Data Security and Integrity

6.1 Introduction

Data is a critical resource in many organizations. One of the challenging problems facing these organizations today is to ensure that only authorized individuals have access to data. Data also has to be protected from malicious corruption. Much of the early work on data security focused on multilevel secure data management systems where users have different clearance levels and data has different sensitivity levels and access to data is governed by the security policies. There were many efforts on securing relational, distributed and object oriented databases. More recently, several aspects of data security are being investigated including data confidentiality, integrity, trust and privacy. Furthermore, securing data warehouses, semantic web, as well as applying data mining for solving security problems are getting a lot of attention.

The concept of security within the network environment includes all aspects of operating systems, software packages, hardware, and networking configurations, as well as any network sharing connectivity. This is not the only scope of security. As a professional, you will find that in today's world, physical security is also linked to IT security. Security cameras need IP addresses and space to store recordings, disaster recovery plans must now cover the entire business entity, and physical access to unauthorized areas can be disastrous for a company. In this topic, we cover the basics of properly securing your network, which includes learning about threats and vulnerabilities, encryption, firewalls and best practices.

6.2 Fundamentals of Secure Networks

The modern computer world is a complex mix of different components with the purpose of making resources available to those who need it. Unfortunately, access to networks by unauthorized users has grown at a rate that far outpaces anything we could have imagined. Researchers react by quickly developing and deploying hardware and software to meet the demands of business and home consumers, but fail to properly test and secure these technologies. This puts networks at risk not only from the professional hacker but also from curious or disgruntled employees. Security is not just a policy or a plan, it is a mindset. You must properly train and cultivate employees to be security aware.

Most things you will learn about networking are relatively straightforward and can be accomplished. Do you want a new file and print server? You install it and set it up, and it either works or it doesn't. If it doesn't work, you proceed to troubleshoot it, fix any issues, and ultimately you complete the task. Network security, on the other hand, is a horse of a different color. You can never finish the project of securing a network and you can't ever be completely certain that a network is secure. How much money you invest in securing a network, how much time you devote to the job, or how much fancy security hardware and

software you install doesn't matter: No network is ever completely secure. (Hilariously, there's a corollary to this: The only secure network is the one nobody can use.)

Having said this, network security is one of the most important jobs facing any network administrator. Good network security helps prevent all the following:

- Company secrets, such as proprietary designs or processes, from falling into the wrong hands (both internally and externally)
- Personal information about employees from falling into the wrong hands
- Loss of important information and software
- Loss of use of the network itself
- Corruption or inappropriate modification of important data

The preceding is a list of only some of the more important things that network security can prevent. If you spend any time thinking about all the information that is stored on and that flows through networks with which you work (and you should spend time thinking about this), you'll probably come up with additional dangers to avoid.

Remember that your network is only as strong as its weakest link, which is usually a human being, a concept that we will discuss further in the "Social Engineering" section later. You will also have to stay current on threats and available patches to be sure that the servers and workstations are properly secured. Learning to implement encryption on sensitive data will be part of your job. Disaster recovery and incidence response plans need to be tested and updated on a regular basis. It all starts with proper planning.

We now look at a few things that will help us to design a secure network and understand how they work. If you design networks securely from the beginning, then you can eliminate problems later because you don't have to go back and rework anything. The first thing you need to do is know what you intend to do and why. Know what you're going to secure and how you're going to secure it, what you're going to block and allow and in order to do this, you might want to go back and look at your security policies. Remember, the security policies pretty much dictate what goes on in a network and why. That way you'll know what you need to do to implement those policies. You might want to take some time and sit down with paper and pencil or on the computer and list the things you want to allow and things you don't want to allow into and out of your network.

Draw out your network design on paper, diagram it out, maybe with something like Visio or maybe just with scratch paper. But diagram everything out, know where the devices are going to be and the connections that connect them and so forth. List the ports and protocols you're going to filter. Know what those are. Decide what you're going to encrypt and when across the network. Naturally, you don't encrypt everything; only the things you need to keep confidential because encryption can add some overhead to your network. So in order to decide what you're going to encrypt and when, you have to classify your data. You have to determine how sensitive it is and if it needs to be encrypted or not.

Now, you use network devices in your design to offer security benefit and of course hubs don't. They have no security benefit whatsoever. But do you need three firewalls or do you need one? So use those devices and those architectures that offer a benefit. If there's no benefit to it for you, then don't use it just because it's the latest and greatest thing out there.

Layer your defenses. Don't rely on one box alone to protect your network. Put some filtering rules on your border devices, such as your border router and put some other filtering rules on your firewalls.

Think about load balancing. If your router has all the rules on it and it gets destroyed by a hacker, now suddenly they're able to come into your network. That's a bad thing and at the same time, you don't want to tax them out too much by having them go through the motions of going through every single rule, 50 miles of rules on a table. So put some on your router, some on your firewall.

Use network security zones to protect both your published servers and internal networks. Published servers are basically servers that you want to allow Internet users outside your network to be able to access. We don't just put those out on the Internet. We put those in a security zone because they require some sort of protection. We also want to protect our internal networks as well. There's the intranet which is your internal network computers and internal servers. Those are the things that are at the core of your network that you definitely want to protect and you don't want outsiders to get to.

6.2.1 Internal Security

Internal security is the process of securing your network from internal threats, which are generally much more common (greater than 75 percent) than external threats. Examples of internal threats include the following:

- Internal users inappropriately accessing information to which they should not have access, such as payroll records, accounting records, or software development information
- Internal users accessing other users' files to which they should not have access
- Internal users impersonating other users and causing mischief, such as sending e-mails under another person's name
- Internal users accessing systems to carry out criminal activities, such as embezzling funds
- Internal users compromising the security of the network, such as by accidentally (or deliberately) introducing viruses to the network (viruses are discussed in their own section later in this chapter)
- Internal users "sniffing" packets on the network to discover user accounts and passwords

To deal with threats such as these, you need to manage the network's security diligently. You should assume that, in the population of internal users, at least some exist who have the requisite sophistication to explore security holes in the network and that at least a few of those may, at some point, have reason to do so. One of the more unpleasant parts of managing security is that you need to expect the worst of people and then you must take steps to prevent those actions you expect. It's not a happy mindset, but it is required to do a good job in the security arena.

6.2.2 Account Security

Account security refers to the process of managing the user accounts enabled on the network. A number of tasks are required to manage user accounts properly, and the accounts should be periodically audited (preferably by a different person than the one who manages them daily) to ensure that no holes exist. Following are a number of general steps you could take to manage general account security:

- Most NOSs start up with a user account called "Guest." You should remove this account immediately because it is the frequent target of crackers.
- Most NOSs start up with a default name for the administrative account. Under Windows NT, the account is called Administrator; under NetWare, it is called either Supervisor or Admin (depending on which version you are using). You should immediately rename this account to avoid directed attacks against the account. (Under NetWare 3.x, you cannot rename the Supervisor account.)

