

# Paper number: 4145

## **Firewalls 101**

Christopher Michael

RMS business systems  
135 Arlington Heights Rd  
Buffalo Grove, IL 60089  
847 215-1661 x 381  
cm@rmsbus.com

# Introduction

This paper is intended to provide an introduction to firewalls especially as they relate to protecting a corporate network from the Internet.

## Who needs security?

There is a very simple test that you can administer to determine if your corporation needs a firewall to protect it from the dangers of the Internet. Answer this question: “Does your company have a dedicated Internet connection?” If the answer is “yes”, then you need a firewall.

Now, you may be thinking that your company doesn’t have anything that would interest a hacker. The truth is that while your company may not attract the attention of commercial hackers, there are plenty of kids who are just looking for machines to break into. Just having someone break into your system is trouble enough. Even if they don’t intend to do damage, once someone you don’t trust has gotten root privileges, you can no longer trust anything on that machine. Your only recourse is to install the operating system and all program files from known clean sources (not from the backup) and validate your data.

There is a potentially more serious problem with leaving machines exposed to the Internet. Hackers often like to cover their tracks by going through several machines before they attack their real target. They will attempt to delete the log files on the intermediate machines in order to make it more difficult to trace them backwards. Your machine, even if it’s not attractive for itself, might be a platform for a hacker to launch an attack against another site. Some legal experts believe that you could be liable for damages if you could not demonstrate that you had taken reasonable steps to prevent a break-in and it’s not necessarily clear how far you have to go to protect yourself.

The legal precedent for this is found in the T. J. Hooper case, which dates back to 1932. Briefly the facts are these:

- A coal barge was caught in a storm and sank.
- The tug boat was not equipped with a radio that would have permitted the crew to receive a storm warning.
- At the time of the accident, it was not industry practice to equip all tug boats with radios. Some had them, some did not.
- The courts ruled that even though the industry hadn’t moved to universally equip tugboats with radios, that the technology was readily available and should have been provided.
- The tugboat line was held to be responsible for the loss.

So if a 12 year old kid uses your computer system to break into a hospital and compromise patient records, you could be liable. When the lawyers start looking around for someone to sue, who do you think it will be: a 12 year old or a corporation? Hint: who has more money?

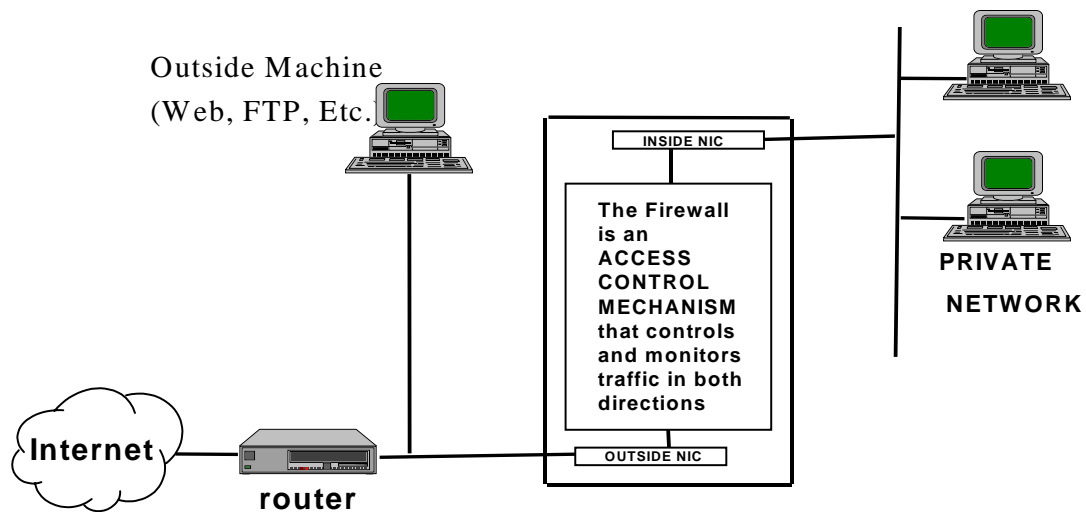
## How much security is enough?

