Global Information Assurance Certification Paper

Author: Cory Steers
Date: 02/08/2001
Level Two Firewalls, Perimeter, Protection, and VPNs
Assignment #1 Security Architecture

This is the security architecture of GIAC Enterprises, an Internet startup company. To meet the company's needs, the architecture will have to provide Internet access for customers to make purchases, suppliers to provide new product, partners to prepare the product, and email and web browsing for employees.

Illustration 1.01, on the next page, is a high level view of GIAC Enterprises' main site, its newly acquired company, or remote location, and a sample business partner. The main GIAC site and its remote location will be illustrated in more detail later.

Notice the private leased line between the main GIAC site and its remote location. This will prove to be necessary for many reasons. Often, site-to-site communications require guaranteed levels of service. Since no one entity has complete control over the Internet, there is no way to guarantee any level of service at all. Additionally, since GIAC's business requires an Internet connection for its customers to reach it, the leased line will allow site-to-site fail over for that Internet connection. If the main GIAC site becomes disconnected from the Internet, it is still possible for the customer to reach the GIAC web site; therefore, it is likely that they will never know there was a problem.

You may also notice an optional private line between GIAC's main site and the example business partner. Similarly, GIAC may need levels of service to a business partner that can't be guaranteed via the Internet. GIAC will need to decide how critical its needs are for communication with that partner to justify the extra expense of that private line.

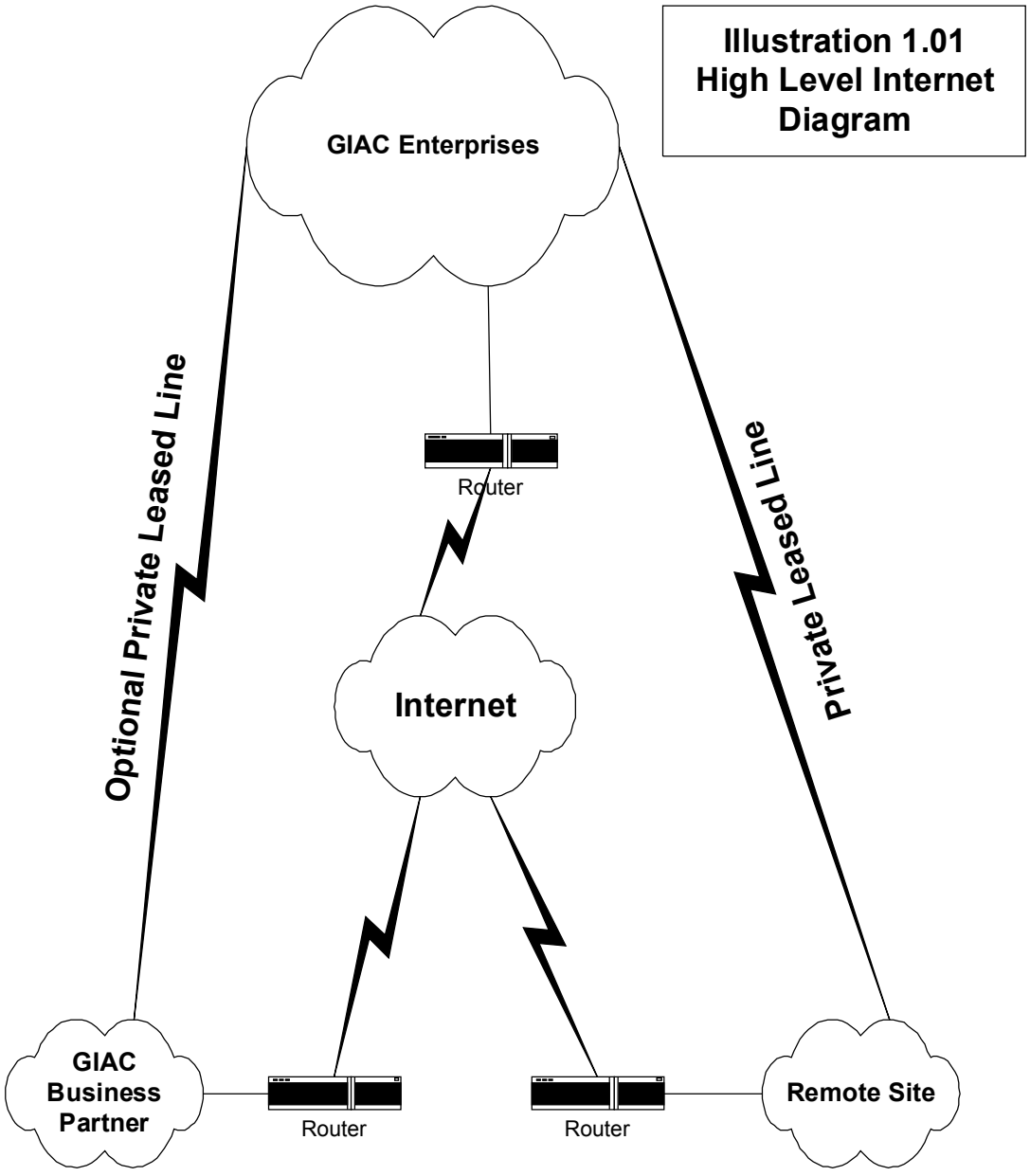


Illustration 1.02, on the next page, is a detailed diagram of GIAC's main corporate site. Some of the components drawn, such as Ethernet switches etc., could have been implied; however, I felt they were needed for illustration purposes. On the same note, there are a few spots where I purposefully left out some necessary equipment because of space considerations. I did not feel the need to discuss them; however, I will note where this missing equipment is. The remote GIAC site should look almost identical to illustration 1.02 except for a few minor things, which I will point out shortly.

There are several key points I want to make regarding the choices I made on my design. Those points are as follows:

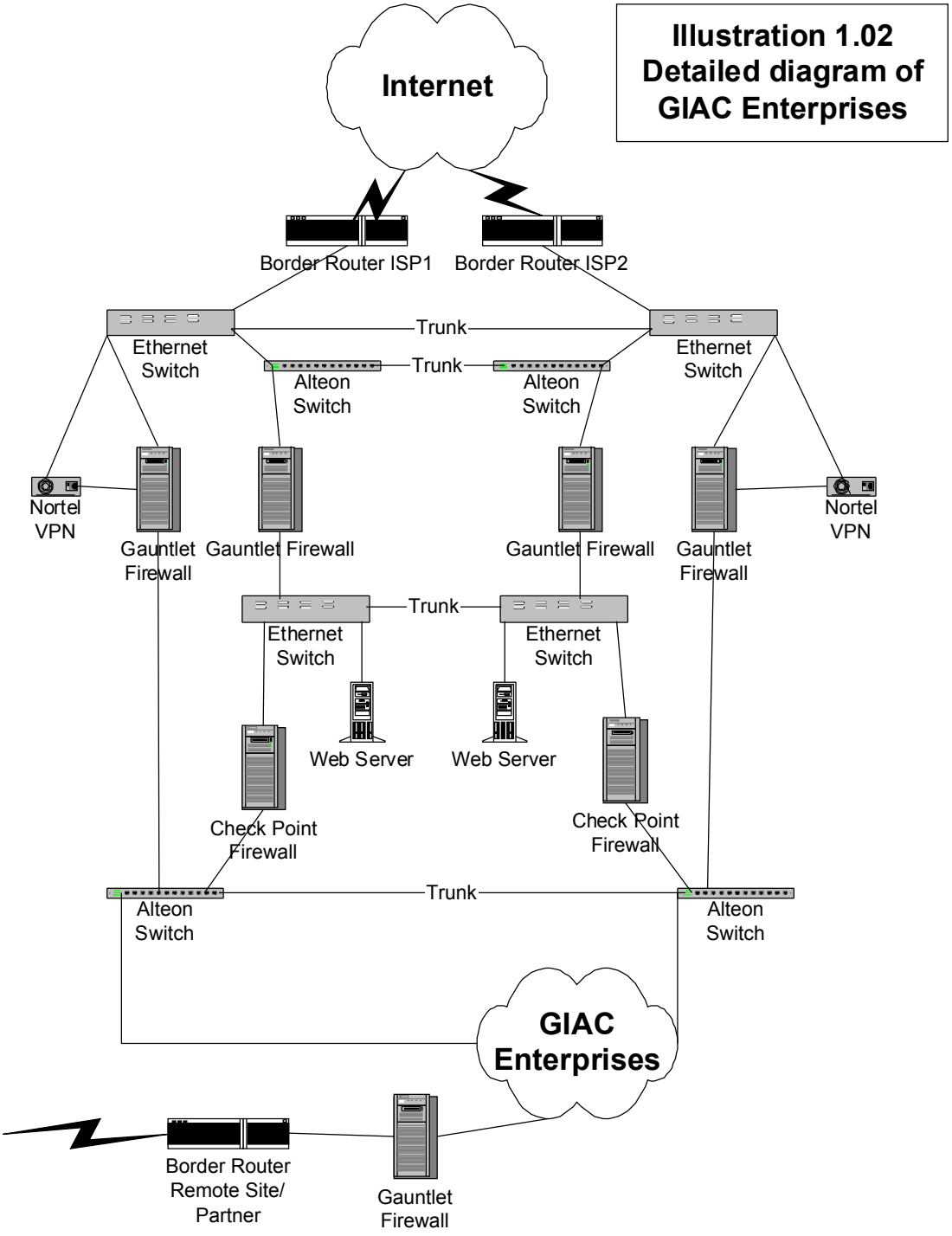
- The extremities I went to, to eliminate single points of failure.
- Vendor choices of products at each level.
- How I plan to leverage Alteon Web switches for high availability.
- My choice of VPN placement within the network.