TIP: As a safety measure, also create a new account to be a backup of your administrative account. Call it whatever you like (although less obvious names are better), give the account security equivalence to the administrative account, and safely store the password. Should something happen that locks you out of the real administrative account, you can use the backup account to regain access and correct the problem.

- You should know the steps required to remove access to network resources quickly from any account and be sure to explore all network resources that might contain their own security systems. For example, accounts will be managed on the NOS (and possibly on each server) and also in specific applications, such as database servers or accounting systems. Make sure that you find out how the system handles removed or deactivated accounts. Some don't actually deny access to resources if the account is removed until the user logs out (or is logged out in some fashion).
- Work closely with the Human Resources department so that it is comfortable working with you on handling security issues related to employee departures. The HR department may not be able to give you much—if any—advance notice, but it needs to understand that you need to know about any terminations immediately, so you can take proper steps. Along the same lines, you want to work out a set of procedures on how you handle accumulated e-mails, files, and other user access both for friendly departures and terminations. Your relationship with the appropriate people in the

HR department is crucial in being able to handle security well, so make sure that you establish and maintain mutual trust.

- Consider setting up a program whereby new users on the network have their assigned permissions reviewed and signed off by their supervisor. This way, you won't mistakenly give someone access to things he or she shouldn't have.

- Another important aspect of account security is account password security. Most NOSs enable you to set policies related to password security. These policies control how often the system forces users to change their passwords, how long their passwords must be, whether users can reuse previously used passwords, and so forth. At a minimum, consider these suggestions for password policies:

- You should cause users to change their main network password every 90 to 180 days (30 days is a common recommendation, but this may be too frequent in most environments).

- You should set the reuse policy so that passwords cannot be reused for at least a year.

- You should require passwords that are at least six characters long. This yields, on a random basis, 366 possible permutations, or a bit over 2 billion possibilities. Using eight characters yields 368, or almost 3 trillion, possibilities. And if the NOS uses case-sensitive passwords, then the possibilities are much larger: 626 (57 billion) and 628 (218 trillion), respectively. Even 2 billion possibilities is a lot. If someone were able to try one password a second, he or she would have to spend 63 years to try that many permutations.

Many password-cracking programs rely on dictionaries of common words and names to reduce dramatically the number of possibilities they have to try. Because of this, encourage users to create passwords that are not words in any language or, if they are words, that they have numbers and other nonalphanumeric characters inserted somewhere in the word so a "dictionary attack" won't work. Also, for networks that support mixed-case passwords, encourage users to use mixed-case characters.

- Make sure that you turn on any policies that monitor for and deal with people entering in wrong passwords. Often called intruder detection, this type of policy watches incorrect password attempts. If too many attempts occur within a set period of time, the system locks the user account, preventing further attempts. May be set this type of feature to lock an account any time five incorrect passwords are entered within an hour and then to lock the account for either a number of days or forever (until it's reset by the administrator). This way, if someone is entering in a large number of incorrect passwords, he or she will have to talk with you to reopen the account, and you can find out why this situation developed and then correct it.

- Novell NetWare and Windows NT enable you to establish limits on when and where a user can log on to the network. You can establish times of day that a user is allowed to log on and you can also restrict a user account to a particular network node. Doing

so for all users on the network is usually overkill, but you may want to consider restricting the administrative account to several different workstations, so someone at a different workstation (or coming in through a WAN connection) cannot log in to the account, even if that person somehow knows the password.

There's an interesting catch-22 concerning network security policies: If you make them too strict, you can actually reduce the security of your network. For example, suppose that you set the network to force a password change once a week and to disallow the reuse of passwords. Most users will be unable to remember from week to week what password they're using and they will naturally resort to writing down their password somewhere in their office. Of course, a written password is much less secure than a remembered password. The trick with network security is to strike a balance between strict security and usability.

6.2.3 File and Directory Permissions

Another type of internal security that you need to maintain for information on your network involves the users' access to files and directories. These settings are actually a bit tougher to manage than user accounts because you usually have at least 20 directories and several hundred files for every user you have on the network. The sheer volume of directories and files makes managing these settings a more difficult job. The solution is to establish regular procedures, then follow them and then periodically spot-audit parts of the directory tree, particularly areas that contain sensitive files.

Also, structure the overall network directories so that you can, for the most part, simply assign permissions at the top levels. These permissions will "flow down" to subdirectories automatically, which makes it much easier to review who has access to which directories.

NOSs allow considerable flexibility in the permissions that they let you set on files and directories. Using the built-in permissions, you can enable users for different roles in any given directory. These roles control what the user can and cannot do within that directory.

Examples of generic directory roles include the following:

- **Create only** This type of role enables users to add a new file to a directory, but restricts them from seeing, editing, or deleting existing files, including any they've created. This type of role is perfect to enable a person to add new information to a directory to which they shouldn't otherwise have access. The directory becomes almost like a mailbox on a street corner: You can put only new things in it. Of course, another user will have full access to the directory to retrieve and work with the files.

- **Read only** This role enables users to see the files in a directory and even to pull up the files for viewing on their computer. However, the users cannot edit or change the stored files in any way. This type of role is good for material published to users who need to view the information, but which they should not change. (Users with read privileges can copy a file from a read-only directory to another directory and then do whatever they like with the copy they made. They simply cannot change the copy stored in the read-only directory itself.)

- **Change** This role lets users do whatever they like with the files in a directory, except they cannot give other users access to the directory.

- **Full control** Usually reserved for the "owner" of a directory, this role enables the owner(s) to do whatever they like with the files in a directory and, further, enables them to grant other users access to the directory. These roles are created in different ways on different NOSs.

Just as you can set permissions for directories, you can also set security for specific files. File permissions work similarly to directory permissions. For specific files, you can control a user's ability to read, change, or delete a file. File permissions usually override directory permissions. For example, if users had change access to a directory, but you set their permission to access a particular file in that directory to read-only, they would have only read-only access to that file.

6.2.4 Practices and User Education

Another important type of internal security concerns the most insecure part of any network: the people. You should be concerned about two things here. First, you need to establish good security practices and habits. It's not enough to design and implement a great security scheme if you do not manage it well on a daily basis. To establish good practices, you need to document security-related procedures and then set up some sort of process to make sure that the employees follow the procedures regularly.

In fact, you're far better off having a rudimentary security design that is followed to the letter than having an excellent security design that is poorly followed. For this reason, keep the overall network security design as simple as possible consistent with the needs of the company. You also need to make sure—to the maximum extent possible—that the users are following prudent procedures. Some of these you can easily enforce through settings on the NOS, but you must handle others through education.