Good question! <pause> Next question?

Setting security policies is a matter of tradeoffs. You get additional security by creating some degree of inconvenience for your users and spending time and money on a firewall. This really becomes an issue when someone wants to allow a service or protocol through the firewall that you believe to be unsafe.

## What is a firewall, anyway?

A firewall is a system of hardware and software designed to control access between trusted and untrusted networks—often between the Internet and a company’s private networks. It’s a means of implementing parts of a comprehensive computer & network security policy.



A firewall:

- Controls access between networks.
- Provides logging and reports.
- Has strong authentication capabilities, such as one time passwords, or “smart cards”.
- Provides an easy mechanism to enable or disable services between networks.
- Has “smoke alarms” to detect unauthorized connection attempts.

## Steps toward Security

Security, contrary to what you might have heard, is easy to achieve. You simply need to remove all users, connectivity, applications, data, operating system, and writeable memory from your computers; encase them in cement and store in a bank vault under constant guard. Now that’s secure! Security and being able to do useful work, is somewhat more difficult to achieve.

There is a logical process that you can go through in your quest for security. The steps, which we will discuss in detail in a moment, are:

1. As part of a comprehensive computer and networking policy, develop and obtain approval for an Internet security policy.
2. Choose your security stance from one of the four “P’s”: Promiscuous, Permissive, Prudent or Paranoid. Your choice will be partly determined by your previously developed security policy.
3. Design your security system and develop your user education program.
4. Implement
5. Test
6. Make any necessary changes in design or user education programs
7. Loop

## Policies

### The comprehensive computing and networking policy

A discussion of a complete computing and network policy is well beyond the scope of this paper, but some of the things that it might include are:

- Password/authentication standards
- Emergency response plans
- Policy on personal use of corporate computers
- E-mail policy
- Physical security for servers & network devices

## Internet security policy

It's very important to have a policy in place and approved before moving on to the next step. Without a policy, you have no basis for your security design. Furthermore, your users will quickly figure out that you're creating policy "on the fly" and you'll be in a weakened position when you have to say "No" to someone's latest Internet toy.

In crafting your policy it's important to keep a sense of proportion. While an Internet connection certainly can pose a threat, so can other things like: random modem connections into your computers or networks, laptops with sensitive data, improperly disposed of printouts and users who post their passwords on their computers. If you make access to the Internet too difficult, users will try to find a way around your restrictions which may, in the end, make your systems less secure than if you'd been a little more accommodating.

The Internet security policy needs to define what protocols you will allow to pass between your networks and the Internet. Here's a table showing some possible choices for four common protocols. Notice we have three networks: Inside, Outside and Public. The Inside network is our corporate network. The Outside is the Internet and the Public network is between the Inside and Outside networks. Having a separate Public network allows us to create separate access policies for public servers like web or ftp servers.

According to the policies defined below, a user on the corporate network would be able to telnet to a host on the Internet, browse Internet web sites, get and send mail to the Internet and download, but not upload files using FTP. From the Internet, selected employees would be permitted to telnet to our corporate hosts and would be able to upload files to them. Anyone would have access to our web and ftp servers.

SERVICE	INSIDE TO OUTSIDE	OUTSIDE TO INSIDE	OUTSIDE TO PUBLIC	PUBLIC TO OUTSIDE
<b>Telnet</b>	Yes	VIP: Yes Others: NO	NO	NO
<b>FTP</b>	Get: Yes Send: NO	Send: Yes Get: NO	Yes	NO
<b>SMTP (Mail)</b>	Yes	Yes	NO	NO
<b>HTTP (Web)</b>	Yes	NO	Yes	NO

Your Internet policies need to include an emergency response plan for dealing with break-ins. While you hope that you'll never have to use it, having a plan in place can prevent a problem from becoming a disaster. If it isn't covered elsewhere, you should also have a policy about personal web browsing and viewing of "adult" sites at work.<sup>1</sup>

## Security stance

Your security stance of promiscuous, permissive, prudent or paranoid will be determined partly by your security policy and partly by your disposition. By *stance* we mean how you'll approach the problem of implementing your policies. A little explanation should make it all clear.