GIAC's sole source of income and customer contact is through the Internet. Thus, it is critical for the customer to be able to reach the GIAC web site when they want to. This means 99.999% (affectionately termed five nines) availability or better. Because of this need, I have built redundancy into every component of GIAC's Internet infrastructure. Notice the two Internet connections from two different providers. The extra bandwidth may be needed anyway, but by having multiple providers, no single provider can leave you inaccessible. While you could support multiple serial connections on one router, that router could fail. Each router needs to be plugged into separate Ethernet switches in case a single switch fails. The switches are connected with a trunk. This allows multiple switches to share a single VLAN/network. You can begin to see the pattern; no single failure can stop the business.

For conservation of space, I did not show redundant components for the leased line connection to the remote site. However, it is important here too. A second router and serial connection can be substituted with an ISDN line and dialup router. It would depend on the capacity needs of the business. Also, not displayed in the illustration is intrusion detection (IDS) equipment. IDS is crucial for analyzing potential threats to the network.

The reason I mentioned that the remote sight could look identical to the GIAC corporate site is because you may want location redundancy. In the event of a natural disaster, such as a tornado or hurricane, the corporate site itself may be wiped out. Again, can GIAC afford to be unavailable to its customers?

For my border routers I chose Cisco equipment with the latest IOS and a compliment of access list (ACL) filter rules. These will be described in more detail in assignment 2. They are running open shortest path first (OSPF) on the Ethernet interfaces and border gateway protocol (BGP) on the serial interfaces.



Firewalls are a key piece of protecting a networks perimeter. In fact, they are arguably the single most important piece. That's why I feel that proxy based firewalls are the only choice for border protection. Stateful inspection firewall solutions running on the same hardware are faster; therefore, some people feel that they should be used to protect your service network or demilitarized zone (DMZ) so that they are not a bottleneck slowing your customer's access. I disagree. If your firewall isn't fast enough, regardless of its type, you either buy a bigger server/appliance or you buy an additional component and load balance the traffic between them. In short, eliminating a bottleneck is just a matter of money. Granted, most companies run on a fixed budget; however, when your network is compromised, you risk losing money from the loss of assets, private information and you run the risk of public embarrassment. How much is your name worth? Can you put a dollar amount on the loss of business that could result in public embarrassment? Besides, proxy firewalls are fast. Chances are, if you're firewall can't keep up, it's because your Internet traffic is high enough that you will be able to justify that additional capacity. For this reason, I've chosen Gauntlet proxy firewall, by NAI. Currently at version 5.5, with 6.0 currently in beta testing, Gauntlet is a proven firewall product.

Most firewall products offer a black box/device solution as well as running the software on a server on top of a mainstream OS. There are some performance and possible political reasons for choosing a black box solution. Usually the OS on the black box is similar to what you would run on a server, only it's been stripped down and customized for speed, simplicity, and security. Often, your company may have a policy stating that only servers running a certain OS are allowed. That OS may not be able to run a firewall product. This is another good reason to use the device solution.

I prefer implementing a firewall on servers, running some form of UNIX. There are many added benefits to running this configuration. In a large organization it can become cumbersome to keep track of all your firewalls. With a full fledged OS, you can run many jobs out of cron to build information files and even send them to an administration server. Often times, the black box OS is missing key pieces that the server OS has. Additionally, the server OS is probably more mature and less prone to exploits. The proprietary OS on the black box device needs to be treated as a new OS. It may be built from a mature server OS, but it has been changed. Sometimes exploits are created out of this process.

I chose to use separate firewall servers for protecting the DMZ and proxying outgoing traffic for internal network Internet access. I think a small straightforward rule set is more secure than a large complicated one. The more complicated your rules are, the more likely you are to make a configuration mistake. It is also more efficient.

The two outer Gauntlet firewalls in the illustration are handling traffic destined for the Internet from GIAC's network. They are proxying all allowed traffic and relaying Internet mail using Gauntlet's secure mail solution. The inner two Gauntlet firewalls are protecting the service network, or DMZ. They allow customers and business partners/suppliers to interact with GIAC via web interface.

The firewalls protecting the inside of the DMZ are Check Point. In situations where you have layered firewalls it is a good idea to make each layer a separate type of firewall. Personally, I like to switch between proxy firewall and stateful inspection firewall for each layer. The idea is if one firewall is compromised the attacker needs to compromise a different type of firewall to get through the next layer. Another advantage to having Check Point on the back end of the DMZ is that it logs all types of traffic (TCP, UDP, ICMP, etc.). Sometimes less secure traffic is needed between the DMZ and the trusted network. Gauntlet (although changed for 6.0) can't proxy, or even log, UDP traffic in its current production release.

The Alteon switches are being used for load balancing. The two switches at the top of the diagram are for load balancing traffic coming into the DMZ. With matching switches at the remote site, site-to-site load balancing/fail over is possible as well. The two Alteon switches at the bottom of the diagram are for load balancing outbound Internet traffic and traffic destined to the DMZ from the internal network. There would need to be additional Alteon switches inside the DMZ for a complete high availability solution. I've left them out of the diagram to conserve illustration space as I didn't feel the need to document them in depth.

I chose the Nortel Contivity appliance for GIAC's VPN needs. The Contivity is a sound product with good features. You'll notice that they are not load balanced with Alteon switches. They have their own high availability features. I protect the VPN device with a firewall from the GIAC network, but I did not protect the Internet interface. This design falls into the category "mix and match" from Chris Brenton's Network Design and Performance presentation, at SANS New Orleans 01/01. I feel that the outside of the Nortel device is very secure. If you want to talk to that interface, you better be talking Internet Key Exchange (IKE), Authentication Header (AH) protocol, or Encapsulation Payload protocol (ESP); otherwise, your traffic will get dropped.

On the other hand, the trusted side of the VPN may be considerably more open. After all, it may be listening on a management port or set of ports. Additionally, it is the gateway for any devices that are tunneled through it. Never forget that the majority of attacks come from the inside.

I would place IDS listening devices between the Ethernet port and switch of each router and between the "inside" Network Interface card (NIC) and switch of each firewall. That would be the DMZ side of the DMZ Gauntlet firewalls and the trusted side of the DMZ Check Point firewalls.

In conclusion, this document has proposed an intranet infrastructure for GIAC Enterprises, a growing Internet startup company. The architecture specifies filtering routers, firewalls, and VPNs. Customer and partner access to GIAC, and GIAC employee access to the Internet has also been discussed.

