1

Introduction

Internet security is certainly a hot topic these days. What was once a small research network, a home for greybeard researchers and future millionaire geeks, is now front-page material. Internet security has been the subject of movies, books, and real-life thrillers.

The Internet itself is an entirely new thing in the world: a marketplace, a backyard fence, a kind of library, even a telephone. Its growth has been astounding, and the Web is ubiquitous. We see URLs on beer bottles and TV commercials, and no movie trailer would be complete without one.

The Internet is a large city, not a series of small towns. Anyone can use it, and use it nearly anonymously.

The Internet is a bad neighborhood.

1.1 Security Truisms

We have found that Internet security is not very different from other forms of security. The same concepts used to design castles apply to the construction of a Web server that offers access to a corporate database. The details are different, and the technical pieces are quite different, but the same approaches, rules, and lessons apply.

We present here some important maxims to keep in mind. Most have stood the test of thousands of years.

There is no such thing as absolute security.

We can raise the attacker's cost of breaching our security to a very high level, but absolute guarantees are not possible. Not even nuclear launch codes are absolutely secure; to give just one example, a U.S. president once left the codes in a suit that was sent off for cleaning [Feaver, 1992].

This fact should not deter connection to the Internet if you need the access. Banks don't have perfect security either; they are subject to robberies, fraud, and embezzlement. Long experience

has taught banks which security measures are cost-effective, and they can account for these expected loses in their business plans. Much of the remainder is covered by insurance.

The Internet is new, so the risks are less well understood. As more services are connected, we will get a better idea of which measures are most effective, and what expected losses may occur. The chief problem is that the net offers such fat targets to anonymous attackers.

Security is always a question of economics.

What is the value of what you are protecting? How much time, effort, money, and risk are your opponents willing to spend to get through your defenses?

One spook we know reports that there is a $100,000,000 surveillance device that can be thwarted with something you can buy in a hardware store for $40. This is the kind of leverage we defenders have in our favor—small steps can raise big barriers.

Keep the level of all your defenses at about the same height.

It makes no sense to fit a bank vault with a screen door in the back, yet we have seen equivalent arrangements on the Internet. Don't waste time and money on one part of your defenses if other parts have glaring weaknesses. A firewall makes little sense if the perimeter has numerous breaches. If you don't check the contents of parcels leaving the building, is it worth blocking outgoing *ftp* connections?

There are many factors to Internet security. Is the firewall secure? Are your people trained to resist "social engineering" attacks (see Section 5.2)? Can you trust your people, and how far? Are there holes in the perimeter? Are there back doors into your systems? Has the janitor sold out to your opponents?

An attacker doesn't go through security, but around it.

Their goal is to find and exploit the weakest link.

Put your defenses in layers.

This is called the *belt-and-suspenders* approach, or *defense in depth*. If one layer fails, perhaps the backup will save you. The layers can take many different forms, and are often conceptual, rather than physical.

This concept has been a vital component of security for thousands of years. Most castles have more than one wall. For example, one of the authorized roads into Edo Castle in Tokyo was protected by three *banshos*, or guard-houses; the samurai there were charged with watching the retinues of visiting dignitaries. The typical immune system has many overlapping components, and some are redundant.

*It's a bad idea to rely on "*security through obscurity.*"*

You should assume that your adversaries know all of your security arrangements; this is the safest assumption. It's okay to keep your setup secret—that's another layer your opponent has

to surmount—but don't make that your only protection. The working assumption at the *National Security Agency (NSA)* is that serial number 1 of any new device is hand-delivered to the enemy. Secrets often end up pasted to terminals, or in a corporate dumpster.

Sometimes the *appearance* of good security will be enough to help deter attackers. For example, the Great Wall of China is a familiar image, and an icon of security. It deterred many attacks, and suppressed unwanted trade, which was one of its design goals. Some parts, however, used rice for mortar, and we have heard that some remote parts of the Wall were simply piles of rock and earth. Such cheats remind us of some contemporary security arrangements. Ghengis Kahn marched through the gates of the wall and into Beijing without trouble; insiders had paved the way for him.

We advocate security without these cheats. It's a good sign if you can't reach a host you are working on because the only way in is broken somehow, and even you don't have a back door.

Keep it simple.