Some tips to make this easier are as follows:

- Spell out for users what is expected of them in terms of security. Provide a document for them that describes the security of the network and what they need to do to preserve it. Examples of guidelines for the users include choosing secure passwords, not giving their passwords to anyone else, not leaving their computers unattended for long periods of time while they are logged in to the network, not installing software from outside the company, and so forth.

- When new employees join the company and are oriented on using the network, make sure that you discuss security issues with them.

- Depending on the culture of the company, consider having users sign a form acknowledging their understanding of important security procedures that the company expects them to follow.

- Periodically audit users' security actions. If the users have full control access to directories, examine how they've assigned permissions to other users.
- Make sure that you review the security logs of the NOS you use. Investigate and follow up on any problems reported. It's a good idea to document any security-related issues you investigate. While most are benign, occasionally you may find one in which the user had inappropriate intent. In such cases, your documentation of what you find and what actions you take may become important.

While it's important to plan for the worst when designing and administering network security, you also need to realize that most of the time security issues arise from ignorance or other innocent causes and not from malicious intent.

6.2.5 External Security

External security is the process of securing the network from external threats. Before the Internet, this process wasn't difficult. Most networks had only external modems for users to dial in to the network and it was easy to keep those access points secure. With the advent of the Internet and the fact that nearly all networks are connected to it, however, external security becomes much more important.

Earlier, you read that no network is ever totally secure. This is especially true when dealing with external security for a network connected to the Internet. Almost daily, hackers discover new techniques that they can use to breach the security of a network through an Internet connection. Even if you were to find a book that discussed all the threats to a specific type of network, the book would be out of date soon after it was printed.

Three basic types of external security threats exist:

Front-door threats: These threats arise when a user from outside the company somehow finds, guesses, or cracks a user password and then logs on to the network. The perpetrator could be someone who had an association with the company at some point or could be someone totally unrelated to the company.

Back-door threats: These are threats where software or hardware bugs in the network's OS and hardware enable an outsider to crack the network's security. After accomplishing this, the outsider often finds a way to log in to the administrative account and then can do anything he or she likes.

Denial of service: These are attacks that deny service to the network. Examples include committing specific actions that are known to crash different types of servers or flooding the company's Internet connection with useless traffic (such as a downloading a huge file during working hours).

A fourth type of external threat exists: computer viruses, Trojan horses, worms, and other malicious software from outside the company. These threats are covered in their own section below.

6.2.5 Viruses and Other Malicious Software

Unfortunately, an increasing array of malicious software is circulating around the world. Many different types of this software exist, including the following:

Viruses: A computer virus is a program that spreads by infecting other files with a copy of itself. Files that can be infected by viruses include program files (.COM, .EXE, and .DLL) and document files for applications that support macro languages sophisticated enough to allow virus behavior (Microsoft Word and Excel are common targets of macro-based viruses).

Worms: A worm is a program that propagates by sending copies of itself to other computers, which run the worm and then send copies to other computers. Recently, worms have spread through e-mail systems like wildfire. One way they spread is by attaching to e-mails along with a message that entices the recipients to open the attachment. The attachment contains the worm, which then sends out copies of itself to other people defined in the user's e-mail address book, without the user knowing that this is happening. Those recipients then have the same thing happen to them. A worm like this can spread rapidly through the Internet in a matter of hours.

Trojan horses: A Trojan horse is a program that purports to do something interesting or useful and then performs malicious actions in the background while the user is interacting with the main program.

Logic bombs: Logic bombs are malicious pieces of programming code inserted into an otherwise normal program. They are often included by the program's original author or by someone else who participated in developing the source code. Logic bombs can be timed to execute at a certain time, erasing key files or performing other actions.

More than 20,000 known viruses are in existence today, with more being written and discovered daily. These viruses are a major threat to any network, and an important aspect of your network administration is protecting against them. To protect a network from virus attacks, you need to implement some sort of antivirus software. Antivirus software runs on computers on the network and "watches" for known viruses or virus-like activity. The antivirus software then either removes the virus, leaving the original file intact, quarantines the file so it can be checked by an administrator, or locks access to the file in some other fashion. Even in an entire book devoted to the subject of network security, you can't learn all you need to know to make a network as secure as possible. New threats are discovered constantly and the changing software landscape makes such information quickly obsolete.

6.3 Cryptography

Cryptography can be defined as the conversion of data into a scrambled code that can be translated and sent across a public or private network. In data and telecommunications, cryptography is necessary when communicating over any un-trusted medium, which includes just about any network, particularly the Internet. Cryptography comes from the Greek words for "secret writing." It has a long and colorful history going back thousands of years.

Historically, four groups of people have used and contributed to the art of cryptography: the military, the diplomatic corps, diarists, and lovers. Of these, the military has had the most important role and has shaped the field over the centuries. Looking back in history we find that Julius Caesar was an early user of cryptography. He sent messages to his troops in a simple but ingeneous method. A letter in the alphabet was replaced by one say 5 positions to the right. So, an "**A**" would be replaced by an "**E**", "**B**" by "**F**" and so on. Hence **RETURN** would become **VJYZVS**. But as it can be seen, this cipher can be easily broken by either figuring out a pattern, by brute force or by getting ones hands on a plaintext and ciphertext combination to deduce the pattern. A basic encryption model is shown in Figure 6-1.

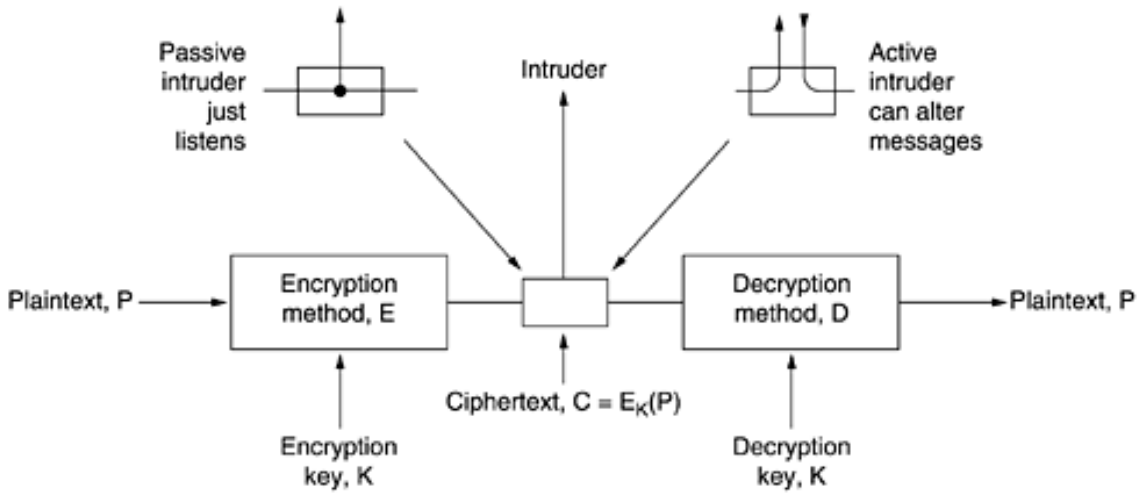

Figure 6-1. The encryption model

The messages to be encrypted, known as the plaintext, are transformed by a function that is parameterized by a key. The output of the encryption process, known as the ciphertext, is then transmitted, often by messenger or radio. We assume that the enemy, or intruder, hears and accurately copies down the complete ciphertext. However, unlike the intended recipient, he does not know what the decryption key is and so cannot decrypt the ciphertext easily. Sometimes the intruder can not only listen to the communication channel (passive intruder) but can also record messages and play them back later, inject his own messages, or modify legitimate messages before they get to the receiver (active intruder). The art of breaking ciphers, called cryptanalysis, and the art devising them (cryptography) is collectively known as cryptology.