Author: Cory Steers
Date: 02/16/2001
Level Two Firewalls, Perimeter, Protection, and VPNs
Assignment #2 Security Policy

The purpose of this assignment is to provide a detailed security policy based on the architecture designed in assignment #1. For this assignment, security policy is defined as specific ACLs for routers, rulesets for firewalls, VPN configurations, or otherwise appropriate for the device being discussed. For the Check Point component, a brief tutorial will be given on how to apply the rules.

For illustration purposes of this assignment I will assume that GIAC Enterprises uses the 10.x.x.x address space for all internal networks, including remote sites. I will also assume 195.64.211.67/32 was given to GIAC for one of its two routers' serial interfaces and 195.62.178.0/24 was given to GIAC for its presence on the Internet at its primary corporate site. 18.24.242.0/24 was given to GIAC for its remote site Internet connection and 138.132.9.0/24 is the Internet address space for the sample business partner.

First, I will discuss the two border routers. As I mentioned in assignment #1, the routers are Cisco brand. The main security functions of the router are to protect the network from spoofing, control discovery type traffic such as ICMP, and to block source routing. It can also limit tcp and udp port 53 traffic to DNS servers and limit tcp port 25 traffic to only the mail servers; however, you do not want to have too many of these types of ACL rules on the router. The more rules you put on the router, the more memory the router needs and the longer it takes for the router to process packets. Again, for performance reasons you generally do not want the routers to log traffic. A good rule of thumb is to stop and log suspicious traffic at the firewall. The reason to configure the router to block the above mentioned traffic is because it is very easy for the router to block spoofing, source routing, and ICMP traffic. These types of traffic are more annoying than threatening and tend to needlessly fill up log files.

Anti-spoofing ACLs:

```
Interface Serial 0
        ip address 195.64.211.67 255.255.255.240
        ip access-group 10 in

access-list 10 deny 192.168.0.0 0.0.255.255
access-list 10 deny 172.16.0.0 0.15.255.255
access-list 10 deny 10.0.0.0 0.255.255.255
access-list 10 deny 195.62.178.0 0.0.0.255
access-list 10 permit any
```

The above rules are called ingress, or anti-spoofing rules. I'm using a standard access list versus an extended one. Extended lists provide for more configuration flexibility, but also take more processing power from the router. The first three rules block the private address spaces 192.168.0.0/8, 172.16.0.0/12, and 10.0.0.0/4. All Internet routers should do this, as these networks have been reserved for private use and are not to be routed across the Internet. The fourth rule denies any packet with a source address of the network behind the router. Since 195.62.178.0/24 is the network on the primary Ethernet interface, no packets with a source address in this range should be entering on the s0 interface.

Egress ACLs:

```
Interface Ethernet 0
        ip address 195.62.178.1 255.255.255.0
        ip access-group 20 in

access-list 20 permit 195.62.178.0 0.0.0.255
```

The above rule is for egress filtering. Remember, once an access list is applied to an interface, anything not explicitly permitted is implicitly denied. So, I permit anything with a source IP address of GIAC's network, and everything else is implicitly denied. This is important to help keep GIAC's network from participating in a distributed denial of service (DDoS) attack. If more ISP's, and the like, would implement egress filtering, DDoS attacks would happen much less.

It is also very important to secure the router. The enable password should be as secure as a password can be. It should be MD5 encrypted in the configuration. There are several things that can be done to keep your router secure and it is important that you do so.

Next I will discuss the rules on the firewalls. The diagram depicts two different sets of firewalls. There are Gauntlet firewalls proxying Internet access for GIAC's internal network, and separate Gauntlet firewalls for protecting the DMZ. First I will discuss the rules on the Internet access firewalls.

The purpose of these firewalls is to be a single managed point of access to the Internet for GIAC's network. Everything destined for the Internet should funnel through this point. For Internet web browsing, the company will use internal proxy servers not shown in the diagram. There are several advantages to using an actual proxy server, in addition to a firewall. With a decent proxy server solution you can have greater control and logging of web traffic. The proxy server can use a remote database, like ldap , to provide authentication and then log all traffic of the authenticated user. While Gauntlet has the ability to force users to authenticate for web browsing and can offload that to an ldap server, it will not provide as much detail as a good proxy server. Additionally the proxy server can contain a list of blocked sites to help ensure that inappropriate sites are not being visited with company resources. Again, Gauntlet can do this, but I prefer to not

increase the load on the firewalls for this. The biggest advantage to using a proxy server is caching. Not only does this reduce the wait time of highly visited web pages, but every page that can be served up from the proxy server's cache, is one less request utilizing your precious Internet connections. This lowers the needed capacity of the rest of the Internet environment.

With proxy servers in place, the firewall only needs to let the proxy server, or servers, through to the Internet. The firewall's only web related roles are to inspect the traffic as it comes through, protect the GIAC network from the Internet and NAT the proxy servers' internal IP addresses to public IP addresses. GIAC allows JavaScript, but not Java Applets or ActiveX, so the firewall will administer this as well. GIAC will also use these Gauntlet firewalls for its ftp gateway. The firewall will use a remote ldap database for users to authenticate to it before proceeding to ftp sites on the Internet. Additionally, these firewalls will also be the gateway for Internet email. Using Gauntlet's smap process and the latest version of sendmail, the firewall will be able to securely relay incoming and outgoing email while successfully implementing anti-spam and anti-relay features.

With Gauntlet, you have to first define your network group(s). A network group holds a list of IP addresses like 10.33.95.233 and/or networks like 10.33.95.* or 10.33.*. Unfortunately, the networks do not contain a netmask, so you're limited to listing networks in increments of 8 bits like 10.*, 10.10.*, or 10.10.10.*. Then, you have to setup up your services. Services are things like http and smtp. You can also create custom services like tn3270 on port 1023. Next, you can create service groups to coordinate services similar to how network groups coordinate networks. It should be noted that Check Point has a more intuitive way of building these components and component groups. You can create service groups either before or after creating your services, but when networks are created they have to be assigned to a network group. This means the network groups have to be created first. Also, you can not assign a service to a service group from the service configuration screen. The service group must be opened up so that services can be placed in them.

Now that your services, networks, and their groups are defined, you can build your rule set. This is a little different from Check Point. You have source rules and destination rules in Gauntlet. An example of a source rule is saying network group X is allowed to use all the services in service group Y. However, this source rule is not enough to allow traffic to pass. A correlating destination rule, or set of destination rules, is needed to say what destinations group X can perform services Y on. Let's apply this to what is needed for the firewalls proxying employee traffic.

I will create a network group called ProxyServers, create a network object for each proxy server, and then place it in that network group. I will also need a generic 10.* network and I will place that in a network group called GIAC. Then I will setup the http proxy to only allow JavaScript (denying Applets and ActiveX) and put it in a service group called Web. I will also activate the ftp proxy and place it in a service group called FTP. The Web service group will not have any authentication turned on, as our proxy

servers are taking care of all this, but the FTP service group will use LDAP authentication. I will also setup the smtp proxy. This proxy has many anti-relay and anti-spam features, so I will need to configure all of this. Then I can build a SMTP service group with the smtp (smap and smapd) proxy in it and SMTP will have no authentication. Lastly, I will create an AnyNet network group with a network of * in it. Now I am ready to create the source and destination rules. I will create a source rule allowing ProxyServers to use Web, a rule allowing GIAC to use FTP, and a rule allowing AnyNet to use SMTP. Then I'll create a destination rule permitting Web to go anywhere (*), a rule permitting FTP to go anywhere, and a rule permitting SMTP to go anywhere.

