
- **Mixing** – intertwining transactions, information, or communications in a way that cannot be traced to any individual.
- **Proxies** – trusted agents that engage in actions for an individual in a way that the action cannot be traced back to that person.
- **Pseudonyms** – fictional identities that can fill in for real identities in communications and transactions, but are otherwise known only to a trusted entity. (i.e. anonymous user names on social networks)

Threats and Attacks

- **Eavesdropping** – the interception of information intended for someone else during its transmission over a communication channel. *(confidentiality)*
- **Alteration** – unauthorized modification of information. *(integrity)*
  - Man-in-the-middle attack - where information is intercepted, modified and retransmitted
  - Viruses – critical system files are modified to perform some malicious action
- **Denial of Service** – the interruption or degradation of a data service or information access *(availability)*
  - Email spam - fill up and slow down email server
- **Masquerading** – the fabrication of information that is claimed to be someone who is not actually the author. *(confidentiality and/or anonymity)*
  - Phishing – creating a web site that looks like a real bank or other e-commerce site, but is intended only for gathering passwords,
  - Spoofing – which may involves sending on a network, data packets that have false return addresses.
- **Repudiation** – denial of a commitment, which may involves an attempt to back out of a contract or a protocol that requires the different parties to provide receipts acknowledging that data has been received. *(assurance)*

- **Correlation and Traceback** – the integration of multiple data sources and information flows to determine the source of a particular data stream or piece of information. *(anonymity)*

Security Principles

- **Economy of mechanism** – This principle stresses simplicity in the design and implementation of security measures.
- **Fail-safe default** – This principle states that the default configuration of a system should have a conservative protection scheme. For example, when adding a new user to an operating system, the default group of the user should have minimal access rights to file and services.
- **Complete mediation** – every access to a resource must be checked for compliance with a protection scheme. One should be wary of performance improvement techniques that save the results of previous authorization checks, since permissions can change over time. (i.e. systems should require webpages with important information to time out)
- **Open design** – the security architecture and design of a system should be made publicly available. Security should rely only on keeping cryptographic keys secret. Open design allows for a system to be scrutinized by multiple parties, which leads to the early discovery and correction of security vulnerabilities caused by design errors. (i.e. open source software)

- **Separation of privilege** – dictates that multiple conditions should be required to achieve access to restricted resources or have a program perform some action. It also refers to the separation of components of a system, to limit the damage cause by a security breach on any individual component.
- **Least privilege** – Each program and user of a computer system should operate with the bare minimum privileges necessary to function properly. With this function in place, the abuse of privileges are restricted and the damage caused by the compromise of a particular application or user account is minimized. The military concept of need-to-know information is a good example of this principle.
- **Least common mechanism** – In systems with multiple users, mechanisms allowing resources to be shared by more than one user should be minimized. For example, if a file or application needs to be accessed by more than one user, then these users should have separate channels by which to access these resources, to prevent unforeseen consequences that could cause security problems.
- **Psychological acceptability** – states that user interfaces should be well designed and intuitive, and all security-related settings should adhere to what an ordinary user might expect. (could cause security problems such as misconfiguration

- **Work factor** – the cost of circumventing a security mechanism should be compared with the resources of an attacker when designing a security scheme. For example, a system developed to protect student grades in a university database, which may be attacked by snoopers, probably needs less sophisticated security measures than a system built to protect military secrets, which may be attacked by government intelligence organizations. But remember, technology advances so rapidly that intrusion techniques considered infeasible at a certain time may become trivial to perform within a few years.
- **Compromise recording** – this principle states that sometimes it is more desirable to record the details of an intrusion than to adopt more sophisticated measures to prevent it. (i.e. internet-connected surveillance cameras rather than reinforcing doors and windows) The compromise recording principle does not hold as strongly on computer systems, since it may be difficult to detect intrusion and adept attackers may be able to remove their tracks on the compromised machine by deleting log entries.

Access Control Models

One of the best ways to defend against attacks is to prevent them in the first place. By providing a rigorous means of determining who has access to various pieces of information, we can often prevent attacks on confidentiality, integrity, and anonymity.

Access Control Matrices

A useful tool for determining access control rights is the access control matrix, which is a table that defines permission. Each row in this table is associate with a subject, which is a user, group, or system that can perform actions. Each column of the table is associated with an object, which is a file, directory, document, device, resource, or any other entity for which we want to define access to.

Matrix Example

	/etc/password	/usr/bin/	/u/roberto/	/admin
root	read, write	read, write, exec	read, write, exec	read, write, exec
Mike	read	read, exec		
Roberto	read	read, exec	read, write, exec	
backup	read	read, exec	read, exec	read, exec

Access Control Models

(Advantages / Disadvantages)

