

[Computing & Information Technology](#) > [Computing - general](#) >

E ENCYCLOPEDIA ARTICLE

[Print View](#) | [Email a Link](#)

Computer security

Contents [\[Hide\]](#)

- [Technical Controls](#)
- [Security of Programs](#)
- [Security of Operating Systems](#)
- [Database Security](#)
- [Network Security](#)
- [Bibliography](#)

The process of protecting against threats to computing systems. A threat is an event that can cause harm to computers, data or programs, or computations. A failure of computer security occurs because of a vulnerability or weakness in a computing system. A threat agent—person, event, or circumstance—exploits a vulnerability. Computer security involves protecting against failures of availability, integrity or correctness, and confidentiality or privacy.

A failure of availability is also known as denial of service. Partial denial of service is lack of capacity or unacceptable responsiveness. Computer users expect programs and data to be available on demand to meet computing needs. Applications such as power generation, stock trading, and even airplane cockpit navigation and aspects of medical care have become so dependent on computing that loss of availability becomes a serious threat to life or society. Even on a less dramatic level, people have become dependent on computers in aspects of everyday life, and so maintaining expected availability of computers is probably the most important of the three goals of computer security.

For further study:

N NEWS

February 06, 2005: [Outsmarting the Electronic Gatekeeper: Code breakers beat security scheme of car locks, gas pumps](#)

[View all 3 related news...](#)

D DICTIONARY

- [computer security](#)

Integrity means different things in context: sometimes it means that data or programs should be returned exactly as they were when they were recorded, or that modifications to data or programs should be made only by authorized persons, or by authorized programs, or in certain ways, or that the quality of data should be maintained. Data and programs should meet the demands of the way in which they are to be used.

The third category of failure is loss of confidentiality or privacy or inappropriate disclosure. Privacy of personal information or of data about individuals is a significant concern, as is that of sensitive corporate data (such as trade secrets and proprietary information) and government classified data.

The categories of security failures relate to authorized or acceptable behavior. In order for actions to be authorized or acceptable, a security policy must define who is allowed what access to what resources, what is to be protected, against what threats, to achieve what goals.

Computer users explicitly or implicitly perform a risk analysis by assessing the threats to computing, the likelihoods of these threats, and the harm that could occur from actualization of these threats. Risk must be balanced against the value of computing. Users select controls that prevent exploitation of one or more vulnerabilities as a means of lowering risk. Controls can be physical (such as a lock on a door), procedural (such as a policy that only supervisors can modify critical data items), or technical (such as a program control that blocks access to the password file). Controls can alter the balance among the three categories of failure: a high degree of confidentiality can reduce the availability of a system. See also: [Risk assessment and management](#)

Technical Controls

Technical controls can be a part of a computing system or application, or they can be added separately. If controls are added to an existing system, they must be integrated properly so that they function as expected.

Security enforcement mechanisms

A computing system must first be able to determine with high assurance the identity of the person who is seeking access (a process called identification and authentication), and the system must then verify that this person's proposed access is consistent with the security policy (a process called access control).

Identification and authentication

Identification is typically performed by logging in or entering a username. But after entering a name, a user may be asked to prove it, so that the system can be certain that one user is not trying to impersonate another. A user can authenticate an identity based on (1) what the user knows, such as a password, (2) who the user is, such as a physical characteristic (for example, a fingerprint), or (3) what the user possesses, such as a token. An authenticator must be something that cannot be easily forged, lost, forgotten, or guessed, while still making it easy for the legitimate owner to use. Techniques can use two or more approaches.

User passwords are commonly employed. Password guessing attacks use computers and actual dictionaries or large word lists to try likely passwords. Brute force attacks generate and try all possible passwords. To block these attacks, users should choose strong passwords.

Physical characteristics can be determined by biometric devices. In addition to fingerprints, voice recognition, retina patterns, and pictures are used. Although biometric authentication devices can be expensive, they are less susceptible to forgery and guessing than other methods, and they tend to be easy for users to adopt.