---

<sup>1</sup> Of course if you work at an adult web site, you might have a different perspective on this issue.

**Promiscuous stance**

The promiscuous stance calls for few or no controls on Internet access. It's basically "hookup and let's party!"

**Permissive stance**

In the permissive stance the security administrator starts from a position of allowing full Internet access and then attempts to block services or protocols that are identified as dangerous. The problem with this approach is that it requires you to know in advance everything that is dangerous. It also locks you in mortal combat with your users who will be continually inventing new and dangerous things to do—only to have you take them away when (or if) you find out.

**Prudent stance**

Taking a prudent position, the security administrator begins by allowing *no* access to the Internet. As protocols are determined to be both useful and safe they are enabled. Contrasting this with the previous approach we see that the administrator isn't required to identify possible threats—he merely has to understand the services he enables. This is a much smaller universe. Because he starts from a position of allowing no access, he should never have to take something away from users that they've already had. This is the recommended position.

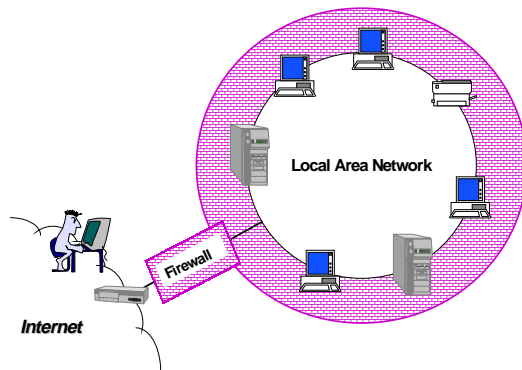
**Paranoid stance**

The paranoid position is to be so concerned about security that you don't allow your users access to anything—or you make it so difficult that you may as well have not given them access. If your need for security is that great, then you shouldn't have an Internet connection in the first place. Users will often find a way around restrictions that they think are unreasonable. This may leave you with undetected security breaches that might only be found after it's too late.

## Design

With a security policy in place and the correct stance selected, you are ready to begin the design phase. Once again, you have choices. At the most basic design level you have a choice between a perimeter defense or a defense in depth.

## Perimeter defense

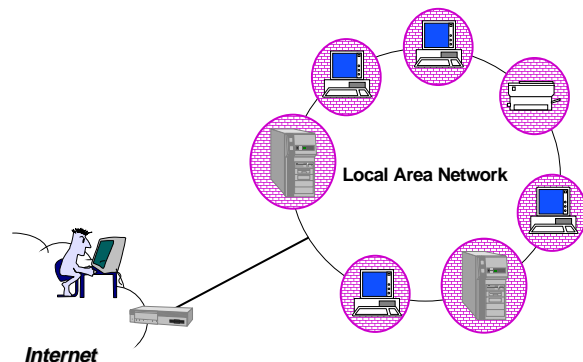


In a perimeter defense, you use a firewall to control access to your internal networks where they attach to the Internet. Everything on the inside of the perimeter is assumed to be safe and trustworthy—an assumption that is probably unwarranted.

The advantage to this approach is that you only have a single machine, the firewall, to worry about. It can be specially hardened against attack and special care can be taken to ensure that patches are kept current.

## Defense in depth

A defense in depth is the exact opposite approach. Instead of defending your network where it connects to the Internet, you attempt to defend each individual machine.



There are a number of problems with this strategy:

- All your defensive work has to be repeated on each machine.
- General-purpose machines are more difficult to secure than a firewall because they have inconveniences such as applications and users.
- It's unlikely that all host machines would be under the administrative control of the security administrator. This means that he would have to rely on other people—who may or may not have an interest in having those policies enforced—to implement security procedures.

## The belt & suspenders approach

Of course you can—and probably should—combine a perimeter defense and a defense in depth for even better security. One of the weaknesses of the pure perimeter defense is that it assumes that people on your own network can be trusted.