The other set of Gauntlet firewalls are for protecting GIAC's DMZ. The only forms of access for GIAC's customers, suppliers, and partners will be web and ftp. The customers will most likely only need web access; however, suppliers and partners might need the ability to drop off product via ftp. You should never allow ftp access directly into the internal network. The DMZ provides a good place for an intermediate ftp server. The Internet user puts the file(s) on the ftp server and the internal GIAC employee will ftp to that same server to retrieve the data. The DMZ is also a good place to house your company's external DNS servers, so the firewalls will need to allow at least udp 53 traffic, if not tcp 53 as well. Let's suppose the DMZ network is 10.50.200.0/24, with the following servers in the DMZ:

10.50.200.51 - External DNS server 1 hostname dn00ext1
10.50.200.52 – External DNS server 2 hostname dn00ext2
10.50.200.101 - web server 1 hostname sv00web1
10.50.200.102 - web server 2 hostname sv00web2
10.50.200.111 - ftp server 1 hostname sv00ftp1
10.50.200.112 – ftp server 2 hostname sv00ftp2

I will need to create a network group called AnyNet, put a * network in it, and then create a network group called FTP containing networks approved to ftp in. Then I will need to configure the http service, ftp service, and create a tcp 53 plug called dns. Typically, tcp 53 means zone transfers, but not always, so we'll let it through and make sure the DNS server is configured to not allow unauthorized zone transfers. Since there are basically no udp proxies (there is a limited snmp proxy) in Gauntlet 5.5, I will have to create a static filter. I will have to allow 0/0 on any port to come through the outside interface and talk to 10.50.200.51/32 on udp port 53, and also allow 0/0 on any port to talk to 10.50.200.52/32 on udp port 53. Additionally, I will need two more rules to allow 10.50.200.51/32 and 10.50.200.52/32 on udp port 53 to come through the internal interface talk to 0/0 on any port. Unfortunately, I will get no logging for static filters. On the upside, Gauntlet 6.0 will not only proxy UDP (which will give us logging and more intelligent inspection), but will log static filters as well (for things like protocols 50 and 51).

Now, I need to configure three service groups. I will call them Web, DNS, and FTP. There will be three source rules; one to allow AnyNet to use Web, another to allow FTP network group to use FTP service group, and a final one to allow AnyNet to use

DNS. Likewise, I will have six destination rules. One to permit Web to go to 10.50.200.101, another to permit Web to go to 10.50.200.102, one to permit FTP to go to 10.50.200.111, another to permit FTP to go to 10.50.200.112, one to permit DNS to go to 10.50.200.51, and one last rule to permit DNS to go to 10.50.200.52.

Now for the Check Point firewalls segmenting the DMZ from the GIAC internal network. The internet web servers will probably communicate to MTS via http/https traffic to save and retrieve information, etc., so we'll need to allow that type of traffic. http and https are excellent methods of communicating to the mid-tear through firewalls. It is a single, manageable, tcp port, allowing information to be tunneled through it (DCOM etc.). However, software designers will inevitably design products using other types of traffic, so be prepared to allow it if necessary. I will also need to allow ftp access from the private network to the ftp servers in the DMZ. Beyond that, nothing else should be let through except traffic needed to manage the servers in the DMZ (ssh, PCAnywhere, etc.).

Next, I will do a tutorial of setting up the Check Point firewall rules. As I mentioned above, I will allow DMZ servers to initiate http and https communications to MTS servers on the backend. I will also allow ssh traffic from two management servers on the inside to the DMZ servers.

Back on the internal network I will say that the MTS servers, sv00mts1 and sv00mts2, reside in network 10.50.100.0/24 and the MTS farm are IP's 10.50.100.101 – 10.50.100.110. The two management servers, sv00ctl1 and sv00ctl2, and the remote Check Point management server, fw00adm1, reside in a 10.50.110.0/24 network and are 10.50.110.41, 10.50.110.42, and 10.50.110.50 respectively. I will suppose the two Check Point firewalls, fw00dmz1 and fw00dmz2, have external interface IP's in the DMZ of 10.50.200.251 and 10.50.200.252, and internal interface IP's of 10.50.50.251 and 10.50.50.252 on a 10.50.50.0/24 network. I will assume that the Check Point remote management server is already set up and that the remote client I am connecting from is already allowed to connect to the management server.

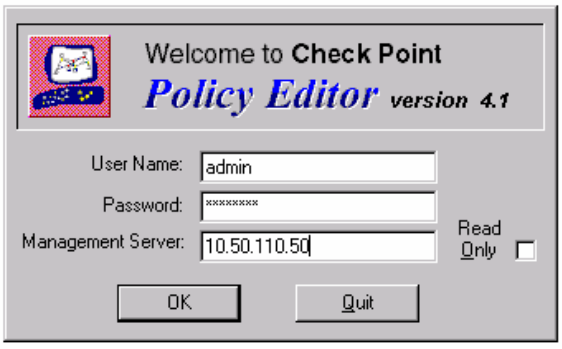


Figure 2.01

First I must log onto the management server. As Figure 2.01 shows, you have to enter a user ID, password, and hostname/IP address of the management server. Once logged in, I will first need to create objects for all the servers/networks you want to list in

the rules. To do this, click on Manage | Network Objects. This will bring up a new window like Figure 2.02 shows. I will need to create new objects for each server listed above. Figure 2.03 shows an example of a server in the DMZ and a server on the inside network. Notice that I've given the server residing in the DMZ a location of "external" and a color of "red", and the internal server a location of "internal" and a color of "green". This is because the DMZ is "external" to the Check Point firewall, and the inside network is "internal". The particular color is not as important as having a color to represent different networks. Since the admin interface is a gui, and everything is represented as objects, the difference in color helps coordinate things.

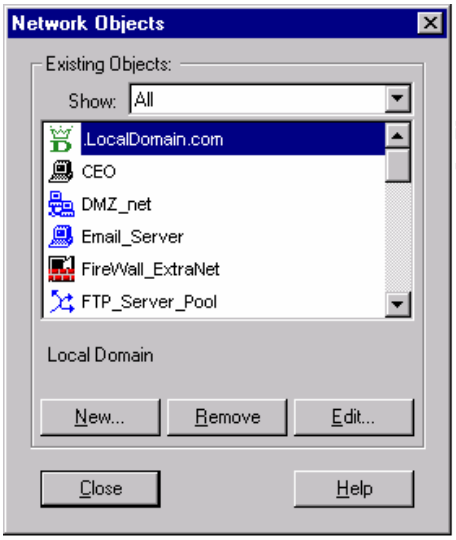


Figure 2.02

I can't forget that I have to create objects for the firewalls themselves too. Figure 2.04 shows an example of this. I have chosen a location of "internal", as well as listing the internal interface's IP address in the address field. This is because the management server will be managing it from the internal network, versus the external network. I am going to create groups for the firewalls and the MTS servers, to keep the rule set clean. You will see this later.

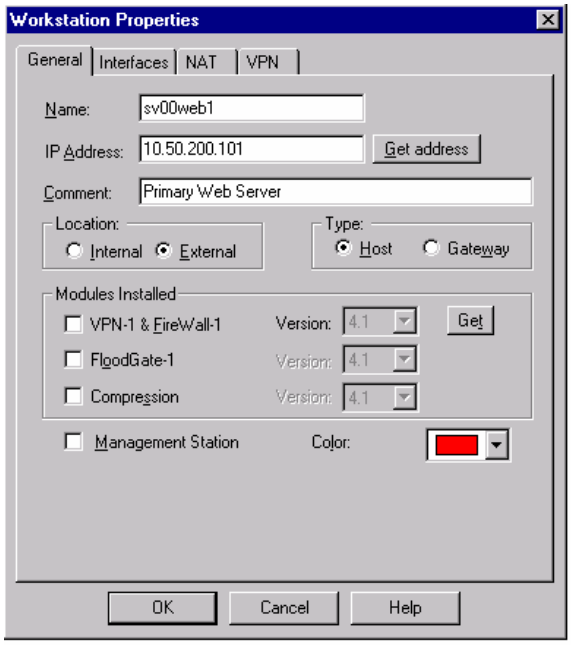
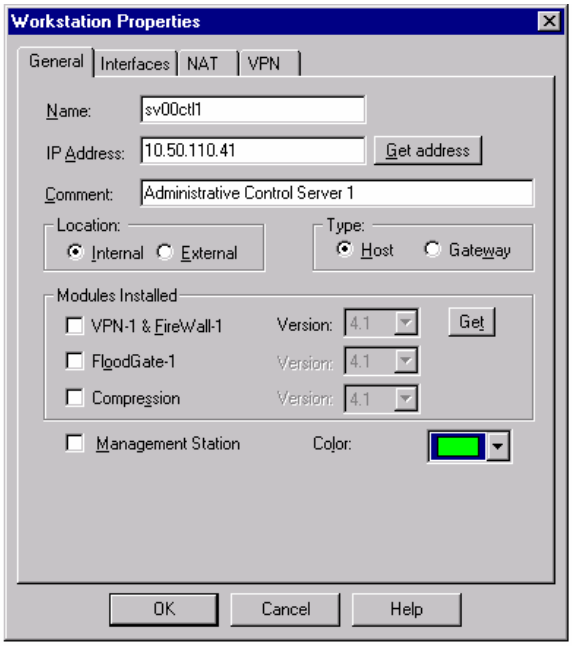