To paraphrase Einstein: Make your security arrangements as simple as possible, but no simpler. Complex things are harder to understand, audit, explain, and get right. Try to distill the security portions into simple, manageable pieces. Complicated security measures often are often not fail-safe.

Don't give a person or a program any more privileges than those necessary to do the job.

In the security field, this is called *least privilege*, and it's a very important concept. A common example of this is the valet key for a car, which lets the valet drive the car, but won't open the trunk or glove box.

Programming is hard.

This quote of Dijkstra is still true. It is very hard to write bug-free programs, and the difficulty increases by some power of the program size. We like crucial security programs to be about a page long. Huge security-sensitive programs have been a constant and reliable source of security problems.

Security should be an integral part of the original design.

Security that is added after the initial design is seldom as reliable. The designer must keep the security assumptions in mind at the design stage or something will be overlooked. Changing security assumptions later on is a surefire source of security trouble. (On the other hand, networks aren't static, either; as you change your network, be sure to examine it for new vulnerabilities.)

If you do not run a program, it does not matter if it has security holes.

Exposed machines should run as few programs as possible; the ones that are run should be as small as possible. Any program, no matter how innocuous it seems, can harbor security holes.

(Who would have guessed that on some machines, integer divide exceptions[1] could lead to system penetrations?)

A program or protocol is insecure until proven secure.

Consequently, we configure computers in hostile environments to reject everything, unless we have explicitly made the choice—and accepted the risk—to permit it. Taking the opposite tack, of blocking only known offenders, has proven extremely dangerous.

A chain is only as strong as its weakest link.

An attacker often needs to find only one weakness to be successful. The good news is that we can usually detect attempts to find the weak link, if we want to. (Alas, most people don't take the time.)

Security is a trade-off with convenience.

It is all but impossible to use technical means to enforce more security than the organizational culture will permit—and most organizational cultures are not very receptive to security systems that get in the way. Annoyed computer users are a major source of security problems. If security measures are onerous, they will go around them, or get angry, or complain to management. (Even intelligence agencies experience this.) Our job as security people is to make the security both as strong and as unobtrusive as possible.

Well-designed security doesn't have to be onerous. Good design and appropriate technology can make security almost painless. The modern hotel door lock contains a computer and perhaps a network connection to a central control room. It is no longer a security problem for the hotel if you forget to turn in your key. The hotel can keep track of its own employees when they enter a room. There are even warnings when a door is left ajar for an unusual length of time. The guest still needs to carry a key, but it works much better. Automobile locks are getting so good that the thief has to physically remove the entire car—a teenager can't hot-wire it anymore. Soon, we will have transmitters and no keys at all. (Of course, transmitters have evolved, too, as car thieves have discovered scanners and replay attacks.)

Don't underestimate the value of your assets.

Often, common everyday data is underestimated. Mundane data can be very important. It is said that pizza shop owners around the Pentagon can tell when a major military action is afoot: They get numerous calls late at night. A reporter we know asserted that he had no sensitive information on his computer. We reminded him of his contact lists, story ideas, partial stories, and so on. Could his competitor across town use this information?

1. See CERT Advisory CA-1992:15, July 21, 1992.

1.2 Picking a Security Policy

> Even paranoids have enemies.
>
> —ANONYMOUS

The idea of creating a security policy may smack of bureaucracy to some, especially an eager technocrat. It brings to mind thick books of regulations and rules that must be read, understood, and followed. While these may have their place, it's not what we are talking about here.

A *security policy* is the set of decisions that, collectively, determines an organization's posture toward security. More precisely, a security policy delimits the boundaries of acceptable behavior, and what the response to violations should be. Naturally, security policies will differ from organization to organization. An academic department in a university has different needs than a corporate product development organization, which in turn differs from a military site. Every organization should have one, however, if only to let it take action when unacceptable events occur.

Your security policy may determine what legal recourse you have if you are ever attacked. In some jurisdictions, a welcome screen has been interpreted as an invitation to guest users. Furthermore, logging policy may determine whether specific logs are admissible as evidence.

You must first decide what is and is not permitted. To some extent, this process is driven by the business or structural needs of the organization. Thus, some companies may issue an edict that bars personal use of corporate computers. Some companies wish to restrict outgoing traffic, to guard against employees exporting valuable data. Other policies may be driven by technological considerations: A specific protocol, though undeniably useful, may not be used because it cannot be administered securely. Still others are concerned about employees importing software without proper permission: a company doesn't want to be sued for infringing on someone else's rights. Making such decisions is clearly an iterative process, and one's choices should never be carved in stone (or etched into silicon).

