# A FIREWALL CONFIGURATION STRATEGY FOR THE PROTECTION OF COMPUTER NETWORKED LABS IN A COLLEGE SETTING[*]

*Dennis Guster, Professor*
*and*
*Charles Hall, Research Assistant*
*MCS Program, Department of Statistics*
*St. Cloud State University*
*St. Cloud, MN 56301-4498*
*320/255-4961*
*guster@mcs.stcloudstate.edu*

As more and more networked college instructional computer laboratories are attached to the Internet, the need for protection from hackers becomes evident. The attacks perpetrated can take many forms from information compromise to introducing viruses. However, one of the most dangerous is a passive takeover of a host so that it may relay attacks on other sites which allows the hacker to cover his or her tracks.

Configuring any given host to be secure from this type of attack is a most challenging endeavor. This problem is especially magnified if the domain in question contains hosts configured by students. This is especially true for students just beginning a course sequence in networking or operating systems. The operating systems when installed with the default settings are often ripe with security holes. These problems range from unblocked I-O ports (OSI/4) such as UUCP to permitting potentially dangerous types of ICMP traffic such as routing requests. It cannot be expected that beginning students have the knowledge to combat these security problems, although as they progress through the course sequence, hopefully they will. Furthermore, this first course that involves OS configuration provides prerequisite knowledge, and therefore unprotected machines may need to exist in an educational network domain.

Because security measures cannot be invoked on every machine in the domain, the security measures need to be applied through a firewall as traffic enters and leaves the domain.

Therefore, the paper proposed herein will describe how that firewall should be configured and will focus on what types of incoming/outgoing UDP, TCP and ICMP traffic should be denied or accepted.

## THE PROBLEM

Hacking on the Internet has reached epidemic proportions. A recent CBS report stated that even the CIA couldn't keep pace with the logic and technology employed by today's hacking community.

These attacks occur at a frequency that is hard to imagine. A recent article reported an attack rate of one attempt every 1.5 seconds. This statistic further supports the need for a proactive security plan [1]. This volume is certainly disturbing, but how sophisticated are the attacks? The same article reported that the vast majority of the attacks were probes using software readily available and publicized on the Internet. This would indicate that a large number of hackers might in fact just be kids playing around with something they deem as "cool". Furthermore a good many of the attacks are easily traced back to the source. This brazenness would tend to indicate there is little fear on the hacker's part of getting caught or the consequences thereafter. While these amateur attacks make up the majority of threats, the truly professional hackers still present the greatest risk. Therefore, a comprehensive security plan must address both types of attacks.

An analysis of preferred targets reveals educational and governmental institutions to be in the forefront. The hacking problem has gotten so out of hand that the federal government is even considering splitting their activity from the Internet via a separate VPN (virtual private network) [2]. Due to their mission, educational networks have a more difficult philosophical decision about isolating traffic. Because educational institutions' goals focus on learning, one can interpret their primary usage to be learning by doing, experimentation and research. Therefore, these activities need to take place on the real Internet not a VPN facsimile, if students are to gain experience in the computing environment they will use after graduation.

What are the ramifications for those institutions that choose to ignore the security problem? In addition to disrupted service and compromised data, the question of liability arises. It appears future liability will become a legal issue, and victims will seek recompense from any party involved in a break-in [3]. Besides protecting the outside world from any potential student hackers, educational institutions need to be aware that if a hacker uses a computer on their site as a relay to cover their tracks that institution is potentially liable. Therefore, it is critical for educational institutions to develop a comprehensive security plan because they are prime hacker relay sites. Specifically, in legal terms, they must exercise reasonable care in the prevention of hacking activity.

Besides the dreaded relay scenario what other breaches are of concern? For example, stolen hardware may be of concern beyond its hardware value. In other word, it may contain scripts that would allow a hacker to log in as a legitimate user and create havoc in your internal network. Furthermore, this same account could access resources in the outside world and appear as one of your users [4]. In fact, the legitimate user of that account could even become a suspect in a federal investigation [5]. Therefore, it is imperative to protect your users from outside threats. In educational environments it is equally important to protect the outside world from your students. Many students use the Internet where they intentionally or unintentionally receive exposure to the hacking culture. Furthermore, many students take legitimate classes that contain material useful in the hacking process, and they may decide to use the Internet to experiment with their new knowledge. In either case a proactive protection policy is paramount.