Figure 2.03



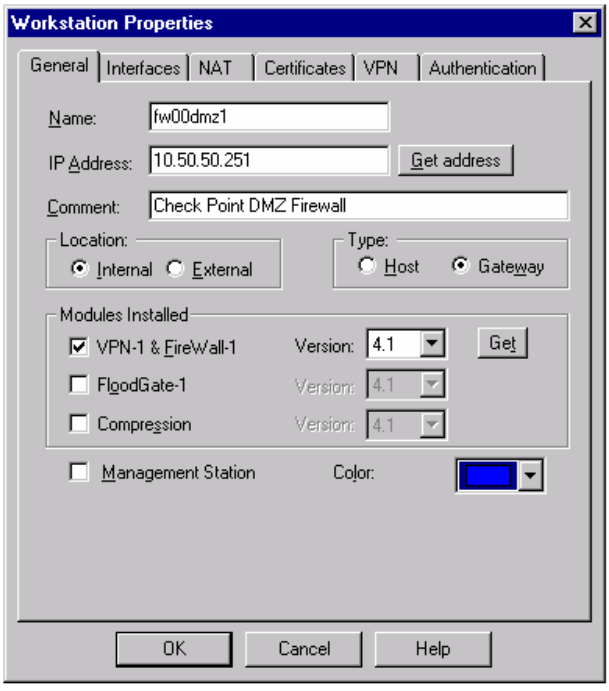


Figure 20.4

Now that all the objects are created, I can add the rules. Click on Edit | Add Rule. Either bottom or top is OK, as this is the first rule. I will end up with a blank rule like Figure 2.05. Then I can right click on the individual columns and use the pop up window, as shown in Figure 2.05, to fill in the necessary information. Once completed I will end up with a rule-set like Figure 2.06.

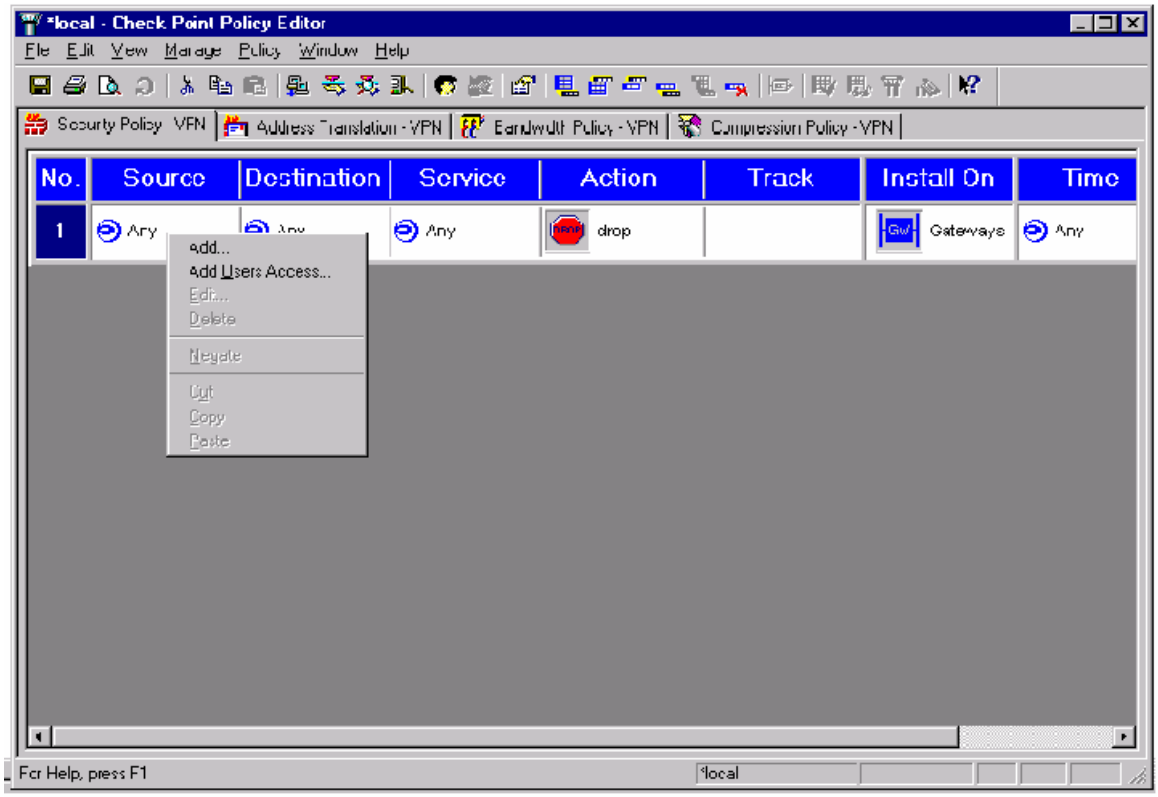
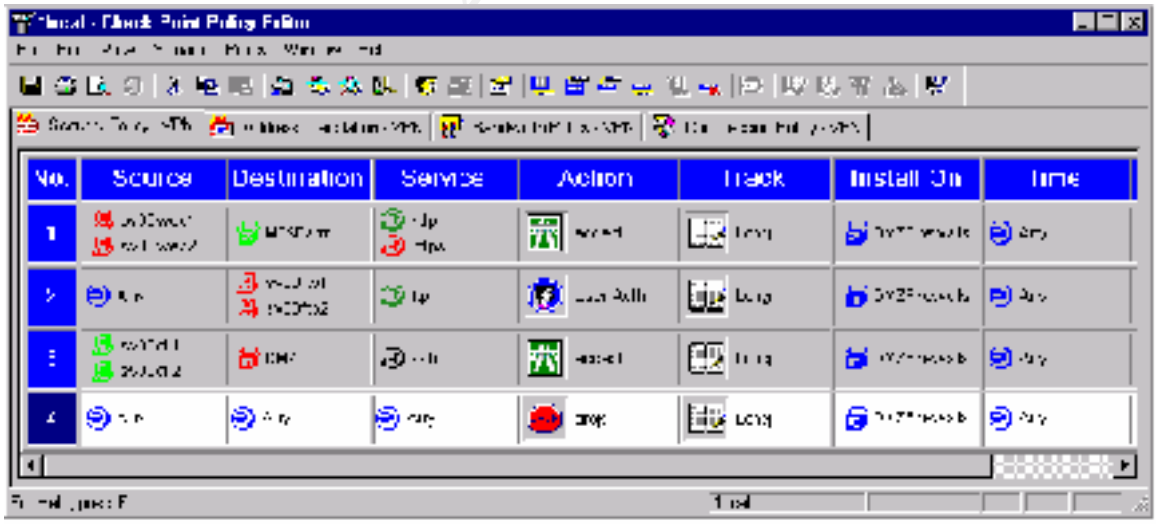


Figure 2.05



Finally, I need to configure the VPN devices. These devices will be used to provide remote access for remote employees and business to business (B2B) tunnels for business partners and remote sites. I will need to allow our employees and other on sight personnel, access to GIAC's network. To do this I will use a PKI solution so that there is strong authentication to keep out unauthorized people. For the B2B connections, I will use pre-shared secrets. Of course, these secrets will need to either be passed out of band,

or encrypted before transferring. I will allow B2B connections between GIAC's main site and its remote site, GIAC's main site and its business partner, and GIAC's remote site and its business partner.

The Nortel Contivity has firewall capabilities. These capabilities even allow for restricting (by user) destination addresses and services. So, for more control and protection, these types of rules can be added.