Advantage

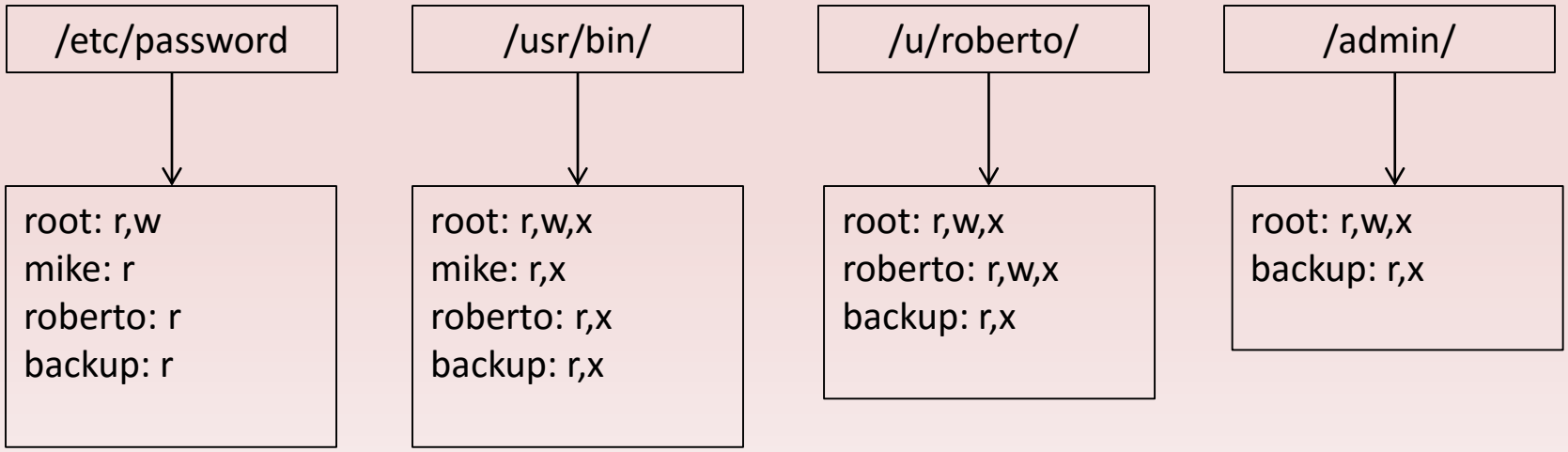
An access control matrix allows for fast and easy determination of the access control rights for any subject-object pair. Also, the access control matrix gives administrators a simple, visual way of seeing the entire set of access control relationships all at once.

Disadvantage

The size of the matrix, (number of rows and columns), can make it difficult for a system manager to fine the desires access control in a timely manner.

Access Control Lists

The access control list (ACL) model takes an object-centered approach. It takes each column of the access control matrix and compresses it into a list by ignoring all the subject-object pairs in that column that corresponds to empty cells.



Access Control Lists

(Advantages / Disadvantages)

Advantage

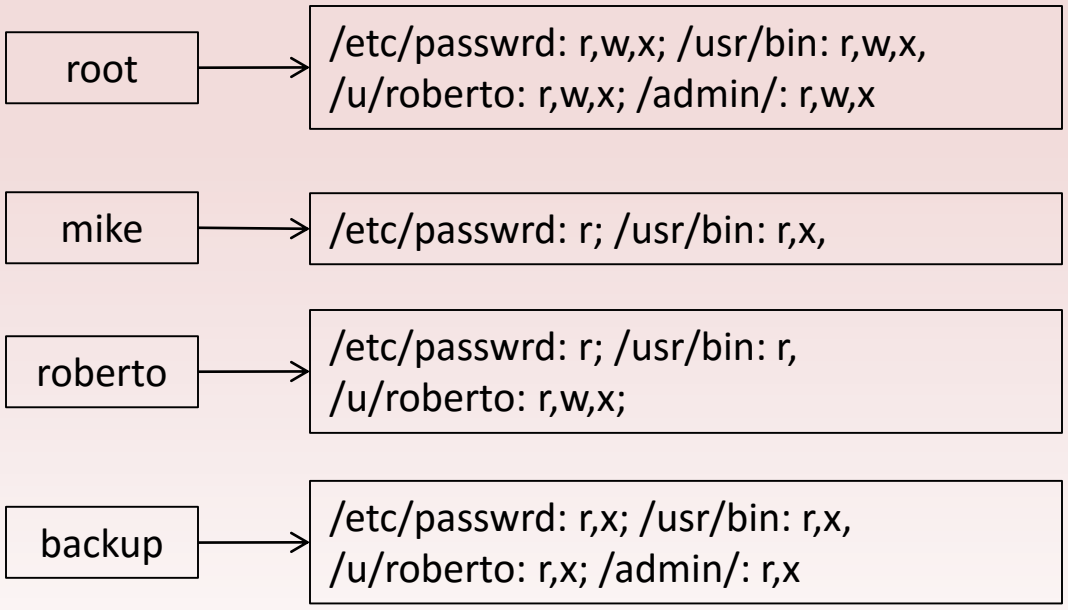
Access control lists are much smaller than an access control matrix. Also, the ACL of an object can be stored directly with that object as part of its metadata, which is particularly useful for file systems. (*The system need only consult the ACL of that object for it's access controls.*)

Disadvantage

An access control list do not provide an efficient way to enumerate all the access rights for a given subject. In order to determine all the access rights for a given subject, a secure system based on ACLs would have to search the access control list of every object looking for records that involves the record in question. (*Requires that a complete search of all the ACLs in the system to determine the access, whereas an access control matrix involves examining the row for the subject.*)

Capabilities

Another approach known as capabilities, takes a subject-centered approach to access control. It defines, for each subject s, the list of the objects which s has nonempty access control rights, together with the specific rights of each subject object. Thus, it is essentially a list of cells for each row in the access control matrix, compressed to remove any empty cells.



NOTE: r = read, w = write, x = execute

Capabilities

(Advantages / Disadvantages)

Advantage

The capabilities and access control list have the same advantages in regards to space. The system administrator only needs to create and maintain access control relationships for subject-object pairs that have nonempty access control rights. It also makes it easy for an administrator to quickly determine, for any subject, all the access rights that the subject has.

The administrator still needs to read off the capabilities list for that subject. Each time a subject ***s*** requests a particular access right for an object ***o***, the system needs only to examine the complete capabilities list for ***s*** looking for ***o***. If ***s*** has that right for ***o***, then it is granted it. (If the size of the capabilities list for a subject is not too big, this is a reasonably fast computations.