It is hard to form these policies, because they boil down to specific services, which can be highly technical. You often need someone with both the clout of a CEO and the expertise of a security wizard. The wizard alone can't do it; security policies can often be trumped by business plans [Schneier, 2000].

1.2.1 Policy Questions

To devise a security policy, you must answer several questions. The first question is obvious:

> *What resources are you trying to protect?*

The answer is not always obvious. Is it the CPU cycles? At one time, that made a great deal of sense; computer time was very expensive. That is no longer true in most situations, supercomputers being a notable exception.

More seriously, a host—or rather, a host running certain software with certain configuration files—has a name, an identity, that lets it access other, more critical resources. A hacker who

compromises or impersonates a host will usually have access to all of its resources: files, storage devices, cryptographic keys, and so on. A common goal is to eavesdrop on Net traffic that flows past the host. Some hackers are most interested in abusing the identity of the host, not so much to reach its dedicated resources, but to launder further outgoing connections to other, more interesting, targets. Others might actually be interested in the data on your machine, whether it is sensitive company material or government secrets.

The answer to this first question will dictate the host-specific measures that are needed. Machines with sensitive files may require extra security measures: stronger authentication, keystroke logging and strict auditing, or even file encryption. If the target of interest is the outgoing connectivity, the administrator may choose to require certain privileges for access to the network. Maybe all such access should be done through a daemon or proxy that will perform extra logging.

Often one wants to protect all such resources. The obvious answer is to stop the attackers at the front door, i.e., not let them into the computer system in the first place. Such an approach is always a useful start, although it tacitly assumes that one's security problems originate from the outside.

This leads us to our second major question:

Who is interested in attacking you?

Techniques that suffice against a teenager with a modem are quite useless against a major intelligence agency. For the former, mild encryption might do the trick, whereas the latter can and will resort to wiretapping, cryptanalysis, monitoring spurious electronic emissions from your computers and wires, and even "black-bag jobs" aimed at your machine room. (Do not underestimate the teenager, though. He might get the coveted midnight-to-eight janitorial shift in your machine room [Voyager, 1994].) Furthermore, the intelligence agency may well try the easy stuff first.

Computer security is not a goal, it is a means toward a goal: information security. When necessary and appropriate, other means should be used as well. The strength of one's computer security defenses should be proportional to the threat. *Other defenses, though beyond the scope of this book, are needed as well.*

The third question one must answer before deploying a security mechanism represents the opposite side of the coin:

How much security can you afford?

Part of the cost of security is direct financial expenditures, such as the extra routers, firewalls, software packages, and so on. Often, the administrative costs are overlooked. There is another cost, however, a cost in convenience and productivity, and even morale. Too much security can hurt as surely as too little can. Annoyed by increases in security, good people have left companies. Finding the proper balance is tricky, but utterly necessary—and it can only be done if you have properly assessed the risk to your organization from either extreme.

One more point is worth mentioning. Even if you do not believe you have valuable assets, it is still worth keeping hackers out of your machines. You may have a relaxed attitude, but that may not be evident to the attackers. There are far too many cases on record of systems being trashed by hackers who thought they had been detected. (Someone even tried it on us; see Chapter 16.)

1.2.2 Stance

> The moral of this story is, anything you don't understand is dangerous until you do understand it.
>
> Beowulf Schaefer in *Flatlander*
> —LARRY NIVEN

A key decision in the policy is the *stance* of your design. The stance is the attitude of the designers. It is determined by the cost of failure and the designers' estimate of that likelihood. It is also based on the designers' opinions of their own abilities. At one end of the scale is a philosophy that says, "We'll run it unless you can show me that it's broken." People at the other end say, "Show me that it's both safe and necessary; otherwise, we won't run it." Those who are completely off the scale prefer to pull the plug on the network, rather than take any risks at all. Such a move might be desirable, but it is usually impractical these days. Conversely, one can best appreciate just how little confidence the U.S. military has in computer security techniques by realizing that connecting machines containing classified data to unsecured networks is forbidden.