I must not forget about the firewall protecting the leased line connection to GIAC's remote site. As to what needs to be allowed through, is hard to define. As both sides of the connection are GIAC's network, it may need to be a fairly open firewall. At the very least, I will need to have similar rules as the Check Point firewalls at the back of the DMZ, so that matching web servers in a DMZ at the remote site can contact backend/mid tear servers at the main site, and the inverse, in case one set of servers at a particular location is down. In retrospect, this firewall is probably not needed with the current design. If the different locations housed different functions, then it might be important to segment access.

In conclusion, this assignment has discussed the configuration of the security devices that make up the architecture designed in assignment #1. This assignment listed router ACLs, firewall rules, and discussed VPN configurations. A brief tutorial was given for the Check Point firewall configuration.

Author: Cory Steers
Date: 03/02/2001
Level Two Firewalls, Perimeter, Protection, and VPNs
Assignment #3 Audit the Security Architecture

The purpose of assignment #3 is to provide technical support for a comprehensive information systems audit of GIAC Enterprises. The audit is limited to the Primary Firewall. Support is needed for planning the assessment, implementing the assessment and conducting a perimeter analysis.

First I will plan the assessment. The audit is limited to the Primary Firewall, but there are multiple firewalls at the perimeter level. They are all the same type of firewall, but providing two different functions. Both sets of firewalls should be assessed.

The firewalls should be scanned off shift, one at a time. The Alteon Switches can be configured to remove specific devices from the load balancing "loop". As long as they are scanned one at a time, when utilization is low, no outages or periods of slow response should occur.

I decided to search the Internet looking for vulnerabilities. The TIS web page for Gauntlet 5.5 patches listed several patches www.tis.com/support/patch55.html; however, it was hard to tell if the patches were fixing security bugs or not. On Security focus I did a search for Gauntlet vulnerabilities (www.securityfocus.com). I could only find 3 bugs, and only one of them was for version 5.5. Furthermore, the vulnerability requires the firewall to be running CyberPatrol. CyberPatrol is a product for restricting access to specific sites. GIAC is using proxy servers to provide this functionality so it will be turned off. The SANS web site (www.sans.org) mentioned Gauntlet being vulnerable to Jolt2, but only on the WebShield products and NT servers. Since GIAC is running its firewalls on servers, and those servers are not running Windows OS, GIAC is not susceptible to this exploit either.

Since I was unable to find any exploits for Gauntlet that matched the version and platform GIAC was running, I decided to concentrate on the SANS Top 10 vulnerabilities. Those are:

1. Spoofing/Spoofed Addresses
2. Login Services (ftp, ssh, telnet, NetBIOS, r-services, etc.)
3. RPC and NFS (portmapper, rpcbind, NFS, lockd)
4. NetBIOS
5. X Windows (ports 6000 – 6255)
6. Naming Services (DNS, LDAP, etc.)
7. Mail (smtp, pop, imap)
8. Web (http, https/ssl, proxy ports like 8000, 8080, etc.)
9. Small Services (time, tcp/udp ports below 20)
10. Miscellaneous Protocols (tftp, finger, nntp, ntp, lpd, syslog, etc.)

One of the largest threats to a Gauntlet firewall is services running on the server itself. By default, with no rules in the firewall, Gauntlet denies absolutely everything to the network(s) it's protecting; however, by default, it denies nothing destined for the firewall itself. It has an extensive script that runs during install to disable most of the services mentioned above. This script is not foolproof. Fortunately, this concern is covered by the SANS Top 10 as well.

GIAC's Internet routers should be protecting GIAC's network from being spoofed from the Internet; however, the routers might get compromised and the firewall might get spoofed from the inside. To test this we could simply put a machine on the same immediate network segment as the firewall with an incorrect network configuration. There are also tools, such as NAI's CyberCop, that can send spoofed packets.

To check for the other 9 vulnerabilities we can first use a tool like nmap to determine if the potentially vulnerable services are running. Nmap is an excellent tool to scan for listening services. Once the services listening for remote connections are found, we can determine what potential vulnerabilities are in GIAC's network.

The costs of performing this assessment, should be low. Most of the costs will come from consultant and employee fees. There is no need for expensive equipment to perform these tests.

Now it's time to implement the assessment. We will start with one of the firewalls proxying Internet traffic for GIAC's network. Vulnerability #1, spoofing, was attempted by running an nmap scan with the –D option. This option is for confusing intrusion detection by making the scan look like it's coming from multiple hosts. This will attempt to hide you as the real scanner. In some instances, if you specify your real address to be used after several fake addresses, your address won't show up in IDS logs at all. My purpose is to simply give the firewall a source address that should not be seen on the outside interface.

The result was the following log message in Gauntlet:

Mar 7 09:50:25 hostname gfw: securityalert: source not allowed on interface: TCP if=qfe0 srcaddr=10.1.1.5 srcport=1049 dstaddr=195.62.178.10 dstport=80

This alert is saying that a packet destined for tcp port 80 at 195.62.178.10 had a source of 10.1.1.5. Since the firewall expects 10.* addresses to be on the internal interface an not the external interface, it denied the request.

Initial discovery of the remaining top 10 vulnerabilities can be covered with a simple nmap scan. Since I was not concerned with being detected, the first scan I ran had options –sT and –sU. Option sT says to do a basic tcp port scan on all "interesting" ports. Interesting means all ports listed in the nmap-services file. This includes all ports below 1024 and several others totaling 2042. Option sU causes a basic udp port scan on all "interesting" ports. Here are the results of this scan:

Starting nmap V. 2.53 by fyodor@insecure.org (www.insecure.org/nmap/)
Interesting ports on (195.62.178.10):
(The 3074 ports scanned but not shown below are in state: closed)

Port	*State*	*Service*
21/tcp	*open*	*ftp*
25/tcp	*open*	*smtp*
43/tcp	*open*	*whois*
80/tcp	*open*	*http*
113/tcp	*open*	*auth*
123/udp	*open*	*ntp*
443/tcp	*open*	*https*
514/udp	*open*	*syslog*

Nmap run completed -- 1 IP address (1 host up) scanned in 11 seconds

As you can see, there are several ports "listening" on the firewall. If I type a "netstat –an" from a command prompt, you'll see the following services listening.

tcp	*0*	*0*	**.80*	**.**	*LISTEN*
tcp	*0*	*0*	**.443*	**.**	*LISTEN*
tcp	*0*	*0*	**.21*	**.**	*LISTEN*
tcp	*0*	*0*	**.25*	**.**	*LISTEN*
tcp	*0*	*0*	*127.0.0.1.9696*	**.**	*LISTEN*
tcp	*0*	*0*	**.8004*	**.**	*LISTEN*
tcp	*0*	*0*	**.43*	**.**	*LISTEN*
tcp	*0*	*0*	*127.0.0.1.7777*	**.**	*LISTEN*
tcp	*0*	*0*	*127.0.0.1.23*	**.**	*LISTEN*
tcp	*0*	*0*	*10.50.110.10.23*	**.**	*LISTEN*
tcp	*0*	*0*	**.113*	**.**	*LISTEN*
tcp	*0*	*0*	*10.50.110.10.22*	**.**	*LISTEN*
udp	*0*	*0*	*195.62.178.10.123*	**.**	
udp	*0*	*0*	*10.50.110.10.123*	**.**	
udp	*0*	*0*	*127.0.0.1.123*	**.**	
udp	*0*	*0*	**.123*	**.**	

Notice that there are a lot more services running than what nmap found. Since we ran our nmap scan from the Internet side, it will only find services listening as *.<port> or 195.62.178.10.<port>. A lot of the services are listening as 10.50.110.10.<port>, which is its internal interface, or 127.0.0.1.<port> which is the loopback interface. However, even this does not account for all the differences. Notice that the firewall is listening on all interfaces (*) for port 8004, but nmap didn't catch it. Port 8004 was not in nmap's services file, so it didn't check for that port. One final thing to take note of is that nmap found udp 514, but it is not listed in the "netstat –an" output. I was under the impression that by commenting the syslog (udp 514) entry out in the /etc/services file,

that syslog would stop listening remotely. It appears that I was wrong. For some reason, it did hide it from netstat … probably a bug. This udp 514 is important because prior to Gauntlet 5.5, the netperm-table, which is one of Gauntlet's configuration files, automatically contained an entry blocking access to that port; however, in 5.5 they did not, which required administrators to add it manually.