Disadvantage

The subjects are not associated directly with objects. Therefore, the only way to determine all the access rights for an object ***o*** is to search all the capabilities lists for all the subjects. With the access control matrix, such a computation would simply involve searching the column associated with object ***o***.

Role-Based Access Control

In role-based access control (RBAC), administrators define roles and then specify access control rights for these roles (groups), rather than for subject directly.

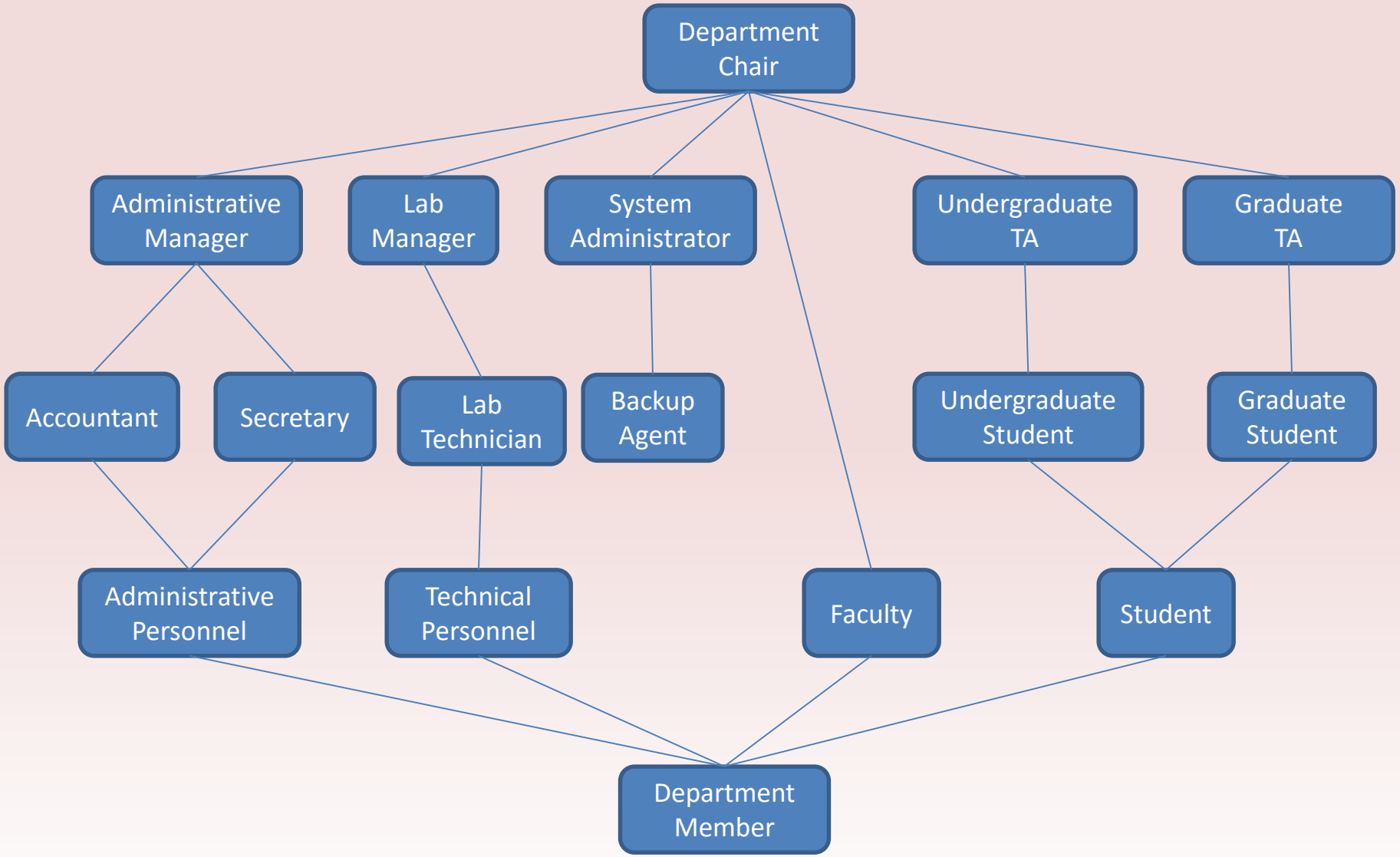
For example at a university, there could be the following roles, faculty, student, administrative personnel, administrative manager, backup agent, lab manager. Each role is granted the access rights that are appropriate for the class of users associated with that role.

The access rights for any subject are the union of the access rights for the roles that they have. For example a student that is part-time as a system administrator's assistant to perform backups on a departmental file system would have the roles "student" and "lab manager".

A hierarchy can be defined over roles so that access rights propagate up the hierarchy. (Ex: if a role (R1) is above another role(R2), in the hierarchy, then R1 inherits the access rights of R2.)

Hierarchies of roles simply the definition and management of permissions thanks to the inheritance property.

Example of Hierarchy of Role



Computer Science Department

Cryptographic Concepts

Technological solutions are the primary mechanism for enforcing security policies and achieving security goals. That's were cryptography comes in. Cryptographic techniques are used to achieve a broad range of security goals.

– Encryption: a means to allow two parties to establish confidential communication over an insecure channel that is subject to eavesdropping.