6.4 Encryption

As organizations and individuals have connected to the Internet in groups, many have begun eyeing its infrastructure as an inexpensive medium for wide-area and remote connections. The Internet is an international network consisting of individual computers and computer networks that are all interconnected by many paths. Unlike Local Area Networks where access is physically restricted to authorized users, the Internet is a public network and can be accessed by anyone.

Now more than ever, moving vast amounts of information quickly and safely across great distances is one of our most pressing needs. The basic idea of cryptography is to hide information from prying eyes. On the Internet this can be your credit card numbers, bank account information, health/social security information, or personal correspondence with someone else.

The concept behind encryption is quite simple - make the data ineligible for everyone else except those specified. This is done using cryptography - the study of sending 'messages' in a secret form so that only those authorized to receive the 'message' be able to read it. The easy part of encryption is applying a mathematical function to the plaintext and converting it to an encrypted cipher. The harder part is to ensure that the people who are supposed to decipher this message can do so with ease, yet only those authorized are able to decipher it.

We of course also have to establish the legitimacy of the mathematical function used to make sure that it is sufficiently complex and mathematically sound to give us a high degree of safety. The essential concept underlying all automated and computer security application is cryptography. The two ways of going about this process are conventional (or symmetric) encryption and public key (or asymmetric) encryption.

6.4.1 Private Key (Symmetric) Encryption

Private Key encryption also referred to as conventional, single-key or *symmetric encryption* was the only available option prior to the advent of Public Key encryption in 1976. This form of encryption has been used throughout history by Julius Caesar, the Navaho Indians, German U-Boat commanders to present day military, government and private sector applications. It requires all parties that are communicating to share a common key.

A conventional encryption scheme has five major parts:

- ***Plaintext*** - this is the text message to which an algorithm is applied.
- ***Encryption Algorithm*** - it performs mathematical operations to conduct substitutions and transformations to the plaintext.
- ***Secret Key -*** This is the input for the algorithm as the key dictates the encrypted outcome.
- ***Ciphertext -*** This is the encrypted or scrambled message produced by applying the algorithm to the plaintext message using the secret key.
- ***Decryption Algorithm -*** This is the encryption algorithm in reverse. It uses the ciphertext, and the secret key to derive the plaintext message.

When using this form of encryption, it is essential that the sender and receiver have a way to exchange secret keys in a secure manner. If someone knows the secret key and can figure out the algorithm, communications will be insecure. There is also the need for a strong encryption algorithm. What this means is that if someone were to have a ciphertext and a

corresponding plaintext message, they would be unable to determine the encryption algorithm.

There are two methods of breaking conventional/symmetric encryption - brute force and cryptanalysis. Brute force is just as it sounds; using a method (computer) to find all possible combinations and eventually determine the plaintext message. Cryptanalysis is a form of attack that attacks the characteristics of the algorithm to deduce a specific plaintext or the key used. One would then be able to figure out the plaintext for all past and future messages that continue to use this compromised setup.

6.4.2 Public Key Encryption

The year 1976 saw the introduction of a radical new idea into the field of cryptography. This idea centered around the premise of making the encryption and decryption keys different - where the knowledge of one key would not allow a person to find out the other. Public key encryption algorithms are based on the premise that each sender and recipient has a private key, known only to him/her and a public key, which can be known by anyone. Each encryption/decryption process requires at least one public key and one private key. A key is a randomly generated set of numbers/ characters that is used to encrypt/decrypt information.

A public key encryption scheme has six major parts:

- ***Plaintext -*** this is the text message to which an algorithm is applied.
- ***Encryption Algorithm -*** it performs mathematical operations to conduct substitutions and transformations to the plaintext.
- ***Public and Private Keys*** - these are a pair of keys where one is used for encryption and the other for decryption.
- ***Ciphertext -*** this is the encrypted or scrambled message produced by applying the algorithm to the plaintext message using key.
- ***Decryption Algorithm -*** This algorithm generates the ciphertext and the matching key to produce the plaintext.

6.5 Encryption Algorithms

Broadly speaking, there are three types of cryptographic algorithms: secret key algorithms, public key algorithms, and hashing algorithms. Secret key algorithms are symmetric in the sense that both participants in the communication share a single key. In contrast to a pair of participants sharing a single secret key, public key cryptography involves each participant having a private key that is shared with no one else and a public key that is published so everyone knows it. To send a secure message to this participant, you encrypt the message using the widely known public key. The participant then decrypts the message using his or her private key. The third type of cryptography algorithm is called a hash or message digest function. Unlike the preceding two types of algorithms, cryptographic hash functions

typically don't involve the use of keys. Instead, the idea is to map a potentially large message into a small fixed-length number, analogous to the way a regular hash function maps values from a large space into values from a small space.

Different encryption algorithms use proprietary methods of generating these keys and are therefore useful for different applications. The next sections give some details about some of these encryption algorithms. Strong encryption is often distinguished by the key length used by the algorithm. To reemphasize, cryptography algorithms like DES, RSA, and MD5 are just building blocks from which a secure system can be constructed. Figure 6-2 gives a simple taxonomy that illustrates this point.