It is important to note that I did several other scans with nmap for the sole purpose of trying to evade logging by Gauntlet. The SYN scan came close, as the logs were delayed … probably until the OS closed the half open ports. I also performed a FIN scan, an ACK scan, and an Xmas tree scan. None of these three even correctly found processes running. Gauntlet will not fool a good port scanner, but it will log it. Just because nmap finds an open port does not mean Gauntlet will allow access to that service.

Now, for the firewalls protecting the DMZ. If you remember from illustration 1.02, these firewalls are not directly plugged into the 195.62.178.0/24 network. They have a layer of Alteon's in front of them. These firewalls have 10.* addresses on the external interfaces. This makes it extremely hard to scan these firewalls. Aside from unplugging them from the Alteon switch and plugging it into your probe machine, via crossover cable, there is no good way to reach them. The Alteon will most likely have one or more virtual IP addresses (VIPs) that serve as the destination for the services being provided by the DMZ. The only way to get to the firewalls at all, is to build static routes on your probe machine making these VIPs the gateway for the network that the firewalls are on; however, doing this does not necessarily make them accessible. The Alteon switch is much more intelligent than something like a Cisco Local Director. The Alteons are only going to accept http and ftp traffic on those VIPs so that is all the nmap scan should report. Of course, to scan the Alteons themselves, as well as trying to scan through them (scan the VIPS) would be a good idea, but to discuss it in detail is outside the scope of this assignment. We will just assume that these firewalls have the same ports listening, but have an extra layer of protection with the Alteon switches. These firewalls will not be listening on smtp port 25, as that service is not running on them.

Now that we know what ports may be vulnerable, we can verify whether or not they are, while also verifying that the firewalls are enforcing the security policy. To do this, we will simply try to run an appropriate application to connect to the services running. Again, we will start with the firewalls proxying Internet traffic for GIAC's network. Here is the services that are running and listening on the outside interface:

http – tcp 80
https - tcp 443
ftp - tcp 21
smtp - tcp 25
whois - tcp 43
auth - tcp 113
ntp - tcp 123
syslog - udp 514

The processes http, https, smtp, and ftp, are owned by Gauntlet proxies. Since our rules, with the exception of smtp, are for inside IP's to talk outside, no attempts to these ports will reap any rewards. You will simply get a series of "connection refused" messages on the client and security alerts on the firewall; however, smtp will answer. In fact, you are free to relay mail to these firewalls or even telnet to the port and type in sendmail commands. Of course, you could try to exploit it. The program listening on port 25 is not sendmail, but Gauntlet's smap program. It is much simpler and far more secure. There are no known exploits for it at this time. Let's focus on the non-Gauntlet services.

Remember, it is possible that Gauntlet is denying access to services that nmap found open, so you will actually need to further test to know the extent of the vulnerability. This does not mean you do not need to worry about disabling unused ports, as a future configuration mistake could suddenly leave you exposed.

To see if Gauntlet is allowing remote logging to the server, I simply tried to send a log message to it. First I set up /etc/syslog.conf on my machine with the following line:

user.debug @firewall

I then restarted syslog, and used logger to send the following message:

logger –p user.debug –t TEST "test message"

If Gauntlet is not protecting this service, then your syslog message file will have something with the words "test message" in it. If Gauntlet is protecting this service, then you will get a message like:

Mar 8 16:03:54 hostname gfw: securityalert: packet denied by local screen: UDP if=lan1 srcaddr=10.50.95.37 srcport=514 dstaddr=10.50.101.10 dstport=514

Similar types of things can be done for whois, auth, and ntp. It has been debated as to whether or not to run auth. If you are paranoid, you will turn it off; however, it is not the most insecure thing in the world to leave it running either. Sendmail servers like when it is on and these firewalls are proxying mail. There is no excuse for whois. It should NOT be running on a firewall. Ntp is debatable. It might be needed to keep server's clocks in sync. You can always leave it running and deny access to it from the outside, via Gauntlet.

Assuming that the Gauntlet firewalls protecting the DMZ have the same vulnerabilities, we will need to fix them in the same way. Remember, due to the Alteon switches, it is both easy and difficult to test the rules on these firewalls. Assuming the Alteon's are correctly configured, it is possible that the firewall could be miss-configured and not even know it from a remote perspective. Nonetheless, to be thorough, a full scan should be attempted of the eternal network to find machines to exploit. This could be

done in a similar way as the other firewalls; therefore, there is no need to further discuss this in detail.

The results of the perimeter analysis can be summed up like so: For the most part, the perimeter of GIAC's network is secure. A few possible vulnerabilities were found at the firewall. I recommend that the vulnerable services be disabled on the firewalls and plans be made to periodically analyze the firewall devices on a reoccurring basis. As an added precaution, GIAC should also add more ingress filters to the Internet border routers to further protect the firewalls. Beyond that, I feel that GIAC's perimeter network design is sound and should not be altered beyond the changes just mentioned.

In conclusion, this assignment has provided technical support for a comprehensive information systems audit of GIAC Enterprises. Support was provided for planning the assessment, implementing the assessment and conducting perimeter analysis of GIAC's primary firewalls.

Author: Cory Steers
Date: 03/12/2001
Level Two Firewalls, Perimeter, Protection, and VPNs
Assignment #4 Audit the Security Architecture

The purpose of this assignment is to select a previously posted GCFW practical and attack it. The attack will have three parts to it. First, I will attack the firewall itself trying to find vulnerabilities in the product chosen. Next, I will initiate a distributed denial of service (DDoS) attack against the design, limited to tcp SYN, udp, or ICMP floods. Finally, I will attempt to compromise an internal system through the perimeter.

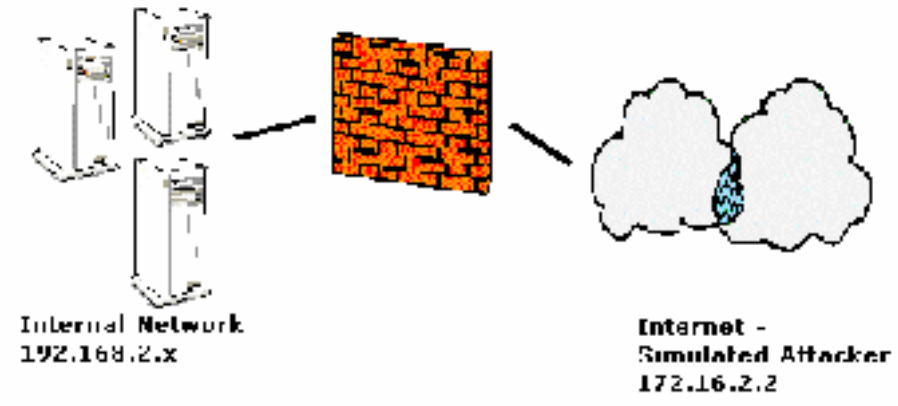
The GCFW practical that I have chosen is Dujko Radovnikovic's practical, created September 18, 2000 (www.sans.org/y2k/practical/Dujko_Radovnikovic.doc). Dujko's design consists of a Linux ipchains firewall, protecting a service network. The firewall rules, as Dujko's listed them, is not allowing any access to the screened network. Since this is secure, but impractical, I will simply assume that the rules to allow access to a DNS server and a web server, which are commented out, are not commented out, and therefore allowing that access. Here is a list of the ipchains rules. Remember, the rule to allow DNS and http are no longer commented out.

```
#!/bin/sh
#
# rc.firewall
#
SCREENEDIF="eth1"
SCREENEDNET="192.168.2.0/24"
SCREENEDIP="192.168.2.1"
#
EXTERNALIF="eth0"
EXTERNALNET="172.16.0.0/12"
EXTERNALIP="172.16.2.1"
# Variable used to store location of ipchains binary
IPCHAINS="/sbin/ipchains"
#
# Enable IP Forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward
#
# Flush everything
# Incoming packets from outside network
$IPCHAINS -F input
# Outgoing packets from internal network
$IPCHAINS -F output
# Forwarding/masquerading
$IPCHAINS -F forward
#
# Spoofing and Source Route Protection
for x in /proc/sys/net/ipv4/conf/*; do
if [ -f $x/rp_filter ]; then
echo 2 > $x/rp_filter
```

```
fi
if [ -f $x/accept_source_route ]; then
echo 0 > $x/accept_source_route
fi
if [ -f $x/log_martians ]; then
echo 1 > $x/log_martians
fi
done
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
#
## Login Services
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 21 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 22 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 23 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 139 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 512:514 -l -j DENY
#
## RPC and NFS
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 111 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 111 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 2049 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 2049 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 4045 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 4045 -l -j DENY
#
## NetBIOS in Windows NT
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 135 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 135 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 137 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 138 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 139 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 445 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 445 -l -j DENY
#
## X Windows
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 6000:6025 -l -j DENY
#
## Naming Services
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 53 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 192.168.2.3 53 -l -j ACCEPT
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 53 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 389 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 389 -l -j DENY
#
## Mail Services
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 25 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 109 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 110 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 143 -l -j DENY
#
## Web
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 192.168.2.4 80 -l -j ACCEPT
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 80 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 443 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 8000 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 8080 -l -j DENY
```

```
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 8888 -l -j DENY
#
## Small Services
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 1:20 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 1:20 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 37 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 37 -l -j DENY
#
## Miscellaneous
#
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 69 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 79 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 119 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 123 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 514 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 515 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 161 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 161 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 162 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p udp -s 0/0 -d 0/0 162 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 179 -l -j DENY
$IPCHAINS -A input -i $EXTERNALIF -p tcp -s 0/0 -d 0/0 1080 -l -j DENY
#
# ICMP
#
$IPCHAINS -A input -i $EXTERNALIF -p icmp --icmp-type 8 -s 0/0 -d 0/0 -l -j DENY
$IPCHAINS -A output -i $EXTERNALIF -p icmp --icmp-type 0 -s 0/0 -d 0/0 -l -j DENY
$IPCHAINS -A output -i $EXTERNALIF -p icmp --icmp-type 3 -s 0/0 -d 0/0 -l -j DENY
$IPCHAINS -A output -i $EXTERNALIF -p icmp --icmp-type 11 -s 0/0 -d 0/0 -l –j DENY
#
# All data that is not specified to be accepted, and that does not match any of the DENY
# rules above will be effectively denied by the following rules.
#
$IPCHAINS -A input -j DENY
$IPCHAINS -A output -j DENY
$IPCHAINS -A forward -j ACCEPT
```