Example:

M = message (plaintext),

E = encryption algorithm,

C = ciphertext (encrypted text)

D = decryption algorithm,

Before the message is sent is it encrypted. The encryption process is denoted by:

C = E(M)

Once the message is received, a decryption algorithm is applied to recover the original plaintext. It is denoted by:

M = D(C)

The decryption key is normally attained by the input of a secret number or string. Likewise, the encryption algorithm uses encryption key to encrypt information.

Before the start of encryption can begin, the two parties need to agree on the ground rules they will be using to encrypt and/or decrypt the message. A cryptosystem consists of seven components.

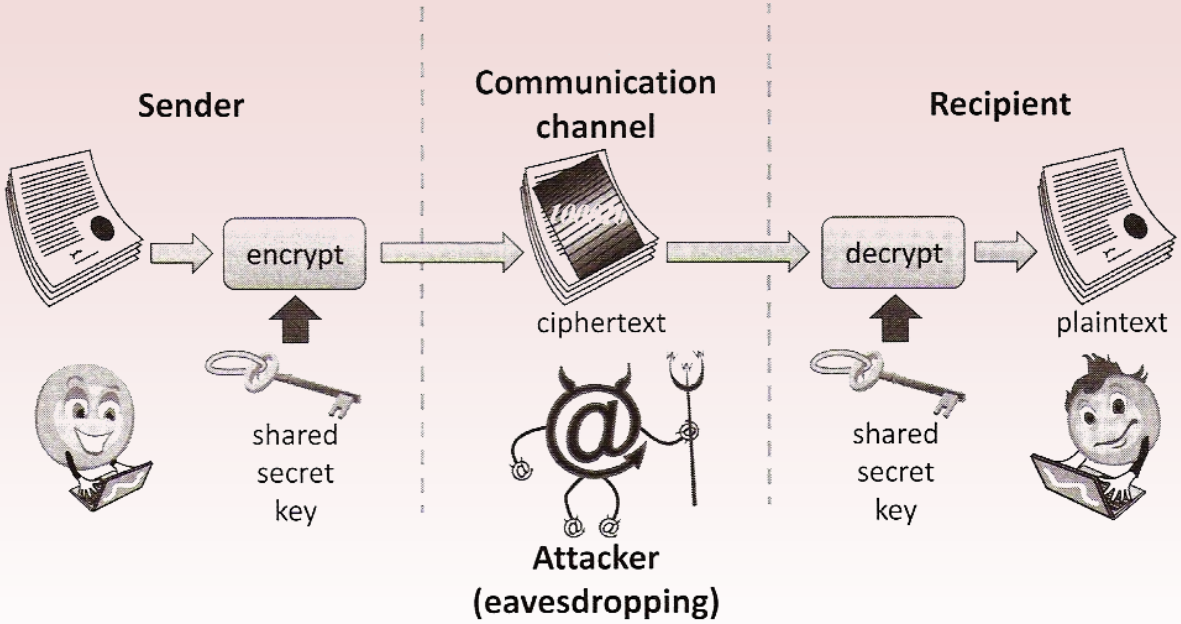
1. The set of possible plaintexts
2. The set of possible ciphertexts
3. The set of encryption keys
4. The set of decryption keys
5. The correspondence between encryption keys and decryption keys
6. The encryption algorithm to use
7. The decryption algorithm to use

Example

Let *c* be a character of the classical Latin alphabet (which consists of 23 characters) and *k* be an integer in the range [-22, +22]. We denote with *s(c,k)* the circular shift by *k* of character *c* in the Latin alphabet. The shift is forward when k > 0 and backward for k < 0. Example, *s(D, 3) = G*, *s(R, -2) = P*, *s(Z, 2) = B*, and *s(C, -3) = Z*. In the Caesar cipher, the set of plaintexts and the set of ciphertexts are the strings consisting of characters from the Latin alphabet. The set of encryption keys is {3}, that is, the set consisting of number 3. The set of decryption keys is {-3}, that is the set consisting of the number -3. The encryption algorithm consists of replacing each character x in the plain text with *s(x, e)*, where e = 3 is the encryption key. The decryption algorithm consists of replacing each character x in the plaintext with *s(x, d)*, where d = -3 is the decryption key. Note the encryption algorithm is the same as the decryption algorithm and that the encryption and decryption keys are one the opposite of the other.

There are modern cryptosystems that are much more complicated than the Caesar cipher, and much harder to break. The Advanced Encryption Standard (AES) algorithm, uses keys that are 128, 196, or 256 bits in length, so that it is practically infeasible for an eavesdropper, to try all possible keys in a brute-force attempt to discover the corresponding plaintext from a given ciphertext.

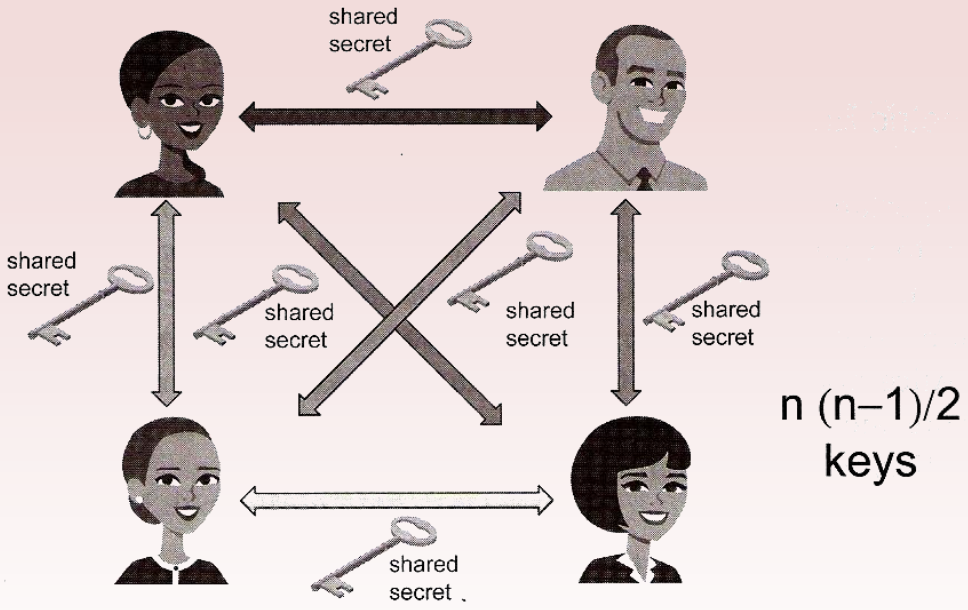
Symmetric Encryption – is when the same key *k* is used for both encryption and decryption. Symmetric cryptosystems are also called shared-key cryptosystem.