See also: [Fingerprint](#)

Tokens are widely used. Identity cards containing magnetic stripes or other elements that can be sensed combine computer access control with physical access control. Some identity cards contain small processors; these cards, called smart cards, can perform computation to guard against forgery. Another form of token resembles a pocket calculator; it is used for challenge-response, in which the user is given a challenge, a problem to solve, to which the calculator provides part of the answer. Each time the user seeks authentication, a different challenge is set, with a different corresponding answer. Challenge-response is especially desirable with computer network authentication because, even if someone intercepts the challenge and response, the interceptor cannot later reuse the response for impersonation because the challenge will be different.

Access control

The system uses a validated user identity to limit the actions the user can perform. An access control policy is a series of acceptable triples (user, object, action), such as (system administrator, password file, modify), meaning that the user "system administrator" is allowed to perform the action "modify" on the object "password file." An access control list (ACL) is a set of these triples. Access control lists can be represented as a two-dimensional matrix, as a set of rules, or in other ways.

Before permission to access an object is allowed, a reference monitor (also known as a reference validation mechanism or access control mechanism) checks that the access is allowable. A reference monitor must be complete (invoked to validate every reference permission), correct (made to implement the intended access control policy exactly), and tamperproof (unable to be disabled).

Reference monitors can simply process a representation of the access control policy in list or table form. Alternatively, they can process capabilities, which are prevalidated access "tickets." The access control system gives a user a capability to perform a certain access on a particular object, and the user later presents the capability to a reference monitor, which will inspect the capability and allow the access. Capabilities are useful in networked and distributed systems, in which access control may be done at one point and actions on objects may be done elsewhere. The Kerberos system is a popular distributed access control system based on capabilities.

Cryptography

A valuable tool in computer security is cryptography or encryption. Encryption is the process of transforming data so that the data are not readily understandable to unauthorized users; consequently unauthorized people cannot modify encrypted data in a knowledgeable way. Encryption is an important tool in computer security because it provides concealment and immunity to modification, so that it can be used to protect integrity and confidentiality. Cryptography alone is not a security control; it is a technology upon which controls can be based.

Encryption is a transformation function. Data to be encrypted are called plaintext, and the encrypted form is called ciphertext. The encryption function E is denoted $C = E(P)$, where P is a body of plaintext and C is its resulting ciphertext. There may also be a decryption function D that reverses the effect of encryption, so that $P = D(C) = D(E(P))$.

Encryption is a means by which two people can share a secret: one encrypts the secret and the other decrypts it. To facilitate sharing secrets using encryption, people use keyed encryption. A key is an additional parameter that tailors the encryption function, in the same way that a key tailors a basic lock so that it can be opened only by someone possessing the key. The key must be supplied to encrypt or decrypt, and so a single algorithm can represent many different encryptions with different keys.

Key management—arranging the distribution, revocation, and replacement of cryptographic keys—is very difficult, especially in a situation with many users who have no common basis for trust, as is true in many computing systems. The implementation of encryption must be secure in order for the encrypted results to protect the data adequately. See also: [Cryptography](#)

Security of Programs

Computer programs are both part of the protection and part of the things protected in computer security. Programs implement access controls and other technical security controls. But those same programs must be protected against accesses that would modify or disable their ability to protect. And those programs must be implemented correctly.

Correctness, completeness, and exactness

A computer program is correct if it meets the requirements for which it was designed. A program is complete if it meets all requirements. Finally, a program is exact if it performs only those operations specified by requirements. Computer security requires correct, complete, and exact programs, and nothing more. A program has inevitable side effects. For example, a program inevitably assigns values to internal variables, uses computing time, and causes entries to be generated in audit logs. Although side effects seem benign, they can be used maliciously to leak information. The exactness requirement really concerns only those significant operations specified by requirements, but in security almost any side effect can be significant. Determining which additional actions are security-relevant is difficult, if not impossible.

Correctness and completeness can be determined to some degree by careful testing, although with large or complex systems it may be infeasible to test all possible situations. It is difficult to test security systems appropriately, because they can be large and complex, and because it is hard to simulate all the environments and approaches by which systems can be attacked.

Malicious code

Computing is so fast and complex that users cannot know everything a program is doing. Programs can be modified or replaced by hostile forms, with the replacements seeming outwardly the same as the originals. The general term "malicious code" covers Trojan horses, viruses, worms, and trapdoors. Malicious code has been present in computing systems since the 1960s, and it is increasingly prevalent and serious in impact. Unfortunately, there are no known complete forms of protection against malicious code.