## User education

As part of your planning process it is essential that you develop a plan to educate your users about the need for security and their responsibilities. If they understand the dangers involved they will be more likely accept the inconveniences caused by security measures and less likely to try for an end run.

# Firewall technologies

## Operating system

Firewalls have traditionally been Unix-based, but with the growth of NT, we are seeing NT firewalls appear. My advice is to avoid NT. With Unix it's fairly easy to know what services are running—you either run them as daemons or they're started by inetd. With NT it's not nearly so clear. With Unix, if you want to prevent routing, you can take that part out of the kernel and recompile. With NT, it's not so simple.

## Technologies

There are three major technologies used to build firewalls: Static filtering, dynamic filtering, and application proxies. Each of them has characteristic strengths and weaknesses that determine the character of the firewalls that employ them. It's important to understand the differences among them so that you can make an informed choice between firewall vendors' offerings. But first...

## About packets

In order to understand how firewalls work and the differences between different firewall technologies, you need to understand something of how information flows in a TCP/IP network. I'm going to simplify a lot here, so you TCP gurus just chill out.

The packet is a basic hunk of information that travels on an IP network. Like the letter you drop in the mail, it has the message part, or data, and it has the envelope part, or header.

The header information is used by the network to direct the data from its source to its destination. Some of the information that's contained in the header is protocol, source & destination address, source & destination ports, syn & ack flags and a sequence number. If you don't know what all that stuff is, that's okay. The important thing for now is just that that information is there and can be used by firewalls to make decisions about what to let through and what to block.

The destination port numbers, for example, often indicate what service a connection is being made to: 21 is ftp, 23 is telnet, 25 is SMTP, 80 is a world wide web server and 110 is popmail. So if a firewall sees a connection coming in headed for port 23 and you've disallowed telnet connections, it's going to block it. The syn flag indicates that the pack is part of an already established connection, rather than the beginning of a new one. Sequence numbers are used to ensure that the data is delivered in the correct order and that no packets have been lost.

The data part is, well, data. It's the information that you're sending from one place to another. It could be part of the text of an e-mail message, part of a picture from a web page—anything that travels over the network.

## Static packet filtering

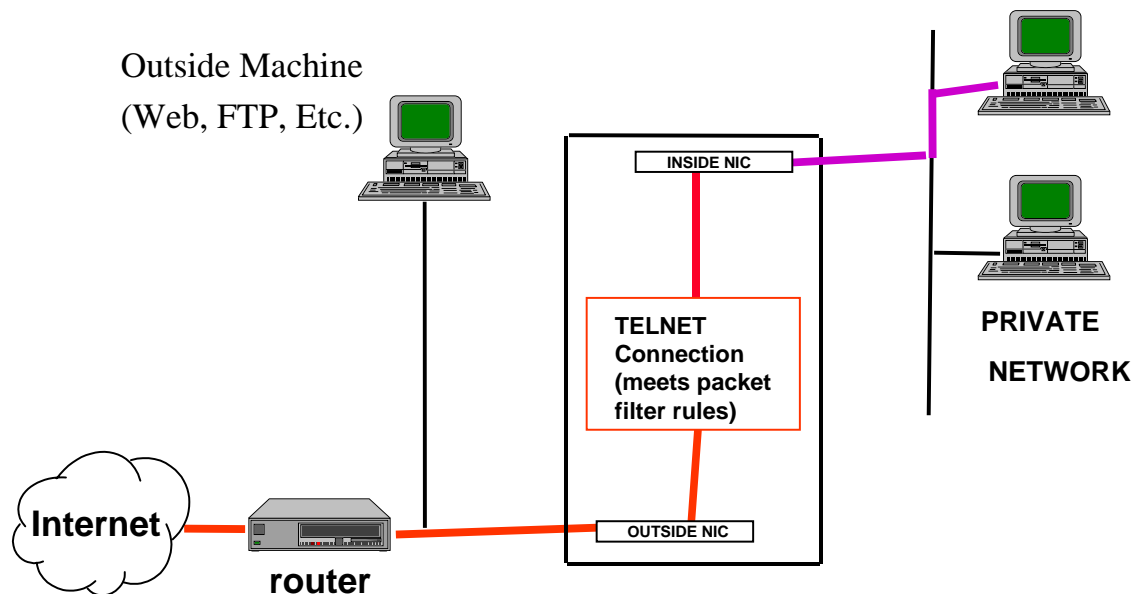
The packet filter is called *static* because they don't change according to what's going on. As static packet filter accepts or rejects a packet on the basis of information contained in the packet header. If the packet meets the rules, it is passed through the router transparently. If not, it is dropped.