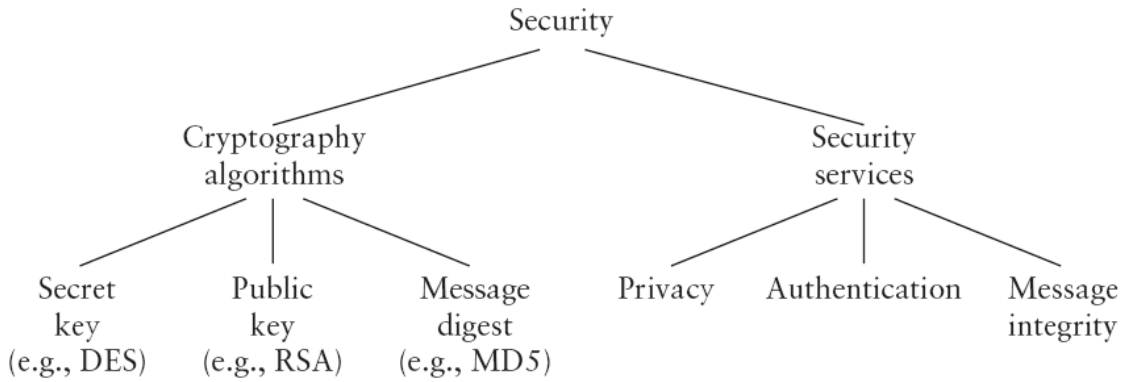

Figure 6-2: Taxonomy of network security

6.5.1 RSA

In 1977, shortly after the idea of a public key system was proposed, three mathematicians, Ron Rivest, Adi Shamir and Len Adleman gave a concrete example of how such a method could be implemented. To honour them, the method was referred to as the RSA Scheme. The system uses a private and a public key. RSA is the most popular method for public key encryption and digital signatures today.

6.5.2 DES/3DES

The Data Encryption Standard (DES) was developed and endorsed by the U.S. government in 1977 as an official standard and forms the basis not only for the Automatic Teller Machines (ATM) PIN authentication but a variant is also utilized in UNIX password encryption. DES is a block cipher with 64-bit block size that uses 56-bit keys. Due to recent advances in computer technology, some experts no longer consider DES secure against all attacks; since then Triple-DES (3DES) has emerged as a stronger method. DES (Data Encryption Standard) is the best-known example of a secret key encryption function.

6.5.3 BLOWFISH

Blowfish is a symmetric block cipher just like DES or IDEA. It takes a variable-length key, from 32 to 448 bits, making it ideal for both domestic and exportable use. Bruce Schneier designed Blowfish in 1993 as a fast, free alternative to the then existing encryption algorithms. Since then Blowfish has been analyzed considerably, and is gaining acceptance as a strong encryption algorithm.

6.5.4 IDEA

International Data Encryption Algorithm (IDEA) is an algorithm that was developed by Dr. X. Lai and Prof. J. Massey in Switzerland in the early 1990s to replace the DES standard. It uses the same key for encryption and decryption, like DES operating on 8 bytes at a time. Unlike DES though it uses a 128 bit key. This key length makes it impossible to break by simply trying every key, and no other means of attack is known. It is a fast algorighm, and has also been implemented in hardware chipsets, making it even faster.

6.5.5 SEAL

Rogaway and Coppersmith designed the Software-optimized Encryption Algorithm (SEAL) in 1993. It is a Stream-Cipher, i.e., data to be encrypted is continuously encrypted. Stream Ciphers are much faster than block ciphers (Blowfish, IDEA, DES) but have a longer initialization phase during which a large set of tables is done using the Secure Hash Algorithm. SEAL uses a 160 bit key for encryption and is considered very safe.

6.5.6 RC4

RC4 is a cipher invented by Ron Rivest, co-inventor of the RSA Scheme. It is used in a number of commercial systems like Lotus Notes and Netscape. It is a cipher with a key size of up to 2048 bits (256 bytes), which on the brief examination given it over the past year or so seems to be a relatively fast and strong cypher. It creates a stream of random bytes and 'XORing' those bytes with the text. It is useful in situations in which a new key can be chosen for each message.

6.6 Authentication Protocols

Authentication is the technique by which a process verifies that its communication partner is who it is supposed to be and not an imposter. Verifying the identity of a remote process in the face of a malicious, active intruder is surprisingly difficult and requires complex protocols based on cryptography. Many people confuse authorization with authentication. Authentication deals with the question of whether you are actually communicating with a specific process. Authorization is concerned with what that process is permitted to do. For example, a client process contacts a file server and says: I am Scott's process and I want to delete the file cookbook.old. From the file server's point of view, two questions must be answered:

1. Is this actually Scott's process (authentication)?
2. Is Scott allowed to delete cookbook.old (authorization)?

Only after both of these questions have been unambiguously answered in the affirmative can the requested action take place. The former question is really the key one. Once the file server knows to whom it is talking, checking authorization is just a matter of looking up entries in local tables or databases. For this reason, we will concentrate on authentication in this section.

There are many different authentication protocols such as:

- Authentication and Key Agreement (AKA)
- CAVE-based_authentication
- Challenge-handshake authentication protocol (CHAP)
- CRAM-MD5
- Diameter
- Extensible Authentication Protocol (EAP)
- Host Identity Protocol (HIP)
- Kerberos
- MS-CHAP and MS-CHAPv2 variants of CHAP
- NTLM, also known as NT LAN Manager
- Password-authenticated key agreement protocols
- Password Authentication Protocol (PAP)
- Protected Extensible Authentication Protocol (PEAP)
- RADIUS
- Secure Remote Password protocol (SRP)
- TACACS and TACACS+
- RFID-Authentication Protocols

6.7 Privacy

6.7.1 Internet Privacy

George Orwell was born as Eric Arthur Blair on June 25, 1903 in Motihari, Bengal, India. He was a prolific writer with such classics as 1984 in 1949 and Animal Farm in 1945. Though the year 1984 came and passed, many of his points regarding the lack of privacy and freedoms are coming true. They might not be as blatant as in his book, but they are being passed as laws and implemented every day. It is not just governments around the world that want to gain this control, but corporations are also looking to gain profits by getting more and more information on their customers.

The questions most often put forward are, '*Is individual* privacy dead?' and '*What is the role of technology as we slip into this type of society?*' Technology has helped us live longer, more fuller lives but its unchecked applications are also threatening our privacy. Luckily, we can use technology to protect ourselves as we go about our daily lives.

A few decades ago, only governments and diplomats used encryption to secure sensitive information. Today, secure encryption on the Internet is the key to confidence for people wanting to protect their privacy, or doing business online. E-Commerce, and secure messaging are just some of the applications that rely on encryption to ensure the safety of data. In many companies that have proprietary or sensitive information, field personnel are required to encrypt their entire laptops fearing that in the wrong hands this information could cause a huge damage.

6.7.2 Identity Theft

Identity theft is a growing problem in today's society. It is relatively easy to pull off and very devastating for the victims. There are thousands of cases every year where people see the fraudulent use of their identity to rack up credit card bills and ruin their reputations and credit histories. The Internet is definitely a factor here and is often pointed to as a culprit. But it can also be used to fight back and ensure that one's privacy is maintained.

6.7.3 Social Engineering

Social Engineering is both incredibly complex and amazingly simple. Social Engineering is defined as a non-technical act of manipulating a person to accomplish goals that may or may not be in the "target's" best interest. This may include obtaining information, gaining access, or getting the target to take certain action. Social Engineering relies heavily on human interaction and often involves tricking other people to break normal security procedures. A social engineer runs what used to be called a "con game".