A Trojan horse is a program that has an undocumented function in addition to an apparent function. For example, a program may ostensibly format and display electronic mail messages while also covertly transmitting sensitive data.

A virus is a program that replicates and transfers itself to another computing system. When executed, each copy can also replicate, so that the infection spreads at a geometric rate. A virus typically inserts its replicated copy into another executable program so that when the other program is executed, so is the copy of the virus. Viruses often perform covert malicious actions.

A worm is a program that, like a virus, seeks to replicate and spread. However, the goal of the worm is only to spread and consume resources. The malicious effect of the worm is denial of service by exhaustion of resources.

A trapdoor is an undocumented entry point into a program. The trapdoor is inserted by a programmer to allow discreet access to a program, possibly with exceptional privileges. A user who had legitimate access at one time might have installed the trapdoor as a means of obtaining access in the future.

All these forms of malicious code are serious security threats for several reasons. First, malicious code can be relatively small, so that it is not readily detected. Second, its actions can be concealed: If a program fails to perform as it did, the change is evident, but an attacker can cause the change to be subtle, delayed, or sporadic, making it very difficult to detect, let alone diagnose and correct. The covert effect of malicious code can be almost anything: It can delete files, transmit messages or files, modify documents or data files, and block a user from accessing a computer system. The attack can be transmitted in pieces that activate only when the entire attack has been delivered. Finally, protecting against malicious code is difficult: The only known totally effective countermeasure is not to accept any executable items from anyone else, a solution that is scarcely acceptable for current networking and information-sharing environments.

Security of code

It is infeasible for a user to determine that a program is secure. The user has little evidence on which to base an opinion, an insecure program may intentionally hide its weaknesses, and many users have little control even over the sources from which programs are derived. Even well-intentioned programmers can fail. Beyond principles of good software engineering, the security field uses rigorous independent evaluation and penetration testing to add confidence that a program is secure. See also: [Computer programming](#); [Software engineering](#)

Independent evaluation

Users need an independent quality assessment, both that the programs meet their stated requirements and that they have no obvious security requirements. An international effort has led to the publication of the *Common Criteria for Information Technology Security Evaluation* (CC). The successor to the U.S. *Trusted Computer System Evaluation Criteria* (TCSEC or "Orange Book") and the European *Information Technology Security Evaluation Criteria* (ITSEC), the CC specifies a structure for defining security and functionality requirements (a protection profile) and a description of a product meeting those requirements (a security target) so that the security target can be evaluated objectively against the protection profile. The evaluation then serves as a seal of approval that, with a stated degree of confidence on the part of the evaluators, the product has met the requirements. The CC scheme has been applied to operating systems, firewalls, intrusion detection systems, cryptographic systems, database management systems, and networking products.

Penetration testing

Because programs intended to enforce security must be robust enough to enforce it rigorously, a special form of security testing is sometimes performed. Ordinary testing is directed toward verifying that the program does what it is expected to do; penetration testing is intended to determine if there is a way to make the program not do what it is expected to do. Penetration testers use known flaw types, common actual (and well-known) attacks, and intuition to study the system, hypothesize flaws that would prevent it from enforcing security, and conduct experiments to confirm or refute the flaws hypothesized.

Security of Operating Systems

Operating systems are the basis of security enforcement in a computing system because the operating system has control over the system resources: the file system, printers, shared buffers, memory, time, and shared programs. The original purpose of an operating system was to promote efficient and equitable sharing of computing resources, which led swiftly and naturally to security enforcement.

Operating system security functions

Operating systems are the first contact of a user with a computing system. Thus, the operating system performs user identification and authentication, as a basis for controlling resource usage and providing each user with a unique computing environment. Operating systems control access to primitive resources, such as files, devices, network connections, programs, and computing time. Most operating systems generate an audit log of what security-relevant events have occurred to the resources they control.

Operating system design for security

Operating systems are the point of fundamental security enforcement. Consequently, for strong security enforcement, security cannot be added successfully to an operating system after the design is partially completed. The reference monitor is the gate that controls access to all security-relevant objects. Several examples of secure operating systems have been implemented based on the reference monitor concept. These examples have involved thoughtful design followed by intense analysis of the design and its implementation. See also: [Operating system](#)

Database Security

A database is a collection of records containing fields, organized in such a way that a single user can be allowed access to none, some, or all of the data. Typically the data are shared among several users, although not every user will have access to every item of data. A database is accessed by a database management system that performs the user interface to the database.