If you only wanted to allow traffic to a web server, for example. You could close every port except port 80. To allow popmail, you'd open port 110 and so forth. You could even allow popmail to your mail server, but not to other machines. This doesn't sound to hard to manage, except that not all protocols are as simple as the examples I just gave. Telnet and ftp require connections on their well known port numbers and additional ports above 1024 that they negotiate as part of the connect procedure. This means, in practice, you have to leave all ports above 1024 open, since you don't have any way to automatically open the correct port. This leaves you open to whatever people can find to exploit in that range.

Static packet filtering does have its place—if you absolutely can't get anything better, it will give you some protection. You need to have a detailed knowledge of Internet protocols and of your router. Any attempt to go beyond the simplest kinds of filters quickly become very complex and error prone.

## Dynamic packet filtering

Dynamic packet filtering uses intelligent management of packet filters to eliminate some of the problems associated with static filters. Dynamic packet filters are aware of protocols such as telnet and ftp and dynamically open ports above 1024 as required and close them when the session terminates. They maintain internal tables of ongoing connections so that they are able to detect if an incoming packet is part of an established connection.



### Packet filtering firewall

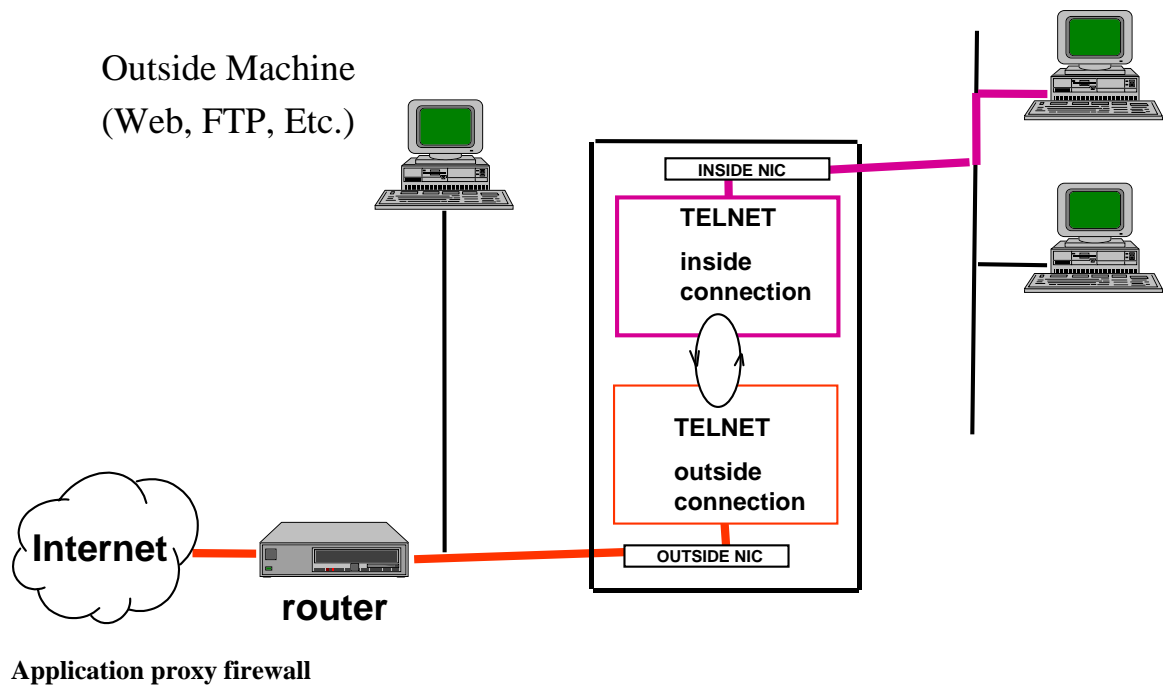
As with a static packet filter, if a packet meets the rules, a direct connection is established between the source and destination hosts just as if there were no firewall in the middle. This is problematic, because it means that bugs in the destination host software are accessible to the source host. Sendmail, which handles Internet mail on Unix machines, is often exploited in this way.