Symmetric Key Distribution

Symmetric cryptosystems, including the AES algorithm tend to run fast, but they require some way of getting the key *k* to both parties without an eavesdropper discovering it.

Suppose n number of parties wanted to exchange the same encrypted message with each other, in such a way that each message can be seen only by the sender and the recipient. Using a symmetric cryptosystem, a distinct secret key is needed for each pair of parties, for a total of n(n-1)/2.

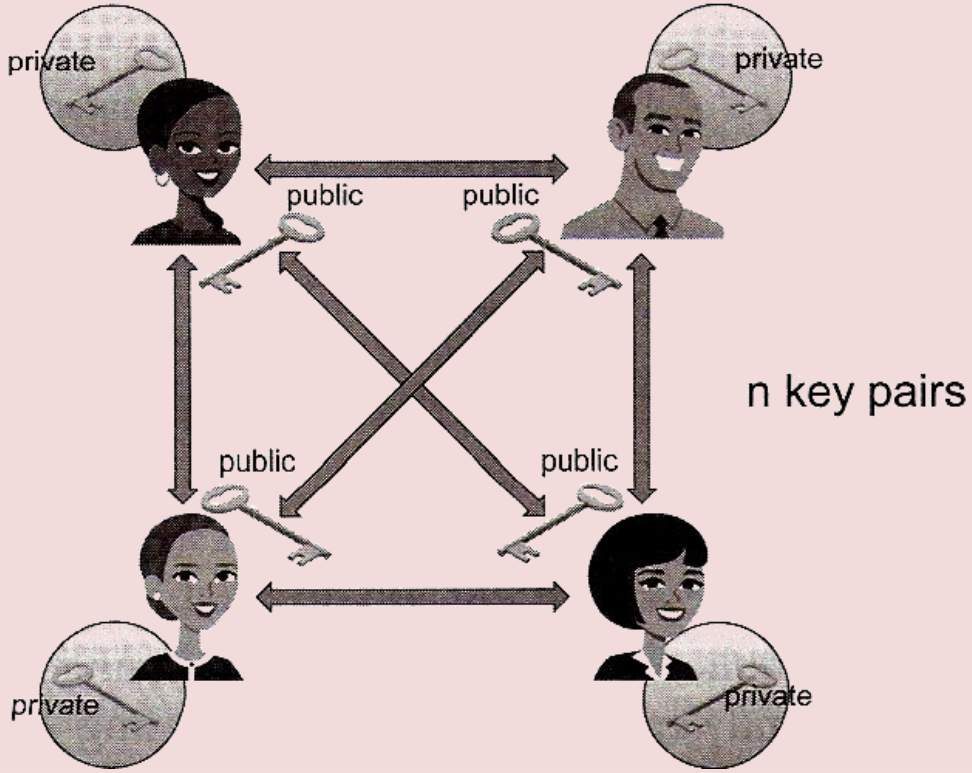


Public-Key Encryption – An alternate approach to a symmetric cryptosystem is the concept of a public-key cryptosystem. In this case, a recipient has two keys: a private key, S_B, which the recipient keeps secret, and a public key, P_B, which is broadcasted widely, possibly even posting it on his web page. In order for the sender to send an encrypted message to recipient, the sender need only obtain the public key, P_B, use that to encrypt the message, M and send the result, $C = E_{P_B}(M)$, to recipient. The recipient then uses the secret key to decrypt the message $M = D_{S_B}(C)$.

Note: In a public-key cryptosystem, the sender uses the public key of the recipient to encrypt and the recipient uses its private key to decrypt.

Advantages

- Public-key cryptosystems sidestep the problem of getting a single shared key to both recipient and sender.
- Only private keys need to be kept secret, whereas public keys can be shared with anyone, including the attacker.
- Public-key cryptosystems support efficient pairwise confidential communication among n users.