Integrity is a much more encompassing issue for databases than for general applications programs, because of the shared nature of the data. Integrity has many interpretations, such as assurance that data are not inadvertently overwritten, lost, or scrambled; that data are changed only by authorized individuals; that when authorized individuals change data, they do so correctly; that if several people access data at a time, their uses will not conflict (for example, two people cannot write a single data item at the same time); and that if data are somehow damaged, they can be recovered.

A database management system is an application program that does not necessarily have a reliable interface to the operating system, and the operating system cannot necessarily be highly trusted. Therefore, a database management system typically maintains its own lists of users and their permitted actions, and the database management system may perform its own identification and authentication of users, quite independent from that performed by the operating system.

Inference and aggregation

Database systems are especially prone to inference and aggregation. Through inference, a user may be able to derive a sensitive or prohibited piece of information by deduction from nonsensitive results without accessing the sensitive information itself. For example, knowledge of any individual's salary may be restricted, but statistical measures (such as mean, median, maximum, and minimum) of the general salary pool may be freely released. If a user can formulate a database query that selects exactly one record (for example, "select all employees who are 6'2" tall and live in Glenwood, Maryland"), the user can then determine a single individual's salary by asking for the mean salary for the select set (of one employee). Various statistical methods make it very difficult to prevent inference.

A related problem is aggregation, the ability of two or more separate data items to be more (or less) sensitive together than separately. For example, neither the longitude nor the latitude of the location of a secret gold mine is sensitive by itself, but these two values together pinpoint the mine. It is infeasible to release one value to some people and the second value to others (even though no single person is told both) because two people could pool their knowledge. Aggregation is extremely difficult to prevent, since users can access great volumes of data from a database over long periods of time and then correlate the data independently.

Database controls

Encryption is sometimes used in database management systems for both confidentiality and integrity. A device called an integrity lock is used to ensure that the contents of a field are not modified. The field may be stored in plaintext, but a cryptographic checksum of the field, the integrity lock, is also stored. Each time an authorized change to the field is made, a new integrity lock is computed and stored; each time the contents of the field are retrieved (in response to a query), a lock is computed on the value of the field and compared with the stored lock. If the two values do not match, the field has been modified without authorization. To guard against swapping data and their locks, the encrypted data may comprise the actual data of the field together with the record identifier and the name of the field. See also: [Database management system](#)

Network Security

Enforcing security across a network has all the problems of local enforcement, coupled with problems brought on by distance and transmission. Identification and authentication, for example, requires not just verifying the identity of a requesting user but also identifying the system to the user. A user does not want to supply a password (or a credit card number or other sensitive information) to the wrong system. Even if the user is assured initially that the remote system is authentic, the user may need assurance continuously throughout a transaction that the remote system has not been commandeered by an attacker or that the communication has not been diverted to a hostile system.

In a network, the same considerations apply as with nonnetworked systems: identification and authentication, access control, and auditing. The two additional areas of concern are the communications link between the local user and the remote system, and the physical and technical security of the remote system. (In fact, the remote system may be multiple systems, with the user's communication passing from one to another without the user's knowledge.)

Communications security

It is necessary to protect the communications lines by which users communicate with remote systems. Lines are subject to interception, redirection, falsification, and interruption, and the attacks range from very crude (simply cutting a wire) to very sophisticated (intercepting a message, modifying portions of it, and retransmitting the modified message).

Strong encryption is very effective in protecting against interception and modification. It is useful to consider a communication from A, through intermediate points X and Y, and ultimately to B. With end-to-end encryption, the message is encrypted at A and travels in encrypted form all the way to B. End-to-end encryption can be applied from an application to a corresponding application, so that the message is protected even as it is stored temporarily on a file system or as it passes through the operating system. This approach requires that part of the message, the routing and destination information, not be encrypted, so that along the way X and Y, which do not share the encryption between A and B, are able to move the message toward its destination.

