Interested in learning more about security?

# SANS Institute InfoSec Reading Room



## Securing Extranet Connections

Doing business today not only means having all your internal systems connected, but it also means having external data connections to many of your business partners and customers. For smaller companies this may mean one or two private connections to other companies. For larger companies the number of external connections can grow rapidly. And for companies that rely on these external connections as an integral part of their business, the number can be in the hundreds. This paper will present one solution to securing a ...

**Securing Extranet Connections**

Jeff Pipping

GSEC Version 1.4, option 2

June 17, 2002

## Introduction

What is an Extranet? In the simplest terms possible, an *extranet* is a type of network that crosses organizational boundaries, giving outsiders access to information and resources stored inside the organization's internal network (Loshin, p. 14).

Doing business today not only means having all your internal systems connected, but it also means having external data connections to many of your business partners and customers. For smaller companies this may mean one or two private connections to other companies. For larger companies the number of external connections can grow rapidly. And for companies that rely on these external connections as an integral part of their business, the number can be in the hundreds.

This paper will present one solution to securing a large number of extranet connections. In particular, the focus will be on the corporation who is the extranet network provider, or at the hub of a large extranet. The extranet network provider's responsibility for security is not only between it and its business partners and customers, but also making sure that partner A can't access systems at customer B (unless specifically requested).

Therefore, this paper will discuss the following issues as they apply to extranet security:

- Limiting extranet access
- Protecting corporate assets
- Authentication and authorization
- Preventing unauthorized communications between customers and business partners
- Providing accountability for extranet access

## Case Scenario

This scenario involves a large corporation with an extranet that connects it to hundreds of customers and business partners. The extranet is actually a private frame relay network with PVCs (permanent virtual circuits) between the corporation and each customer and business partner. Also, there are some PVCs defined between various customers and business partners.

The corporation at the hub of the extranet, which acts as the extranet network provider for its customers and business partners, uses the extranet to access host systems located at

each customer site. The access may be as simple as a Telnet or FTP session, or may involve using a Windows based application such pcAnywhere, Citrix or SQL. Access into the customer's system is typically performed to provide support for the hardware or software on that system. However, there are also cases where our corporation uses the extranet to provide proactive monitoring of the customer's systems on a 24x7 basis. The connection the customer has to the extranet can also be used to connect them to other business partners on the extranet, if requested by both parties.

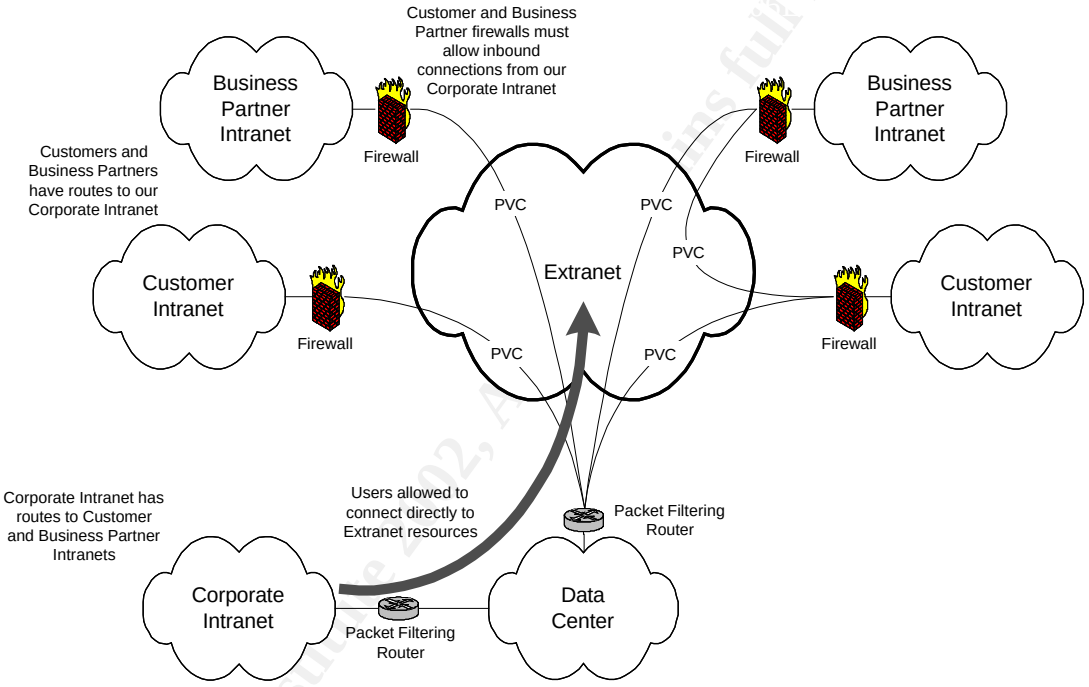
Because information in this industry is covered by federal regulations, it is critical that all available measures be taken to ensure data privacy. Therefore, our corporation acting as the extranet network provider, had to find a way to not only provide secure outbound connectivity from the corporate intranet to the remote customer locations, but also had to be able to offer a secure solution for allowing connectivity from partner A to customer B.