For example, a person using social engineering to break into a computer network would try to gain the confidence of someone who is authorized to access the network in order to get them to reveal information that compromises the network's security. They might call the authorized employee with some kind of urgent problem; social engineers often rely on the natural helpfulness of people as well as on their weaknesses. Appeal to arrogance, appeal to authority, and old-fashioned snooping are typical social engineering techniques.

Another aspect of social engineering relies on people's inability to keep up with a culture that relies heavily on information technology. Social engineers rely on the fact that people are not aware of the value of the information they possess and are careless about protecting it. Frequently, social engineers will search dustbins for valuable information, memorize access codes by looking over someone's shoulder (shoulder surfing), or take advantage of people's natural inclination to choose passwords that are meaningful to them but can be easily guessed. Security experts propose that as our culture becomes more dependent on information, social engineering will remain the greatest threat to any security system. Prevention includes educating people about the value of information, training them to protect it, and increasing people's awareness of how social engineers operate.

6.8 Packet Filtering

On the Internet, packet filtering is the process of passing or blocking packets at a network interface based on source and destination addresses, ports, or protocols. Packet filtering is a

network security mechanism that works by controlling what data can flow to and from a network. Packet filtering is often part of a firewall program for protecting a local network from unwanted intrusion.

In a software firewall, packet filtering is done by a program called a packet filter. The packet filter examines the header of each packet based on a specific set of rules, and on that basis, decides to prevent it from passing (called DROP) or allow it to pass (called ACCEPT).

There are three ways in which a packet filter can be configured, once the set of filtering rules has been defined. In the first method, the filter accepts only those packets that it is certain are safe, dropping all others. This is the most secure mode, but it can cause inconvenience if legitimate packets are inadvertently dropped. In the second method, the filter drops only the packets that it is certain are unsafe, accepting all others. This mode is the least secure, but it causes less inconvenience, particularly in casual Web browsing. In the third method, if the filter encounters a packet for which its rules do not provide instructions, that packet can be quarantined or the user can be specifically queried concerning what should be done with it. This can be inconvenient if it causes numerous dialog boxes to appear, for example, during Web browsing.

6.8.1 Filtering by Address

The simplest, although not the most common, form of packet filtering is filtering by address. Filtering in this way lets you restrict the flow of packets based on the source and/or destination addresses of the packets, without having to consider what protocols are involved. It's not necessarily safe to trust source addresses because source addresses can be forged. Unless you use some kind of cryptographic authentication between you and the host you want to talk to, you won't know if you're really talking to that host, or to some other machine that is pretending to be that host.

There are two kinds of attacks that rely on forgery: *source address* and *man in the middle*. In a basic ***source address*** forgery attack, an attacker sends you packets that claim to be from someone you trust in some way, hoping to get you to take some action based on that trust, without expecting to get any packets back from you. If the attacker doesn't care about getting packets back from you, he doesn't need to be on the path between you and whoever he is pretending to be; he can be anywhere.

In fact, your responses will go to whomever the attacker is pretending to be, not to the attacker. However, if the attacker can predict your responses, he doesn't need to see them. Many (if not most) protocols are predictable enough for a skilled attacker to be successful at this. There are plenty of attacks that can be carried out without the attacker needing to see the results directly. For example, suppose an attacker issues a command to your system that causes it to email your password file to him; if your system is going to send the attacker the password file in the mail, there is no need for him to see it during the attack itself.

The ***man in the middle*** forgery attack depends on being able to carry out a complete conversation while claiming to be the trusted host. In order to do this, the attacking machine

needs to be able to not only send you packets, but also intercept the packets you reply with. To do this, the attacker needs to do one of the following:

- Insinuate his attacking machine into the path between you and the real machine. This is easiest to do near the ends of the path, and most difficult to do somewhere in the middle, because given the nature of modern IP networks, the path through "the middle" can change at any second.
- Alter the path between the machines so it leads through his attacking machine. This may be very easy or very difficult, depending on the network topology and routing system used by your network, the remote network, and the Internet service providers between those networks.

6.8.1 Filtering by Service

Blocking incoming forged packets, as discussed previously, is just about the only common use of filtering solely by address. Most other uses of packet filtering involve filtering by service. For example, a service that transports all files with .flv or .rar as an extension can be filtered and all the packets containing the relevant information can be dropped.

6.9 Advantages of Packet Filtering

6.9.1 One screening router can help protect an entire network

One of the key advantages of packet filtering is that a single, strategically placed packet filtering router can help protect an entire network. If only one router connects your site to the Internet, you gain tremendous leverage on network security, regardless of the size of your site, by doing packet filtering on that router.

6.9.2 Simple packet filtering is extremely efficient

Because simple packet filtering requires paying attention only to a few packet headers, it can be done with very low overhead. Proxying is a fairly time-consuming operation, and adding proxying means directing connections through another program, usually on a machine that otherwise wouldn't be necessary to the routing process. Packet filtering takes place on a machine that was already in the critical path, and introduces a much smaller delay.

However, there is no free lunch; the more work your packet filters do, the slower they will be. If your packet filters behave like proxies, doing complicated data-driven operations that require keeping track of multiple packets, they will tend to perform like proxies as well.

6.9.3 Packet filtering is widely available

Packet filtering capabilities are available in many hardware and software routing products, both commercial and freely available over the Internet. Most sites already have packet filtering capabilities available in the routers they use. Most commercial router products

include packet filtering capabilities. Packet filtering capabilities are also available for a number of general-purpose computers.

6.10 Disadvantages of Packet Filtering

6.10.1 Current filtering tools are not perfect

Despite the widespread availability of packet filtering in various hardware and software packages, packet filtering is still not a perfect tool. The packet filtering capabilities of many of these products share, to a greater or lesser degree, common limitations.

6.10.2 Packet filtering reduces router performance

Doing packet filtering places a significant extra load on a router. As we discussed previously, more complex filters place more load on the router, but in some cases, simply turning on packet filtering on a given interface can also cost you a lot of performance on some routers, because the filtering is incompatible with certain caching strategies commonly used for performance enhancement.

6.10.3 Some policies can't readily be enforced by normal packet filtering routers

The information that a packet filtering router has available to it doesn't allow you to specify some rules you might like to have. For example, packets say what host they come from but generally not what user. Therefore, you can't enforce restrictions on particular users. Similarly, packets say what port they're going to but not what application; when you enforce restrictions on higher-level protocols, you do it by port number, hoping that nothing else is running on the port assigned to that protocol. Malicious insiders can easily subvert this kind of control.

This problem is eased by using more intelligent packet filters; however, in each case, you have to give up some of the advantages of normal packet filtering. For instance, a packet filter can insist that users authenticate themselves before sending packets, and then it can filter packets by username. However, this removes the transparency advantage of normal packet filtering. A packet filter can also do protocol validity checking, but this is less than perfect and also increases filtering overhead.