If one takes this security problem seriously, it is obvious that some type of comprehensive plan needs to be implemented. If the expertise to do so is not available internally, is outsourcing that responsibility a viable option? It would appear that outsourcing is an option, but for most installations not a very effective one [6]. In the article cited above, outsourcing proved to be more expensive, less flexible, done by analysts with questionable expertise and in some ways less secure when compared to an internal solution. The bottom line is your staff is the best guardian of your system. Their loyalty and familiarity with your applications are important foundations in devising a sound security plan. Security has become part of the overhead of conducting business on networked computers and should be treated as such. This may mean a commitment for staff retraining, but in the long run may offer the best solution. Keeping pace with security requirements is certainly a difficult endeavor for educational institutions whose funding is often tied to fixed budgets. However the ramifications of not providing reasonable care in the security arena could become more costly later on in liability suit. Educational institutions need to react now and start developing in-house expertise and programs that teach students security concepts.

## A CASE STUDY

### Environment

One of the characteristics that differentiate an educational from a commercial Internet environment is the need to attach devices to the Internet that have not been optimized for security. Because students are learning, usually in a predefined sequence, they cannot be expected to configure devices that are attack proof. In fact their devices tend to be full of security holes and often become prime targets for hackers. Typically a default installation is done. In other words, they use the quickest way to get the device up and running and connected to the Internet. This environment is a hacker's dream because they love to take over unsecured Unix based host for relays.

Of course beginning students cannot be expected to configure relatively secure machines immediately, but hopefully later on in their careers they will be able to. Therefore, what can be done to enhance security during this vulnerable developmental time? Isolation is a good first step. It is a good idea to place all vulnerable developmental activities on a separate network, but that network can still be connected to the Internet and may have its own domain name. Furthermore this developmental network should have its own firewall in addition to the central campus firewall, which should be the first incoming security filtering point. This scenario offers a higher degree of flexibility and another level of protection not offered by a single centralized firewall system. Also, because control of the inside firewall is often retained at the department level, emergency fixes can be implemented more quickly.

### Firewall Philosophy

The development philosophy for the domain specific secondary firewall is centered on two goals. First, to provide added protection due to the known presence of unsecured systems. Second, to serve as an instructional tool to help students learn about implementing a security policy. To meet these goals it was decided to implement the firewall on a standard Linux release. Although this solution lacks integrity to some extent, it provides a standard easily readable environment from which students can study a sample security policy.

### Foundation Knowledge

The development of the secondary firewall made it clear to the developers that if basic firewall principles were to be taught to students they would need a solid foundation in protocols (particularly Ipv4), internet addressing, processes and logical I-O ports. Furthermore to understand the interrelationship of all of these components an understanding of the OSI (open system interconnect) model would be needed. These observations led to the modification and strengthening of existing data communications classes so that the students would have the necessary background to learn firewall techniques.

### Configuration for the Case Study

The firewall configuration file used in the case study contains the logic used in the initial configuration. Although it has been modified several times since, it provides an excellent starting point that can be explored herein without offering too much complexity or compromising the current live site.

The logic operates under the assumption that certain packets will be allowed through the firewall and others will be rejected. The trick is to configure the firewall to accurately identify each category. This configuration process typically involves deciding what type of incoming traffic to accept/deny and what type of outgoing traffic to accept/deny. The characteristics of this traffic can be defined by several criteria such as source/destination network, network node address, logical I-O port and/or protocol used. For more complex configurations factors such as syn (connection request), flags and options can be controlled for in the security plan.

The logic behind the case study sample configuration follows:

This list of rules is set up into multiple sections to create easier reading.

Each chain has a section, also there are two other sections: Global and No Match.

The Global section is at the beginning and contains all of the rules that need to be checked and processed before the packet can go to the correct chain.

The No Match chain is at the end and is used to either deny or allow all packets that made it through the other chains.

In the Global section, this is how things are set up:

The first thing done is enabling ip forwarding on Linux.

What this will allow us to do is use Network Address Translation (NAT).

Second, flushing all of the chains so there are no rules in them takes place.

This is included in case the ipchains is run again, and to prevent having multiple rules that have been deleted from being used.

Next up is deleting the user-defined chains.

The only reason these rules are here is if the chain's rules have been removed, and the chain is no longer needed. this will get rid of it.

Note: input, output, and forward chains are made by the system and cannot be deleted.

Following that, is creating the user-defined chains that were just deleted.

That is the end of the Global section. In the following parts we will see each chain and the logic behind it.

The Input chain and the Output chain send packets on to the chains depending on if the packet is outgoing or incoming.

Rules inside the forward chain affect any packets that are being passed through the firewall.

In this example rule set Masquerading (MASQ) is turned on; what this does is make all connections coming from the inside look like the request was from the firewall.

MASQ and NAT are almost the same thing. The difference is that NAT can be used in both ways, while MASQ is only one way.

The ICMPCHIN user-defined chain controls what ICMP packet types are allowed in, the ICMPCHOU does the reverse.

UDPCHIN checks all UDP type traffic coming in, where UDPCHOU checks all of it going out.