The solution to securing this extranet involved implementing controls at many different locations and layers of the protocol stack. At the network layer, the routing design was used to limit which networks could communicate with other networks. Also at the network layer, access lists were implemented to control the flow of information between routers. At the application layer, a Socks based application proxy server was implemented to authenticate and authorize connectivity of sessions from the corporate intranet to systems located at the remote customer sites. The Red Hat Linux operating system on the proxy server was hardened, as well as secured using Iptables. In addition, a stateful firewall was implemented between the corporate intranet and the extranet, to protect internal corporate resources.

Finally, security policies were implemented and more detailed information was logged to provide an audit trail and user accountability for accessing customer systems on the extranet.

**Before**

Before the security solutions described in this document were implemented, the only security that was in place for access into customer systems was at the host level at the customer site. When connecting to a customer system, a user on the corporate intranet would typically only be prompted for a user name and password to authenticate to the customer's host system. And occasionally the user wouldn't even get prompted to log in because the customer didn't have the host setup for authentication. The diagram below illustrates the layout of the extranet and how any user on the corporate intranet could connect directly to host systems at any customer site.




With this configuration, there was no central point for authentication; and authorization was dependent upon the operating system settings on the destination host. The only logs the customer had of who accessed their system was from the host or application logs on their system. Our corporation was not able to provide any access reports to the customer telling them who accessed their system and what they did. This was because access may have originated from one of several hundred PCs on the corporate intranet and it would have been difficult, if not impossible, to maintain and collect logs from every PC. Because of this, auditing and accountability were very weak and at times non-existent.

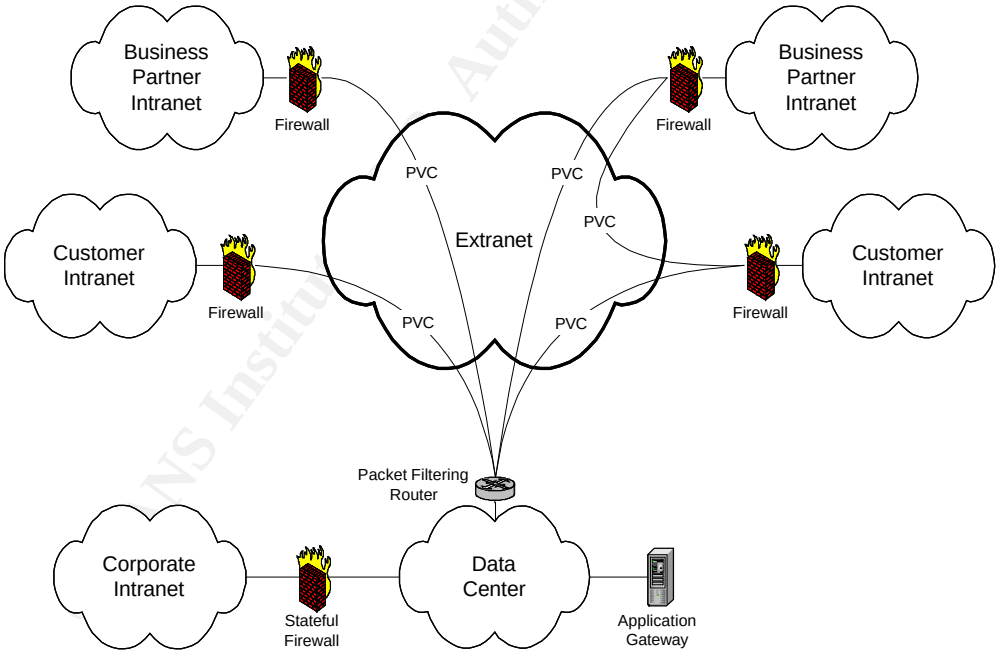
From a routing perspective, the corporate intranet needed routes to the customer intranets, and each customer's intranet needed routes back to our corporate network. So not only could any corporate user connect to a customer system, it was very easy for any user on a