6.11 Firewalls

A firewall is a component placed between computers and networks to help eliminate undesired access by the outside world. It can be composed of hardware, software, or a combination of both. A firewall shown as shown in Figure 6-3 is the first line of defense for the network. How firewalls are configured is important.

Firewalls make it possible to filter incoming and outgoing traffic that flows through your system. A firewall can use one or more sets of “rules” to inspect the network packets as they come in or go out of your network connections and either allows the traffic through or blocks it. The rules of a firewall can inspect one or more characteristics of the packets, including but

not limited to the protocol type, the source or destination host address, and the source or destination port.

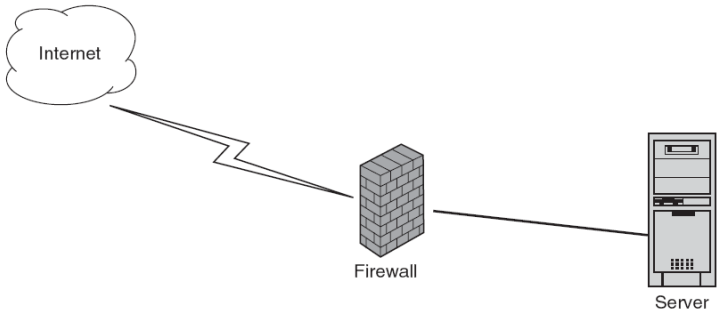

Figure 6-3: Firewall

A firewall may be a hardware device (which may be a computer set up for the task, or a dedicated firewall box that contains a dedicated computer within it) that sits between two networks and enforces network security policies. Generally, firewalls sit between a company LAN and the Internet, but they can also be used between LANs when appropriate.

Very often these days you find that firewalls are used to implement a demilitarized zone (DMZ). This is a network segment located between two firewalls (see Figure 6-4). This is used as a buffer zone to keep the internal network safe from the outside world while offering services that are useful outside of the internal network without allowing the entire network to be available to Internet users. Many times the DMZ contains devices that need Internet access: Web, DNS, and e-mail servers. These servers all have to be hardened to keep them from being attacked by malicious users. Also, care should be taken when choosing what data and services are available on these machines.

Sensitive data should be located on the internal network, not the DMZ. If the DMZ is compromised, you want to be sure that the information that is stored there cannot cause harm to the company should the compromise become a matter of public knowledge. If fact, some states have passed legislation that says you must disclose compromises.

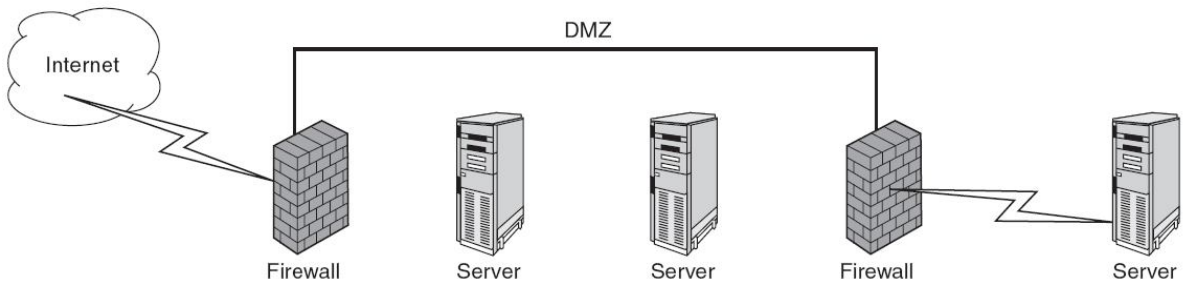

Figure 6-4: A typical secured network with DMZ

That said, firewalls are conceptually very simple devices that can be classified in one of two broad categories: filter-based and proxy-based. We now briefly describe each. To simplify our discussion, we limit ourselves to Internet firewalls.

6.11.1 Filter-Based Firewalls

Filter-based firewalls are the simplest and most widely deployed type of firewall. They are configured with a table of addresses that characterize the packets they will, and will not, forward. By addresses, we mean more than just the destination's IP address, although this is one possibility. Generally, each entry in the table is a 4-tuple: It gives the IP address and TCP (or UDP) port number for both the source and destination.

For example in Figure 6-5, a firewall might be configured to filter (not forward) all packets that match the following description: (192.12.13.14, 1234, 128.7.6.5, 80) This pattern says to filter all packets from port 1234 on host 192.12.13.14 addressed to port 80 on host 128.7.6.5. (Port 80 is the well-known TCP port for HTTP.) Of course it's often not practical to name every source host whose packets you want to filter, so the patterns can include wildcards. For example, (*, *, 128.7.6.5, 80) says to filter all packets addressed to port 80 on 128.7.6.5, regardless of what source host or port sent the packet.

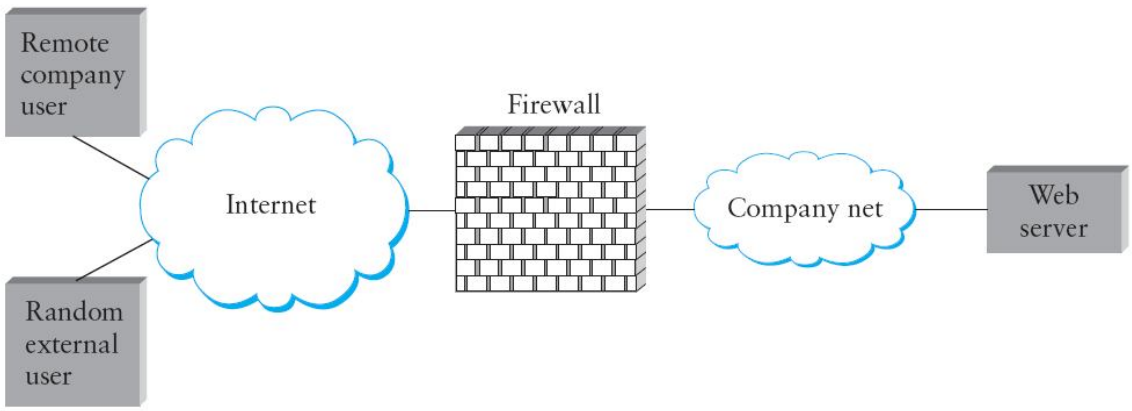

Figure 6-5: Firewall protecting a Web server from some external accesses

6.11.2 Proxy-Based Firewalls

A proxy is a general networking technique that shows up in a number of situations, including firewalls. Generally speaking, a proxy is a process that sits between a client process and a server process. To the client, the proxy appears to be the server; in a sense, the proxy is standing in for the server. To the server, the proxy appears to be the client. Because a proxy imitates both the client and the server, it necessarily has application knowledge built into it. One thing a proxy might do is implement a cache. This allows the proxy to respond to a client request without having to pass the request along to the server. It passes the request on to the server only if it doesn't have the requested item in its cache. Proxies also provide an opportunity to implement a security policy. It is this security role that we now consider.