An alternative is link encryption, in which a message is encrypted just along one path, say from A to X. At X the message is decrypted and reencrypted for the link from X to Y, and so forth. Link encryption can be applied at the physical link from A to X, so that all communications along that path are protected. Key management may be simpler than with end-to-end encryption because the encryptor needs to have keys only for its neighbors, the next points to which it sends communications. The individual user does not have to apply the encryption, manage cryptographic keys, or even understand encryption. However, a message is exposed at X and Y and also within the source and destination computing systems up to the application that handles the message.

Virtual private networks

Organizations can benefit from sharing access to computing systems that are not all located together. To do this, organizations have established virtual private networks (VPNs). These networks approach the security of a private network at costs closer to those of shared public resources. The primary security technique used is encryption. A virtual private network involves a set of known participants since an organization knows which systems will form a part of its network. Thus, cryptographic key management is reasonably straightforward: By some secured means, such as postal mail or a telephone conversation, a seed or master cryptographic key is distributed to all the systems that will participate in the private network. One party of the virtual private network then generates a new key, sometimes called a session key, that will be used for communications with that system or for a specific period of time. The session key is encrypted under the master key to be passed on to the other VPN partners.

Internet security

The Internet, or any similar public network, is subject to threats to its availability, integrity, and confidentiality. A complicating feature is that there is effectively no control on transmissions over the Internet. Consequently, a system connected to the Internet is exposed to any malicious attack that any other Internet user wants to launch.

Availability threats

Availability or denial of service threats range from disabling the connection of a system to flooding or saturating the system. The main cause of this type of threat is natural: loss of power, equipment failure, or a failed communications link. Disabling the physical connection requires access to the target system, but disabling the logical connection does not require physical access, and so it is a more likely choice for a malicious outside attacker. It requires only changing the routing information of the Internet so that the target host's address is no longer recognized. Fortunately, the Internet routing information is very well protected, so that this type of attack seldom succeeds.

A flooding attack is fairly common and difficult to protect against. The attacker either transmits (automatically) a vast amount of data to the target host, overwhelming its ability to handle legitimate communication, or the attacker exploits a flaw in the network communications protocols to cause the target host to generate its own massive volume of traffic to itself. A crafty attacker will falsify the source address of malicious communications, so that it is difficult to trace back to the attacker, making it infeasible to block or disregard communications from the malicious source.

Integrity threats

Threats to integrity include unauthorized changes to the data or programs of the target computing system. Integrity attacks seek to modify the data or programs at a target host. Malicious programs can be introduced by commandeering the target host, delivering the malicious program as if it were from a reliable source, or conveying the malicious program through a Trojan horse attack. Additionally, data can be modified by an attacker at any point on the communications link between the source and destination.

Again, encryption becomes a significant tool to control integrity. However, an advantage of the Internet is that it connects millions of sites, but the disadvantage is that there may be no shared encryption between all pairs of those sites. The difficulty is for two previously unknown entities (representing hosts or users) to authenticate each other. A certificate can be used to establish shared encryption with some degree of trust. A certificate contains an identification, an encryption key, and an encrypted field, called a digital signature, from a third party, attesting to the authenticity of the identity. Certificates can be countersigned by trustworthy third parties to add to their credibility. The encryption key in a certificate provides a basis for two unknown entities to establish a shared encryption.

Confidentiality threats

Confidentiality threats in a distributed environment are closely related to integrity threats. With minor modification, integrity attacks can be made to expose the sensitive data at the remote site or in transit. Several network protocols, such as secure http (shttp) or secure socket layer (ssl), use encryption to protect both confidentiality and integrity of communications on the Internet. See also: [Internet](#); [World Wide Web](#)

Security perimeter

A security perimeter is a logical boundary surrounding all resources that are controlled and protected. The protected resources are called a domain (or enclave or protected subnetwork). There may be overlapping domains of varying protection, so that the most sensitive resources are in the innermost domain, which is the best protected. Protecting the security perimeter may be physical controls, identification and authentication, encryption, and other forms of access control. Two controls that relate especially to the security perimeter are network vulnerability scanning and firewalls.

Scanning

A network vulnerability scan is the process of determining the connectivity of the subnetwork within a security perimeter, and then testing the strength of protection at all the access points to the subnetwork. With a network domain, if a forgotten access point is not secured, its weakness can undermine the protection of the rest of the domain. A network scanner maps the connectivity of a domain, typically by probing from outside the domain, to determine what resources are visible from the outside. Once all outside connections are identified, each is tested with a range of attacks to determine the vulnerabilities to which it is susceptible and from which it needs to be better protected.