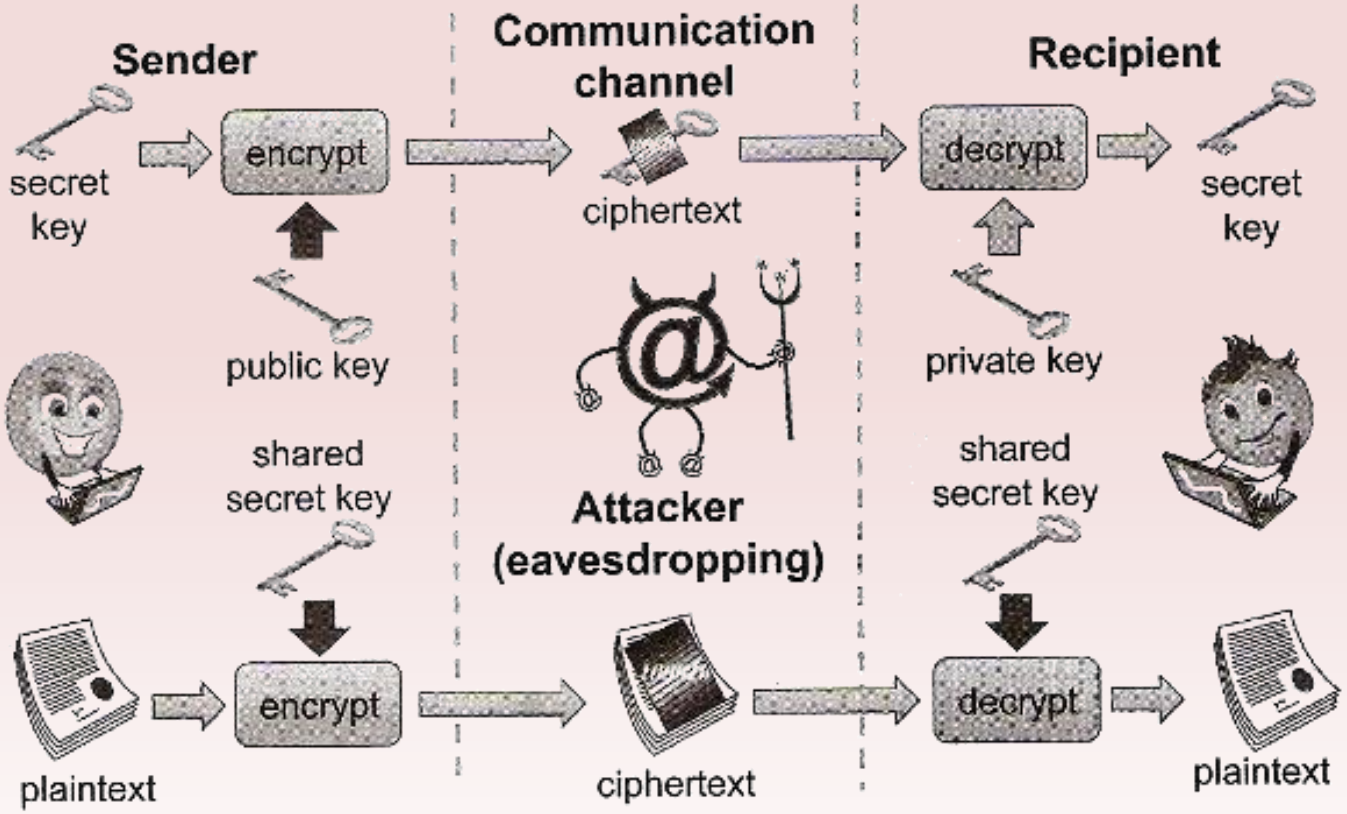


Pairwise confidential communication among n users with a public-key cryptosystem requires n key pairs. (***one per user***)

Disadvantages.

- The encryption and decryption algorithms such as RSA and ElGamal are much slower than those for existing symmetric encryption schemes.
- Public-key cryptosystems require a key length that is one order of magnitude larger than that of symmetric cryptosystems. (***RSA commonly use 2,048-bit keys while AES typically uses 256-bit keys***)

In order to get around these disadvantages, public-key cryptosystems are often used just to allow the sender and recipient to exchange a shared secret key, which they subsequently use for communicating with a symmetric encryption scheme.



Digital Signatures

Another problem that is solved by public-key cryptosystems is the construction of digital signatures. Derived from reversing the order in which encryption and decryption algorithms are applied.

$$E_{P_B}(D_{S_B}(M)) = M$$

This example demonstrates that a sender can give as input to the decryption algorithm a message (M), with it's private key (S_B), which the encrypted algorithm (E_{P_B}) and the public key (P_B) can be applied, yielding the message (M).

Using a Private Key for a Digital Signature

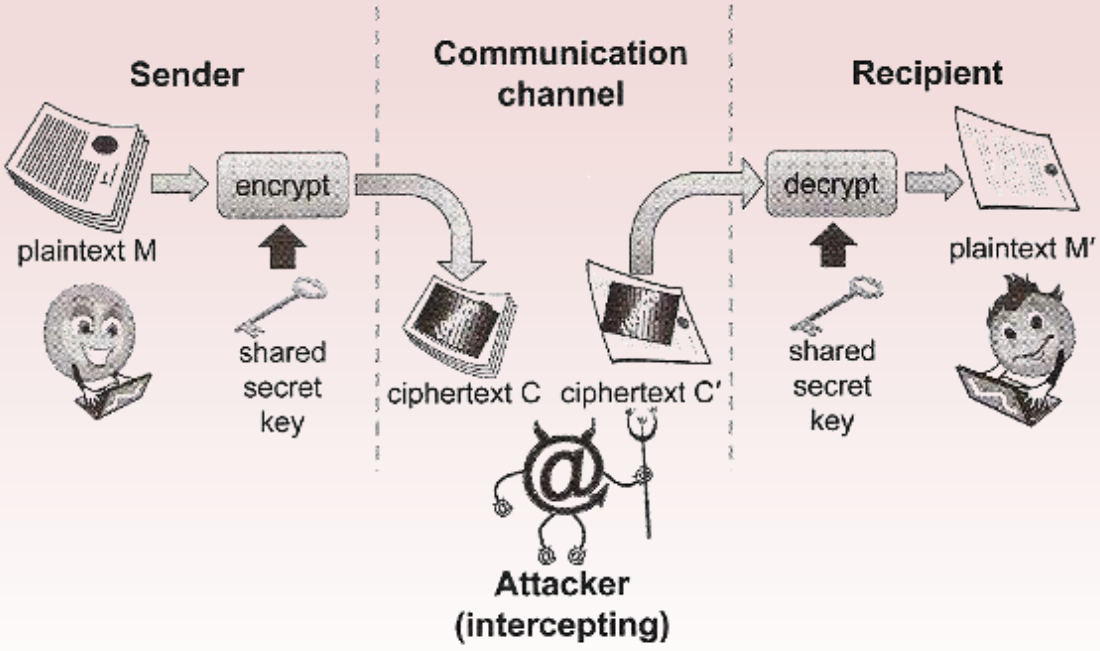
Anyone that knows the senders public key can convert its message. This may seem pointless, but that is exactly the point for a digital signature. (Only the sender could have done such a decryption, because no one else knows the secret key.) So if the sender intends to prove that he is the author of message M, he computes his personal decryption as follows: $S = D_{S_B}(M)$.