(There's another lesson to be learned from the military: Their unclassified machines are connected, and have been attacked repeatedly and with some success. Even though the data is (probably) not classified, it is sensitive and important. Don't underestimate the value of your data. Furthermore, don't rely on air gaps too much; users often rely on "sneaker-net" when they need to move some data between the inside net and the outside one. There are reliable reports of assorted viruses making their way *into* classified networks, and the spooks clam up when you ask if viruses have ever made their way *out*.)

In general, we have leaned toward the paranoid end of the scale (for our corporate environment, we should stress). In the past, we've tried to give our firewalls a fail-safe design: If we have overlooked a security hole or installed a broken program, we believe our firewalls are still safe. This is defense in depth. Compare this approach to a simple packet filter. If the filtering tables are deleted or installed improperly, or if there are bugs in the router software, the gateway may be penetrated. This non-fail-safe design is an inexpensive and acceptable solution if your stance allows a somewhat looser approach to gateway security. In recent years, we've eased our stance on our corporate firewalls. A very tight firewall was inconsistent with the security of our large and growing corporate perimeter.

We do not advocate disconnection for most sites. Most people don't think this is an option anymore. Our philosophy is simple: there are no absolutes. (And we believe that absolutely...) One cannot have complete safety; to pursue that chimera is to ignore the costs of the pursuit. Networks and internetworks have advantages; to disconnect from a network is to deny oneself those advantages. When all is said and done, disconnection may be the right choice, but it is a decision that can only be made by weighing the risks against the benefits.

In fact, disconnection can be self-defeating. If security is too onerous, people will go around it. It is easy to buy a modem and establish a personal IP link.

We advocate caution, not hysteria. For reasons that are spelled out below, we think that firewalls are an important tool that can minimize the risk, while providing most of the benefits of a network connection.

Whether or not a security policy is formally spelled out, one always exists. If nothing else is said or implemented, the default policy is "anything goes." Needless to say, this stance is rarely acceptable in a security-conscious environment. *If you do not make explicit decisions, you have made the default decision to allow almost anything.*

It is not for us to decree what services are or are not acceptable. As stated earlier, such decisions are necessarily context-dependent. The rules we have provided, however, are universal.

1.3 Host-Based Security

If a host is connected to a network, it ought to be up to the host to protect itself from network-borne abuses. Many opponents of firewalls espouse this, and we don't disagree—in theory. It is possible to tighten up a host to a fair degree, possibly far enough that attackers will resort to other less-convenient and less-anonymous avenues of attack.

The problem is that most commercial systems are sold with glaring security holes. Most of the original suite of traditional Internet services are unsafe to some degree. The vendors sell systems this way because these services are popular and useful. Traditional UNIX workstations come with dozens of these services turned on. Routers are generally administered through the *telnet* service, which is subject to at least two easy attacks. Even PCs, which used to be too dumb to have dangerous services, are now beginning to offer them. For example, at least two different packages allow even a Windows 95 or 98 machine to host a simple Web server. Both of these have had very serious security holes. Modern versions of Windows run many more services, resulting in many more potential holes. (Do you know what services are running on your corporate Windows machines? Do you know how to find out, how to disable them, and how to do it reliably on all such machines, including every new one that is delivered? Can you tell if some user has turned a service back on? Do you know what new functions are enabled by vendor service packs?)

The hosts that tend to be safer include the commercial firewalls, which were originally built with security as their primary goal, and *multilevel secure systems (MLSs)*, for the same reason.

The software market is starting to offer relatively secure services. The *Secure Socket Layer (SSL)* provides reasonably easy access to encrypted connections, and numerous similar attempts are evolving.

The old services persist, however. Most hosts in an administrative zone trust one another, so one weak link can compromise the whole cluster. We suspect that it will be a long time before this general situation is improved, so we must resort to perimeter security.

1.4 Perimeter Security

If it is too difficult to secure each house in a neighborhood, perhaps the residents can band together to build a wall around the town. Then the people need fear only themselves, and an invading force

that is strong enough to breach the wall. Alert, well-trained guards can be posted at the gates while the people go about their business. Similarly, the king's residence can be enclosed in another wall, adding an additional layer of defense (at least for the king).