Firewall

It is infeasible to seal off a domain completely against outside access, because a purpose of the domain is to support communication with the outside. For example, a web site may be intended for broad public access. A firewall is a host that functions as a secured gateway between a protected enclave and the outside. The firewall controls all traffic according to a predefined access policy. For example, many firewalls are configured to allow unhindered communication outbound (from the protected domain to a destination outside the domain) but to allow only certain kinds of inbound communication. A firewall can be a separate computer, or firewall functionality can be built into the communications switch connecting the enclave to the external network.

Some firewalls hide internal resources from being visible outside the enclave. An enclave may have registered only the address of the firewall; internal hosts are not registered as part of the Internet or external network, and therefore outsiders have no way to communicate directly with them, only through the firewall. The firewall can be configured to allow only certain types of communications, for example, electronic mail, while blocking communications that could affect the status or integrity or confidentiality of an internal host, such as network management commands. Proxy firewalls simulate the effects of complex interactions across a security perimeter, filtering undesirable information flows.

Unprotected hosts

Hosts that must be accessible to the general public are often situated outside the protected domain, so that they are readily accessible to users without exposing any of the resources inside the domain. An unprotected host is acknowledged to be exposed. Unchanging content of the host can be sealed with integrity checks to detect modification, and if the content is modified, either maliciously or accidentally, the content can be restored readily from a protected copy inside the domain. Data gathered at the open site are moved inside the protected domain for more secure storage.

Intrusion detection

It is most effective to eliminate vulnerabilities, but if that is not possible, it is then desirable to recognize that an attack is occurring or has occurred, and take action to prevent future attacks or limit the damage from the current one. Anti-intrusion technologies are the different ways in which an attack can be detected and countered. Anti-intrusion technologies are prevention, to block an attack; deterrence, to make an attack sufficiently difficult to discourage the attacker; detection, to detect the existence of an attack; deflection, to encourage the attacker to direct the attack elsewhere; and diminution, to limit the negative effect of an attack. Such technologies are often collectively referred to as intrusion detection systems.

Intrusion detection can be either anomaly detection, which seeks to identify an attack by behavior that is out of the norm, or misuse detection, to identify an attack by its attempted effect on sensitive resources.

Intrusion detection systems monitor a computing system in order to warn of an attack that is imminent, is under way, or has occurred.

Incident response

When an attack has occurred, the affected organization will generally study the attack in order to learn what has occurred, how to prevent a recurrence, and what is the extent of the damage. The science of computer forensics is the study of evidence from an attack. Computer forensics is much like any other investigative science in that it involves assembling potentially significant pieces of evidence and using insight, experience, and intuition to suggest connections among the pieces.

In order to assemble the knowledge about attacks and their countermeasures, organizations and countries staff computer emergency response teams (CERTs). These teams collect information about significant vulnerabilities and attack patterns and provide warnings and possible countermeasures. See also: [Local-area networks](#); [Wide-area networks](#)

Charles P. Pfleeger

Bibliography

- B. Cheswick and S. Bellovin, *Firewalls and Internet Security*, Addison Wesley, 1994
- D. Denning, *Cryptography and Data Security*, Addison Wesley, 1982
- D. Denning, *Information Warfare and Security*, Addison Wesley, 1999
- S. Garfinkel and G. Spafford, *Practical Unix and Internet Security*, 2d ed., O'Reilly and Associates, 1996
- S. Garfinkel and G. Spafford, *Web Security and Commerce*, O'Reilly and Associates, 1997
- C. Pfleeger, *Security in Computing*, 2d ed., Prentice Hall, 1997
- A. Rubin et al., *Web Security Sourcebook*, John Wiley, 1997

How to cite this article

Charles P. Pfleeger, "Computer security", in AccessScience@McGraw-Hill, <http://www.accessscience.com>, DOI 10.1036/1097-8542.153950

Copyright ©2007 The McGraw-Hill Companies. All rights reserved. Any use is subject to the [Terms of Use](#) and [Privacy Notice](#).
[Additional credits](#) and copyright information. Customer Privacy Notice



a silverchair information system

The McGraw-Hill Companies