The decryption S serves as a digital signature for message M. The sender sends signature S to the receiver along with message M. Receiver can now recover M by encrypting signature S with sender's public key. $M = E_{P_B}(S)$.

Simple Attacks on Cryptosystems

Man-in-the-Middle Attacks

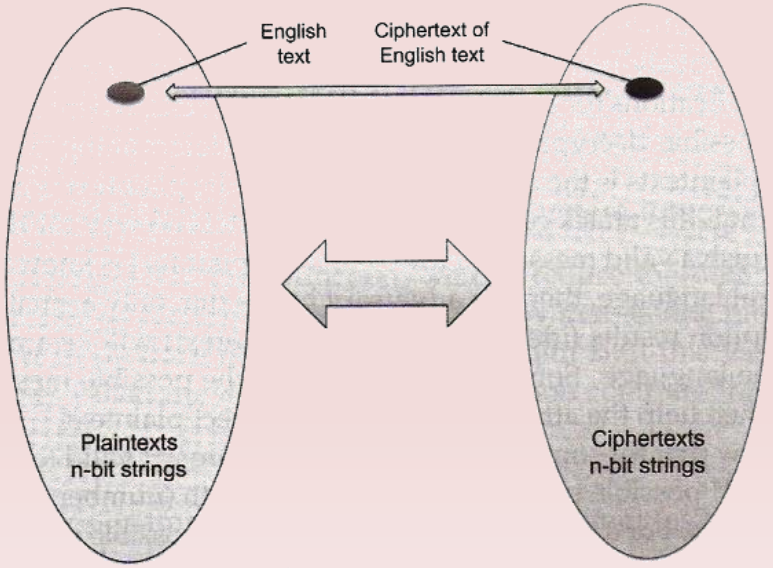
The simple use of a cryptosystem consists of just transmitting the ciphertext, which assures confidentiality, but does not guarantee the authenticity and integrity of the message if an adversary intercept and modify the ciphertext. Suppose a message is sent (ciphertext C), corresponding to message M. The adversary modifies C into an altered ciphertext (C') to be received. When the receiver decrypts C', he obtains a message M' that is different from M. Thus, the receiver is led to believe that he was sent M' instead of M. (*The same holds true for digital signatures.*)



Note that in the previous two attacks of man-in-the-middle, the adversary can arbitrarily alter the transmitted ciphertext or signature. However, the adversary cannot choose, or even figure out, what would be the resulting plaintext since he does not have the ability to decrypt. Thus, the two examples are effective only if any arbitrary sequence of bits is a possible message.

Brute-Force Decryption Attack

Now, suppose instead that valid messages are English text of up to t characters. With the standard 8-bit ASCII encoding, a message is a binary string of length n=8t. However, valid messages constitute a very small subset of all the possible n-bit strings.



Natural-language plaintexts are a very small fraction of the set of possible plaintexts. This fraction tends to zero as the plaintext length grows. Thus, for a given key, it is hard for an adversary to guess a ciphertext that corresponds to a valid message. If the plaintext is an arbitrary binary string, an attack cannot succeed, as there is no way for the attacker to distinguish a valid message. (***If the plaintext is known to be text in a natural language, the adversary hopes that the small subset of the decryption result will be a meaningful text for the language.***)

Cryptographic Hash Functions

To reduce the size of the a sent message, the cryptographic hash functions are often used. These functions are checksums on messages that have some additional useful properties. The most important being that the function be one-way, which means that it is easy to compute but hard to invert.

Example. Given a message, M, it should be relatively easy to compute the hash value, h(M). But given only a value y, it should be difficult to compute a message M such that y = h(M). SHA-256 hash functions are believed to be one-way functions, and result in values that are only 256 bits long.