### Application proxy

With a firewall employing application proxies, each service that is to be allowed to pass through the firewall is handled by a small program running on the firewall. Unlike the filtering techniques which establish a direct host to host connection, a proxy based firewall passes only *data* between its interfaces. When a packet hits the firewall, it is directed to the proxy associated with the destination port. Because the proxy is written specifically for that application, it can base its accept/reject decision on the actual data in the packet, as well as header information and connection status. This allows you to implement features such as Java and active-X blocking. It also allows extensive logging.

The proxy based firewall is arguably the most secure type of firewall. Because only data is passed between networks, it is impossible for hackers to exploit bugs in host application software that depend upon having a direct connection. The downside of this is that if you want to support a new service, you must write a new proxy. But even that isn't all bad, since it requires you to understand the protocol and thus you are able to access the risk of proxying it through your firewall. Commercial proxy firewalls come with proxies for all the common TCP/IP based services.





### Combination strategies

Some firewalls use a combination of filtering and proxies in an attempt to optimize security and performance.

## What does this all mean?

### Static packet filters

- Better than nothing
- Difficult to support for complex configurations
- Cannot provide as much protection as a firewall
- Often does not have sufficient logging

### Dynamic packet filters

- Provides much more protection than static filters
- Higher performance than proxies
- Easy to add new protocols (doesn't mean they're safe, though)
- Less secure than proxies

### Application proxies

- Most secure type of firewall
- Somewhat less performance than filtering technologies, but okay up to Ethernet speeds, at least
- More difficult to add new services, since a proxy must be written
- Able to add features such as Java blocking for web browsing
- Able to do extensive logging

# Evaluating a firewall

## Criteria

Having arrived at a security design that cries out for a firewall, how do you know which one to choose? I would suggest that the first step is to be clear about what's important to you in a firewall. As with everything else, selecting a firewall involves trading off different criteria to achieve an optimal mix of features. Some of the things you might consider are security, ease of use, software/hardware platform, type of firewall technology and cost. For my money, as long as you meet minimum standards in terms of ease of use and cost, security is the overriding concern. Your mileage, of course, may vary.

## Reviews and certifications

I caution you against putting too much trust on either certification programs or reviews in the technical press. While the National Computer Security Association (NCSA) does put their stamp of approval on firewalls that pass a suite of tests, the testing is at such a basic level that it would be difficult to fail. Because they are an industry-supported group, they are not likely to have opinions about the validity of different firewall technologies.

Reviews can be even more misleading. Firewalls are complex products and the issues that surround them are not obvious to the casual observer. Reviews tend to focus on easily understood features—such as the administration interface—and leave unexplored the more serious issues.

## Build or buy

In a situation where you have more time than money and you are an experienced Unix system administrator with a working knowledge of the C language and TCP/IP networking you can put together your own firewall using tools freely available on the Internet and a free operating system such as free BSD or Linux. If you don't match the above, then you should probably buy.

## And the answer is...

- If you have an Internet connection you need protection. For most sites a firewall is the best way to protect your networks. If you can't afford a firewall, you can get some protection from filters in your router.
- The two main technologies employed by firewalls are dynamic packet filtering, a.k.a. "stateful inspection" and application proxies.
- Dynamic packet filtering is more flexible, but less secure when compared to application proxies. If you do choose a packet filtering firewall you will need to ensure that host applications, such as Sendmail are kept free from known exploitable bugs.
- Get the backing of management and educate your users.

## Pointers to more information

Recommended books and links to firewall and security information can be found at <http://www.rmsbus.com/firewall.htm>