This approach is called *perimeter security*, and it is very important on the Internet. It has two components: the wall and the gate. On the Internet, the gate is implemented with a *firewall*, a configuration of machines and software that allows the townspeople to do their business, without letting the Bad Guys in. To be effective, the wall should go all the way around the town, and be high enough and thick enough to withstand attack. It also must not have holes or secret entrances that allow an attacker to creep in past the guards.

The perimeter approach is not effective if the town is too large. The protected "towns" on the Internet are growing as quickly as the Internet as a whole. Just before it split into three companies, AT&T had several times as many hosts "inside" its perimeter as the entire Internet had when the Morris Worm was released in 1988. No one individual knew the location, the policies, the security, or the connectivity of all of these hosts. Lack of knowledge alone can call into question a perimeter defense.

1.5 Strategies for a Secure Network

1.5.1 Host Security

To some people, the very notion of a firewall is anathema. In most situations, the network is not the resource at risk; rather, it is the endpoints of the network that are threatened. By analogy, con artists rarely steal phone service *per se*; instead, they use the phone system as a tool to reach their real victims. So it is, in a sense, with network security. Given that the target of the attackers is the hosts on the network, should they not be suitably configured and armored to resist attack?

The answer is that they should be, but probably cannot. There *will* be bugs, either in the network programs or in the administration of the system. It is this way with computer security: the attacker only has to win once. It does not matter how thick are your walls, nor how lofty your battlements; if an attacker finds one weakness—say, a postern gate (back door), to extend our metaphor—your system *will* be penetrated. Unfortunately, that is not the end of your troubles.

By definition, networked machines are not isolated. Typically, other machines will trust them in some fashion. It might be the almost-blind faith of *rlogin*, or it might be the sophisticated cryptographic verification used by the Kerberos authentication system [Bryant, 1988; Kohl and Neuman, 1993; Miller *et al.*, 1987; Steiner *et al.*, 1988], in which case a particular user will be trusted. It doesn't matter—if the intruder can compromise the system, he or she will be able to attack other systems, either by taking over *root*, and hence the system's identity, or by taking over some user account. This is called *transitive trust*.

It might seem that we are unduly pessimistic about the state of computer security. This is half-true: we are pessimistic, but not, we think, unduly so. Nothing in the recent history of either network security or software engineering gives us any reason to believe otherwise. Nor are we alone in feeling this way.

Consider, for example, the famous *Orange Book* [Brand, 1985]. The lists of features for each security level—auditing, access controls, trusted path, and the like—got all the attention,

Boom!

Not all security holes are merely bad. Some are truly horrendous. We use a "bomb" symbol to indicate a particularly serious risk. That doesn't mean you can be sanguine about the others—the intruders don't care much how they get in—but it does provide some rough guidance about priorities.

but the higher levels also have much more stringent assurance requirements. That is, there must be more reason to believe that the system actually functions as designed. (The Common Criteria [CC, 1999] made this distinction even clearer.) Despite those requirements, even the most trusted system, with an **A1** evaluation, is not trusted with the most sensitive information if uncleared users have access to the system [Neugent and Olson, 1985]. Few systems on the Internet meet even the **C2** requirements; their security is not adequate.

Another challenge exists that is totally unrelated to the difficulty of creating secure systems: administering them. No matter how well written the code and how clean the design, subsequent human error can negate all of the protections. Consider the following sequence of events:

1. A gateway machine malfunctioned on a holiday weekend, when none of the usual system administrators was available.
2. The backup expert could not diagnose the problem over the phone and needed a guest account created.
3. The operator added the account *guest*, with no password.
4. The expert neglected to add a password.
5. The operator forgot to delete the account.
6. Some university students found the account within a day and told their friends.

Unlikely? Perhaps, but it happened to one of our gateways. The penetration was discovered only when the unwanted guests happened to trigger an alarm while probing our other gateway machine.

Our firewall machines are, relatively speaking, simple to administer. They run minimal configurations, which in and of itself eliminates the need to worry about certain things. Off-the-shelf machines have lots of knobs, buttons, and switches with which to fiddle, and many of the settings are insecure. Worse yet, many are shipped that way by the vendor; higher security generally makes a system less convenient to use and administer. Some manufacturers choose to position their products for the "easy-to-use" market. Our internal network has many machines that are professionally administered. However, it also has many departmental machines that are unpacked, plugged in,

turned on, and thereafter all but ignored. These machines run old releases of the operating system, with bugs that are fixed if and only if they directly affect the user population. If the system works, why change it? A reasonable attitude much of the time, but a risky one, given the intertwined patterns of transitive network trust.