Applications to Digital Signatures and File System Integrity

By using a cryptographic hash function to hash a message, time and space needed to perform a digital signature can be reduced. Ex: Given the message M, the message is hashed, producing the hashed message h(M), then sign the hashed value, (digest of M). $S = E_{S_B}(h(M))$

To verify signature S on a message M, the receiver computes h(M), which is easy, and then checks that - $D_{p_B}(S) = h(M)$

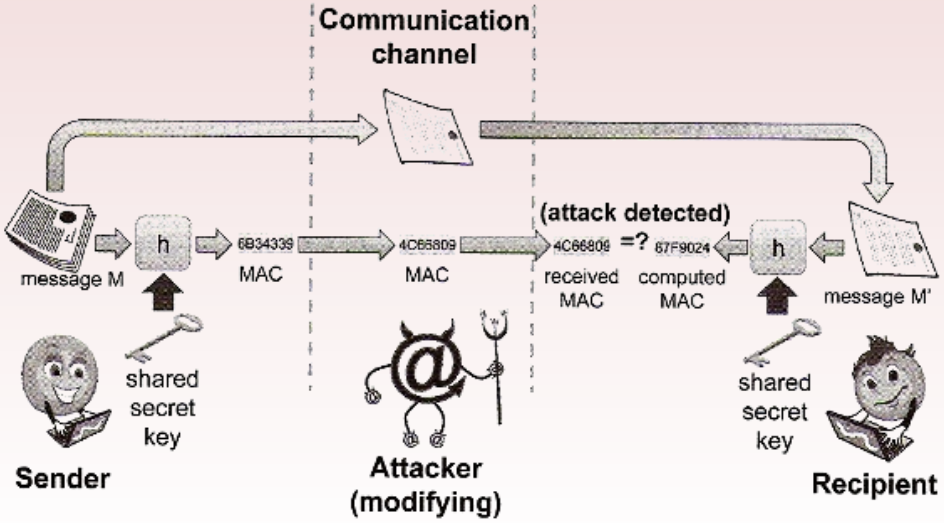
(***Signing a cryptographic digest of the message not only is more efficient than signing the message itself, but it also defends against the man-in-the-middle attach.***)

Message Authentication Codes

A cryptographic hash function h, can be used in conjunction with a secret key shared by two parties to provide integrity protection to messages exchanged over an insecure channel. Example: Suppose two individuals share a secret key K. When the sender want to send a message M to a receiver, he computes the hash value of the key K concatenated with message M: $A = h(K||M)$.

The value A, is called a message authentication code (MAC). Then the sender would send the pair (M,A). Since the communication is insecure, we denote with (M',A'), the received pair. Since the receiver knows secret key K, he computes the authentication code for the receive message M himself. $A'' = h(K||M)$.

If this computed MAC (A'') is equal to the received MAC (A'), then the receiver is assured that M' is the message sent by the sender.



Implementation and Usability Issues

In order for computer security solutions to be effective, they have to be implemented correctly and used correctly. Thus, when computer security solutions are being developed, designers should keep both the programmer and users in mind.

Password

One of the most common means of authenticating people in computer systems is through the use of usernames and passwords.

Password Characteristics

- Easy to remember
  - English like words, pet names, birthdays, anniversaries and last name
- Hard to guess
  - Random sequences of characters
  - Include lower and uppercase letters
  - Numbers and symbols
  - Changed periodically

Directory Attack

The problem with the typical easy-to-remember password is that it belongs to a small set of possibilities. Attackers know all these passwords and have built directories of them.

Ex: The English language

- Less than 50,000 common words
- 1,000 common human first names
- 10,000 common last names
- Only 36,525 birthdays and anniversary for almost every living human on earth, who is 100 years old or younger.

Armed with a dictionary of common passwords, one can perform an attack that is called a dictionary attack, which can break the computer's protection in a few minutes.

Secure Password

Secure passwords take advantage of the full potential of a large alphabet, thus slowing down dictionary attacks.

Example: If a system administrator insists on each password being an arbitrary string of at least eight printable characters that can be typed on a typical American keyboard, then the number of potential passwords is at least 9^{48} = 6,095,689,385,410,816 (at least 6 quadrillion). Even if a computer could test one password every nanosecond it would take on average, at least 3 million seconds to brake one such password (equivalent to one month of nonstop attempts).

There are several tricks for memorizing a complex password. *Note: writing it down is not one of them.* One way may be to memorize a silly or memorable sentence and then take every first letter of each word, capitalizing some, and then folding in some special characters.

Secure Password Examples

The sentence:

"Mark took Lisa to Disneyland on March 15,"

Could generate the password:

MtLtDoM1t5,

An even better password would be:

MtL+DoM15,

Which could last a user a lifetime.

Social Engineering

The three B's of Espionage – burglary, bribery, and blackmail – apply equally well to computer security. Add to these three techniques good old fashion trickery and we come up with one of the most powerful attacks against computer security solutions– social engineering, which refers to techniques involving the use of human insiders to circumvent computer security solutions.

Pretexting

Pretexting involves, a person attempting to attain login information from a helpdesk, etc., through false pretense. Ex: Contacting a helpdesk pretending to be someone else, claiming that you have forgotten your password and the helpdesk assigns and tell the user the password.

Emailing the password to the registered email account can help to secure the password.

Baiting

This attack involves using some kind of “gift” as a bait to get someone to install malicious software. Ex: An attacker could leave a few USB drives in the parking lot of a company with an otherwise secure computer system, even marking some with the names of popular software programs and games. The hope is that some unsuspecting employee will pick up the USB drive on his lunch break, bring it into the company, insert it into an otherwise secure computer, and unwillingly install malicious software.

Quid Pro Quo (Something for Something)

A person may pose as a helpdesk agent who was referred by a coworker. The agent may offer to do legitimate work on the user’s computer. Then, the agent ask the user for their password. The user being appreciative of the work already done on the computer, may feel appreciative of the work and feel free to give the agent the password. If this happens, the user becomes a victim of the quick-pro-quo attack.

Vulnerabilities from Programming Errors

Buffer Overflow Attack

A buffer overflow attack injects code written by a malicious user into a running application by exploiting the common programming error of not checking whether an input string read by the application is larger than the variable into which it is stored (the buffer). Thus a larger input provided by the attacker can overwrite the data and code of the application, which may result in the application performing malicious actions specified by the attacker.