TCPCHIN checks the incoming traffic and the TCPCHOU checks the outgoing traffic.

The UPD and TCP chains can be specified to block all traffic to specific ports and/or IP addresses.

There are currently no rules in the NO MATCH chain; after the firewall is fully operational, we will add rules blocking all packet types that do not match one of the rules in the other chains.

The last line is there to make sure IP forwarding is available.

When implemented the sample configuration file would look as follows:

```
#THE FIREWALL LISTS

#!/bin/sh
```

The first group of commands that contain the -F flush the old rules from the nine defined chains. Input, output and forward are system defined chains and the others user defined chains. The user-defined chains are basically input or output relating to a specific protocol, TCP is connection oriented, UDP is a connectionless datagram, and ICMP is a management protocol. The second group of commands that contains the -X deletes user defined chains which combined with the deletions from the previous group ensures that the slate is clean and new definitions can be made without fear of contamination from prior definitions.

The command group that contains -N creates the user-defined chains and the next three groups define the input, output and forward chains. The chain characteristics are defined next. The ICMP definitions occur in the next two sections: first the ICMPin chain followed by the ICMPout chain. Each section contains code that describes what type of incoming or outgoing ICMP traffic is permitted. For example, echo requests (used in ping) are only permitted outgoing on interface eth1, while echo replies are only permitted as incoming on interface eth0. Also note that any ICMP traffic going in either direction unless explicitly accepted is denied.

The same basic logic is applied to UDP traffic in the next two sections except that traffic not explicitly denied is accepted. For example, ports 6000-6003 (X windows traffic) are denied in both directions, whereas source ports 81-109 are denied as outgoing. Also note that outgoing UDP traffic to port 23 (telnet) is denied to destination device 199.17.40.165.

The next two sections define incoming and outgoing TCP traffic. Again similar logic to the UDP sections is invoked wherein packets not specifically denied are accepted. Some interesting examples include entries with a -y (syn or connection flag) which are designed to block any incoming FTP connection request or any incoming request to nonpriviledged ports 1023 and above. Also of interest is the first statement in the TCP in section that contains an "!". This statement in effect blocks all incoming SMTP (mail) traffic except to the specified device 199.17.40.165. The last section, no match chain, has not been totally configured. Rather entries meeting this criterion are simply logged then analyzed and the appropriate action invoked later.

## DISCUSSION AND CONCLUSIONS

The firewall example offered herein was designed to be a template from which a more sophisticated and useable product could evolve. Because it is a first generation attempt there are security flaws and logic problems in its design. However as a teaching/learning tool to understand basic firewall configurations it is useful. It illustrates how packets can be accepted or rejected based upon their address, protocol, port or network. The basic logic of evaluating packets based on one or more chain definitions is an important process to understand.

It is hoped that an interested educational institution could use the basic premises herein and modify/adapt the basic logic set to their own specific needs. Implementing a successful version took many tries, and there was a constant battle to cover new threats while making sure legitimate traffic was unimpeded. It is apparent that maintaining adequate security is an ongoing process. It should also be noted that although the Linux-based firewall offers a high degree of software flexibility, it may be more prone to attacks because Linux is so popular with hackers. For large networks performance may be a concern. The implemented system in the case study had trouble exceeding a throughput of 6.5 Mbs when run on a 333Mhz Pentium system.

Although not the original intention of this project, perhaps its biggest benefit has been to provide students with experience configuring firewalls. After implementation of the original code proceeded through several versions, it was felt that the original code could be safely shared with students and it was used as a template from which they could build their analysis. The code was well received by students because it was well organized and contained examples from a network domain they were familiar with. It is clear that this type of example coupled with a strong foundation in the OSI model, network addressing, and protocols is needed if educational institutions are to produce the security professionals so badly needed in computing today. In the meantime, providing "reasonable care" in protecting current assets is paramount from both a legal and institutional perspective.

## REFERENCES

[1] Tuesday, Vince. Who's That Knocking At My Door? Go Away!, Computerworld, 35(20), page 52, May 14, 2001.

[2] Verton, Dan. Feds Consider Splitting Net For Security, Computerworld, 35(22), page 1, 73, May 28, 2001.

[3] Vijayan, Jaikumar. IT Security Destined For The Courtroom, Computerworld, 35(21), page 1, 73, May 21, 2001.

[4] Thurman, Mathias. Manager Locks Down As The Feds Move In, Computerworld, 35(25), page 44, June 18, 2001.

[5] Thurman, Mathias. What To Do When The Feds Come Knocking, Computerworld, 35(23), page 62, June 4, 2001.

[6] Tuesday, Vince. Security Outsourcing: Don't Bet On It Yet, Computerworld, 35(24), page 56, June 11, 2001.