(Even a firewall may not be secure. Many firewalls are add-on packages to off-the-shelf operating systems. If you haven't locked down the base platform, it may be susceptible to attack. Apart from that, some firewalls are themselves quite complex, with numerous programs running that must pass very many protocols through the firewalls. Are these programs correct? Is the administration of this complex configuration correct? We hope so, but history suggests otherwise.)

1.5.2 Gateways and Firewalls

'Tis a gift to be simple,
'Tis a gift to be free,
'Tis a gift to come down where we ought to be,
And when we find ourselves in the place just right,
It will be in the valley of love and delight.
When true simplicity is gained,
To bow and to bend, we will not be ashamed
To turn, turn, will be our delight,
'Til by turning, turning, we come round right.

—SHAKER DANCE SONG

By this point, it should be no surprise that we recommend using firewalls to protect networks. We define a firewall as a collection of components placed between two networks that collectively have the following properties:

- All traffic from inside to outside, and vice-versa, must pass through the firewall.
- Only authorized traffic, as defined by the local security policy, will be allowed to pass.
- The firewall itself is immune to penetration.

We should note that these are design goals; a failure in one aspect does not mean that the collection is not a firewall, but that it is not a very good one.

That firewalls are desirable follows directly from our earlier statements. Many hosts—and more likely, most hosts—*cannot* protect themselves against a determined attack. Firewalls have several distinct advantages.

The biggest single reason that a firewall is likely to be more secure is simply that it is not a general-purpose host. Thus, features that are of doubtful security but add greatly to user convenience—NIS, *rlogin*, and so on—are not necessary. For that matter, many features of unknown security can be omitted if they are irrelevant to the firewall's functionality.

A second benefit comes from having professional administration of the firewall machines. We do not claim that firewall administrators are necessarily more competent than your average system

administrator. They may be more security conscious. However, they are almost certainly better than non-administrators who must nevertheless tend to their own machines. This category would include physical scientists, professors, and the like, who (rightly) prefer to worry about their own areas of responsibility. It may or may not be reasonable to demand more security consciousness from them; nevertheless, it is obviously not their top priority.

A third benefit is that commercial firewalls are designed for the job. People can build fairly secure machines when there is a commercial need for it. Occasionally, they are broken, but usually they fail when misconfigured.

A firewall usually has no normal users. This is a big help: users can cause many problems. They often choose poor passwords, a serious risk. Without users, one can make more or less arbitrary changes to various program interfaces if that would help security, without annoying a population that is accustomed to a different way of doing things. One example is the use of handheld authenticators for logging in (see Chapter 7). Many people resent them, or they may be too expensive to be furnished to an entire organization. A gateway machine should have a restricted-enough user community that these concerns are negligible.

Gateway machines have other, nonsecurity advantages as well. They are a central point for mail, FTP, and Web administration, for example. Only one machine need be monitored for delayed mail, proper header syntax, spam control, alias translation, and so on. Outsiders have a single point of contact for mail problems and a single location to search for files being exported.

Our main focus, though, is security. For all that we have said about the benefits of a firewall, it should be stressed that we neither advocate nor condone sloppy attitudes toward host security. Even if a firewall were impermeable, and even if the administrators and operators never made any mistakes, the Internet is not the only source of danger. Apart from the risk of insider attacks—and in many environments, that is a serious risk—an outsider can gain access by other means. Often, a hacker has come in through a modem pool, and attacked the firewall from the *inside* [Hafner and Markoff, 1991]. Strong host security policies are a necessity, not a luxury.

For that matter, *internal* firewalls are a good idea, to protect very sensitive portions of organizational networks. As intranets grow, they become harder to protect, and hence less trustworthy. A firewall can protect your department from intruders elsewhere in the company. Schools must protect administrative computers containing grades, payroll, and alumni data from their general student population. We expect this Balkanization of intranets to increase.