Here's an illustration of the network.



After some analysis and testing of Dujko's ipchains rules, I realized a few rules needed to be added to actually let communication to and from the web server and DNS server to happen. The current set of rules allows the appropriate traffic to the servers, but blocks those servers from being able to respond. So, I am making these additional changes:

$IPCHAINS -A input -i $SCREENEDIF -p tcp -s 192.168.2.4 80 -d 0/0 1024: -l -j ACCEPT
$IPCHAINS -A input -i $SCREENEDIF –p udp -s 192.168.2.3 53 -d 0/0 1024: -l -j ACCEPT

These rules can be added towards the top of the list, or put next to their corresponding inbound accept rules to allow the services to communicate. It should be noted that the DNS server may need to do some tcp 53 traffic outbound, as there are some other tcp type DNS requests besides zone transfers. It is inbound zone transfers you need to worry about. Additional rules would be needed to let this happen as well.

The firewall in Dujko's practical is running ipchains version 1.3.9 on Slackware Linux kernel version 2.2.16. So, I will first need to search for vulnerabilities with this configuration. Amazingly, after several hours combing the net for exploits, I only found a few. Most of the exploits relied on the Linux box running something like an ftp daemon for the exploit to work. Since a firewall typically has no extraneous services running on it, this makes my attack even harder. I decided to pick an attack based on a service that could accidentally be left on. For example, several services are controlled by inetd, so if inetd is left running and /etc/inetd.conf has not been edited, there is potential for several services to be running. The exploit I am trying is based on imapd, which can be configured to run through inetd. Keep in mind, for my exploit to work not only would Dujko have to accidentally leave some services running, but would have had to make a mistake on the ipchains rules too. In all likelihood this exploit, as well as most others, will fail.

The exploit I chose can be found at darknet.syntheticarmy.com/exploits/os/linux/slackware/7.1/lsub.c. This exploit attacks imap to try and gain shell access to the server. To use this exploit you have to:

1. *pull the source code from the net*
2. *compile it*
3. *execute it with the following parameters: <host> <login ID> <password> <type> [offset]*

"Host" is the hostname or IP address of the target server. Login ID is a potential user trying to email from the server and "password" is the password for that account. For the type field, there are choices 0 – 3, which correspond to various vulnerable distributions of imapd on Linux. 0 is for Slackware 7.0, and 1 is for Slackware 7.1. Choosing a type is really just choosing an offset for the exploit. The optional "offset" parameter is to further adjust the offset picked by the type parameter to try and make the exploit more flexible for other versions and distributions of imap.

Since I do not have Dujko's environment to attack, I can not say whether or not this exploit will work. If it works, then I will end up with shell access to the Linux server. Since it is doubtful that Dujko accidentally left any extraneous processes running, and equally doubtful that Dujko's ipchains rules would allow this access anyway, this exploit will most likely fail.

The second piece of the attack is to simply denial of service (DOS) the network. The DOS is restricted to tcp SYN, udp, or ICMP floods. I do; however, have at my disposal 50 machines (drones) that I have compromised to help me with my attack. This is better defined as a distributed denial of service (DDOS) attack.

I have decided to use the Tribe Flood Network 2000 (tfn2k) attack against Dujko's network. The code I am using can be found at packetstorm.securify.com/distributed/tfn2k.tgz. The steps involved in using this code are:

1. *untar the source*
2. *edit src/Makefile to uncomment the appropriate OS for the compile.*
3. *distribute the server daemon (td) to the 50 drones*
4. *build a file listing fully qualified hostnames/IP addresses of my drones*
5. *startup the client (tfn) on my control attack machine, and instruct the drones to attack Dujko's network.*
6. *stop the attack once I get bored, or have succeeded in whatever damage I was trying to accomplish*

The reason for editing the Makefile in step 2 is to compile the code for different machines. This code is flexible enough to run on multiple platforms. It is very unlikely that all 50 of my drones are running the same OS, so I would presumably need to change this Makefile and recompile for different operating systems. It is also quite likely I will need access to a machine of the same OS with a compiler to be able to successfully compile this for different OS.

This attack code is very flexible. With it, I can perform DDOS via tcp syn floods, udp floods and icmp floods. It also spoofs the source address which makes things more interesting.

What makes this type of attack even more alarming is that there is no real way to defeat it. If I have enough resources at my disposal to overpower Dujko's resources, then the attack will effectively block legitimate access to Dujko's network. By resources I mean network bandwidth and the size and number of servers, routers, and the like. It is important that you do not have more bandwidth than your infrastructure can handle, or you will panic your servers and routers causing them to crash. If your hardware matches or surpasses your network bandwidth, you are still in trouble, as an attack could saturate your network connection(s). The only real defense is to set your network up to keep the machines you are protecting from becoming "drones". This means you should try to keep them from being infected with the process and block the types of traffic that would

be coming from them, if they were infected. Of course, everybody in the Internet community needs to do this for it to be completely effective.

For the third piece of the attack, I chose to attack the DNS server that the firewall is protecting. Since this network was built prior to 1/30/01, it is possible that the DNS server has not been updated to account for the TSIG bug found in versions of bind prior to 8.2.3. The exploit I will try is documented at packetstorm.securify.com/0102-exploits/tsl_bind.c. To carryout the exploit, all I need to do is compile the code and execute it. If the DNS server is vulnerable, the exploit will overflow a buffer and I will be given shell access with permissions of the ID that named was running under. Hopefully, these permissions will be root or system. If the DNS server is not vulnerable, no buffer overflow will occur and the exploit will be unsuccessful.

In conclusion, I selected a previously posted GCFW practical and attacked it. First, I attempted to compromise the firewall by running an exploit against it. I then ran the DDOS attack tfn2k against the service network. Finally, I attempted to compromise a DNS server behind the firewall with the recent TSIG exploit.