To understand how a proxy-based firewall works—and why you would want one—consider a corporate Web server, where the company wants to make some of the server's pages accessible to all external users (i.e., it won't work to simply program the firewall to block all external access to HTTP's well-known port 80), but it wants to restrict certain of the pages to corporate users at one or more remote sites. This situation is illustrated in Figure 6-5. There is no way to express this policy as a filter since it depends on the URL contained in each HTTP request.

The solution is to put an HTTP proxy on the firewall. Remote users establish an HTTP/TCP connection to the proxy, which looks at the URL contained in the request message. If the requested page is allowed for the source host, the proxy establishes a second HTTP/TCP connection to the server and forwards the request on to the server. The proxy then forwards the response in the reverse direction between the two TCP connections. This situation is depicted in Figure 6-6. If the request is not allowed, the proxy does not create this second connection, but instead returns an error to the source. In a sense, the firewall dynamically decides what packets to forward and what packets to drop, with the policy embodied in the application-specific proxy.

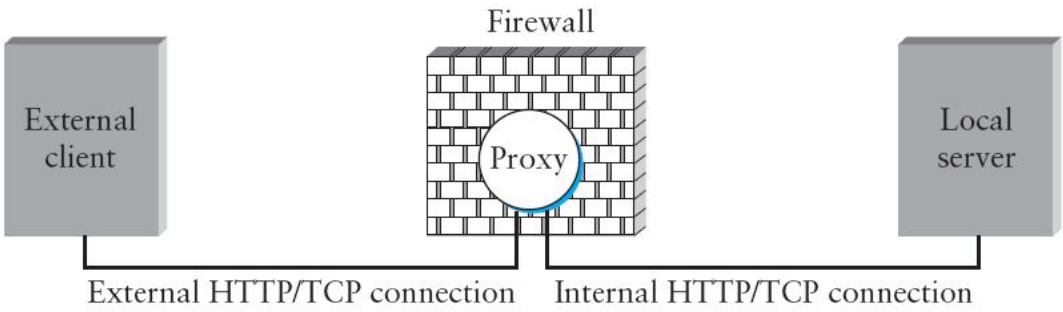

Figure 6-6: HTTP proxy mediating access to a corporate Web server

There are several things to notice about this example. First, the proxy has to understand the HTTP protocol in order to respond to the client. Second, once an HTTP proxy is in place for security reasons, it might be extended to decide which of many local Web servers to forward a given request to, perhaps in an effort to balance the load among the servers. It might also cache hot Web pages, as suggested above. Third, proxies can be defined for applications other than HTTP; for example, FTP and Telnet proxies are quite common.

Finally, proxy-based firewalls can be characterized as being either transparent or classical. A transparent proxy, as the name implies, is not explicitly visible to either the sender or the receiver; it just happens to intercept messages that flow through it. In contrast, the source purposely addresses messages to a classical proxy, which then forwards the message to the ultimate destination.

Consider the simple network shown in Figure 6-7, in which source S sends a message to receiver R through proxy P. If P is transparent, then S addresses the message to R, and the message just happens to pass through P en route to R. P either forwards the message to R, or

not. With a classical firewall, S does not know about R, but instead addresses the message to P. In other words, P acts as an addressable front door to the site. When the message arrives at P, it selects a node "behind it" to which it forwards the message.

Figure 6-7: Simple Internet

6.12 Transport Layer Protocol

Transport Layer Security (TLS) is a protocol that ensures privacy between communicating applications and their users on the Internet. When a server and client communicate, sending unencrypted messages increases the risk that messages can be intercepted or altered. TLS security technology automatically encrypts e-mail messages between servers thereby reducing the risk of snooping, interception, and alteration.

TLS is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. The TLS Record Protocol provides connection security with some encryption method such as the Data Encryption Standard (DES). The TLS Record Protocol can also be used without encryption. The TLS Handshake Protocol allows the server and client to authenticate each other and to negotiate an encryption algorithm and cryptographic keys before data is exchanged.

TLS is an IETF (Internet Engineering Task Force) standard for communicating e-mail securely. When TLS is enabled on the mail servers of both the sender and the receiver of the e-mail, information exchanged between the servers is encrypted in a format that encodes plain text into non-readable form. Mail servers use Simple Mail Transfer Protocol (SMTP) to send and receive messages. When sending encrypted messages, the mail exchange works as follows:

- Each company's e-mail gateway is configured to enable TLS communications for SMTP traffic
- When the sending party (client) connects to the receiving party (server), the sending party checks whether TLS services are offered
- If the receiver offers TLS services, the sender initiates a TLS handshake. The server sends its TLS certificate to the client
- If the sender trusts the certificate of the receiver, a TLS session encryption key is negotiated, the TLS session starts, and the SMTP message is transmitted

6.12.1 Advantages of TLS

E-mail over TLS provides the following advantages compared to traditional (unencrypted) e-mail:

- Protection. E-mail servers can be configured to enforce TLS encryption between named parties and confidential information can be exchanged with reduced risk of snooping or interception
- Every e-mail sent and received is encrypted. When TLS is enforced, no individual review or decision is required to determine whether or not to encrypt an e-mail based on the email's content.
- E-mail encryption is transparent to both the sender and the receiver. Both parties send and read e-mails the same way as they do today.
- TLS is globally accepted and currently available on most, if not all, e-mail servers.
- Industry Standard. There is a growing trend among financial, academic and other institutions to use TLS. Many of these institutions have already implemented TLS.
- E-mail can be easily inspected for viruses. With SMTP over TLS, encryption terminates at partners' e-mail gateways. This means that after messages move inside a company's DMZ firewall, they can be treated just like regular SMTP traffic. Messages can be inspected, scanned and analyzed for malicious content to comply with corporate security policies. This is in sharp contrast to PGP- or S/MIME-style encryption schemes, in which messages are decrypted only at the point of receipt.
- Reduced cost. When company-to-company encryption over TLS is in place, tactical person-to-person systems for encrypting messages are no longer needed. In addition, companies need only purchase TLS certificates for servers, rather than large numbers of enterprise S/MIME certificates for all clients. There typically is no out-of-pocket cost to implement TLS, although there is some effort to set up and test TLS on the server, as there is no need to purchase any software.
- No overhead for end-users. Because no special software is installed on client machines, TLS encryption is "always on" for compliant partners; the process is completely transparent to end-users.
- Rapid deployment. Workstations do not require any additional configuration; only servers need to be modified. The configuration process is also straightforward. Time to value is measured in days and weeks, not months and years.