1.5.3 DMZs

Some servers are difficult to trust because of the size and the complexity of the code they run. Web servers are a classic example. Do you place your external Web server inside the firewall, or outside? If you place it inside, then a compromise creates a launch point for further attacks on inside machines. If you place it outside, then you make it even easier to attack. The common approach to this is to create a *demilitarized zone (DMZ)* between two firewalls. (The name is a poor one—it's really more like a no-man's land—but the phrase is common terminology in the firewall business.) Like its real-world analog in Korea, the network DMZ needs to be monitored carefully, as it is a place where sensitive objects are exposed to higher risk than services all the way on the inside.

It is important to carefully control administrative access to services on the DMZ. Most likely, this should only come from the internal network, and preferably over a cryptographically protected connection, such as *ssh*.

A DMZ is an example of our general philosophy of defense in depth. That is, multiple layers of security provide a better shield. If an attacker penetrates past the first firewall, he or she gains access to the DMZ, but not necessarily to the internal network. Without the DMZ, the first successful penetration could result in a more serious compromise.

You should not fully trust machines that reside in the DMZ—that's the reason we put them there. Important Web servers may need access to, say, a vital internal database, but ensure that the database server assumes that queries may come from an untrusted source. Otherwise, an attacker may be able to steal the crown jewels via the compromised Web server. We'll stress this point again and again: *Nothing* is completely secure, but some situations need more care (and more defenses) than do others.

1.5.4 Encryption—Communications Security

Encryption is often touted as the ultimate weapon in the computer security wars. It is not. It is certainly a valuable tool (see Chapter 18), but if encryption is used improperly, it can hurt the real goals of the organization.

The difference here is between *cryptography*, the encryption methods themselves, and the application or environment using the cryptography. In many cases, the cryptographic system doesn't need to be cracked, just evaded. You don't go through security, you go around it.

Some aspects of improper use are obvious. One must pick a strong enough cryptosystem for the situation, or an enemy might cryptanalyze it. Similarly, the key distribution center must be safeguarded, or all of your secrets will be exposed. Furthermore, one must ensure that the cryptographic software isn't buggy; that has happened, too (see *e.g.,* CERT Advisory CA-1995-03a, CERT Advisory CA-1998-07, CERT Advisory CA-1999-15, CERT Advisory CA-2002-23, and CERT Advisory CA-2002-27).

Other dangers exist as well. For one thing, encryption is best used to safeguard file transmission, rather than file storage, especially if the encryption key is generated from a typed password. Few people bequeath knowledge of their passwords in their wills; more have been known to walk in front of trucks. There are schemes to deal with such situations (e.g., [Shamir, 1979; Gifford, 1982; Blaze, 1994]), but these are rarely used in practice. Admittedly, you may not be concerned with the contents of your files after your untimely demise, but your organization—in some sense the real owner of the information you produce at work—might feel differently.

Even without such melodrama, if the machine you use to encrypt and decrypt the files is not physically secure, a determined enemy can simply replace the cryptographic commands with variants that squirrel away a copy of the key. Have you checked the integrity of such commands on your disk recently? Did someone corrupt your integrity-checker? Or perhaps someone is logging keystrokes on your machine.

Finally, the biggest risk of all may be your own memory. Do you remember what password you used a year ago? (You do change your password regularly, do you not?) You used that password every day; how often would you use a file encryption key?

If a machine is physically and logically secure enough that you can trust the encryption process, encryption is most likely not needed. If the machine is not that secure, encryption may not help. A smart card may protect your keys, which is good; however, an attacker who has penetrated your machine may be able to ask your smart card to decrypt your files.

There is one exception to our general rule: backup tapes. Such tapes rarely receive sufficient protection, and there is never any help from the operating system. One can make a very good case for encrypting the entire tape during the dump process—*if* there is some key storage mechanism guaranteed to permit you to read the year-old backup tape when you realize that you are missing a critical file. It is the *information* that is valuable; if you have lost the contents of a file, it matters little if the cause was a hacker, a bad backup tape, a lost password, or an errant *rm* command.

1.6 The Ethics of Computer Security

Sed quis custodiet ipsos custodes? (But who will guard the guards themselves?)

Satires, VI, line 347
—JUVENAL, C. 100 C.E.

At first blush, it seems odd to ask if computer security is ethical. We are, in fact, comfortable with what we are doing, but that is because we have asked the question of ourselves, and then answered it to our own satisfaction.

There are several different aspects to the question. The first is whether or not computer security is a proper goal. We think so; if you disagree, there is probably a deep philosophical chasm between you and us, one that we may not be able to bridge. We will therefore settle for listing our reasons, without any attempt to challenge yours.

First, in a technological era, computer security is fundamental to individual privacy. A great deal of very personal information is stored on computers. If these computers are not safe from prying eyes, neither is the data they hold. Worse yet, some of the most sensitive data—credit histories, bank balances, and the like—lives on machines attached to very large networks. We hope that our work will in some measure contribute to the protection of these machines.

Second, computer security is a matter of good manners. If people want to be left alone, they should be, whether or not you think their attitude makes sense. Our employer demonstrably wants its computer systems to be left in peace. That alone should suffice, absent an exceedingly compelling reason for feeling otherwise.

Third, more and more of modern society depends on computers, and on the integrity of the programs and data they contain. These range from the obvious (the financial industry comes to mind) to the ubiquitous (the entire telephone system is controlled by a vast network of computers) to the life-critical (computerized medical devices and medical information systems). The problems caused by bugs in such systems are legion; the mind boggles at the harm that could be caused—intentionally or not!—by unauthorized changes to any such systems. Computer security is as important in the information age as were walled cities a millennium ago.

A computer intrusion has already been blamed for loss of life. According to Scotland Yard, an attack on a weather computer stopped forecasts for the English Channel, which led to the loss

of a ship at sea [Markoff, 1993]. (Recent legal changes in the U.S. take cognizance of this, too: hacking that results in deaths can be punished by life imprisonment.)

That the hackers behave badly is no excuse for us doing the same. We can and must do better.

Consider the question of "counterintelligence," the activities we undertake to learn who has been pounding on our door. Clearly, it is possible to go too far in that direction. We do not, and will not, attempt to break into a malefactor's system in order to learn more about the attacks. (This has been done at least once by a government organization. They believed they had proper legal authorization.) Similarly, when we found that our machine was being used as a repository for pirated software, we resisted the temptation to replace those programs with virus-infected versions (but we did joke about it).

The ethical issues go even further. Some people have suggested that in the event of a successful attack in progress, we might be justified in penetrating the attacker's computers under the doctrine of self-defense. That is, it may be permissible to stage your own counterattack in order to stop an immediate and present danger to your own property. The legal status of such an action is quite murky, although analogous precedents do exist. Regardless, we have not carried out any such action, and we would be extremely reluctant to. If nothing else, we would prefer to adhere to a higher moral standard than might be strictly required by law.

It was suggested by a federal prosector that pursuit in this manner by a foreign country would constitute an act of war. This may be a little extreme—a private citizen can perform an act of terrorism, not war. However, acts of terrorism can elicit military responses.

Overall, we are satisfied with what we are doing. Within the bounds set by legal restrictions, we do not regard it as wrong to monitor our own machine. It is, after all, *ours*; we have the right to control how it is used, and by whom. (More precisely, it is a company-owned machine, but we have been given the right and the responsibility to ensure that it is used in accordance with company guidelines.) Most other sites on the Internet feel the same way. We are not impressed by the argument that idle machine cycles are being wasted. They are our cycles: we will use them as we wish. Most individuals' needs for computing power can be met at a remarkably modest cost. Finally, given the currently abysmal state of host security, we know of no other way to ensure that our firewall itself is not compromised.

Equally important, the reaction from system administrators whom we have contacted has generally been quite positive. In most cases, we have been told that either the probe was innocent, in which case nothing is done, or that the attacker was in fact a known troublemaker. In that case, the very concept of entrapment does not apply, as by definition, entrapment is an inducement to commit a violation that the victim would not otherwise have been inclined to commit. In a few cases, a system administrator has learned, through our messages, that his or her system was itself compromised. Our peers—the electronic community of which we are a part—do not feel that we have abused their trust.

Of course, cyberwarfare is now an active part of information warfare. These rules are a bit genteel in some circumstances.

1.7 WARNING

In the past, some people have interpreted our descriptions of our security mechanisms as an invitation to poke at us, just to see if we would notice. We are sure, of course, that their hearts were pure. Conceivably, some of you might entertain similar misconceptions. We therefore humbly beseech you, our *gentle readers*:

PLEASE DON'T.

We have quite enough other things to do; it is a waste of your time and ours, and we don't really need the extra amusement. Besides, our companies' corporate security departments seldom exhibit a sense of humor.