customer network to attempt to connect to, or hack into, systems on our corporate intranet. To make things worse, there was no stateful firewall between the corporate intranet and the extranet, only a router with basic access control lists.

From the customer's and business partner's perspective, they had to configure their firewalls to allow inbound connectivity from our corporate intranet, which also made them feel insecure.

## Implementing the Changes

The next few sections will cover the changes that were implemented during the migration to the new extranet security environment. We will cover the security policies that were put in place, the new routing design, the router based access control lists and the stateful firewall configuration. We will also cover setting up the application proxy server with Red Hat Linux 7.2, hardening the OS, implementing iptables, configuring the proxy server software and explaining how the logs provide user accountability. Below is a diagram illustrating the logical layout of the extranet with these proposed security solutions in place.



**Security Policies**

In an extranet environment, where one particular organization, i.e. the extranet network provider, is shouldering the design, implementation and administration of the extranet, most of the security policy enforcement also lies with this organization. The extranet network provider is responsible for creating legal documents that must be signed by all extranet customers and business partners. In this scenario, our organization is the one that develops and maintains these security policies. All clients and business partners on the extranet must sign this agreement. In addition, if two or more customers or business partners request that communications be opened up between their locations, they must sign an additional agreement stating that they authorize this connectivity. It also states that the responsibility of securing this connection at each end belongs to the customers or business partners involved, and not with the network provider.

According to the "Information Security Management Handbook", the extranet security architecture and policies should support the following statements:

- Secure network connectivity must be provided using a dedicated line or using a VPN.
- The extranet must NOT provide a routable path to the participant networks (i.e., the extranet provider's network should not allow packets to flow between partner networks).
- The extranet must be securely partitioned from the corporate intranet.
- Extranet users must be uniquely identified using adequate authentication techniques.
- Authorization must adhere to the least-privilege principle.
- Extranet managers will receive monthly access reports to verify the proper use of the network. (Tipton, p. 104).

As you continue reading through this document, you will see our solution has addressed all of these issues.

**Routing Design**

Located at the edges of the frame relay network, including the hub and all remote locations, are Cisco routers which are owned and managed by the communications company that provides the frame relay network. These routers are configured based on design and security requirements set forth by our organization, which acts as the extranet network provider. The router configurations are only accessible by the communications company. For troubleshooting and auditing purposes, our corporation is allowed to view the current configuration of any of the routers.

To maintain standards and avoid IP addressing conflicts, all clients and business partners are required to present registered addresses on the extranet. Because most locations use private addressing (RFC 1918) on their internal network, this requires them to NAT their internal addresses to external, registered addresses. This provides added protection for the clients and business partners, since they are not exposing the addressing scheme of their internal network to the hub or other business partners.

As a first line of defense in our extranet security solution, static routes were implemented across the extranet to closely control which networks can communicate with other networks. There are no default routes and no dynamic routing protocols. The standard configuration of a remote site router has static routes to the network segment at the hub that hosts the application proxy server, and also routes for any internal networks that must be accessible from the hub.

```
route 2.2.2.0 255.255.255.0 Serail0.1  (route to proxy server subnet at hub)
route 10.1.2.0 255.255.255.0 10.1.1.2  (router to internal network)
```

If a customer requests connectivity with another business partner, a static route is added to the frame relay router at their location for the network(s) they need to access at the business partner. This is of course after both parties involved sign agreements, stating that they will allow this to happen and are responsible for the security of the connection. This is an example of a route that would be added on the customer's router:

```
route 3.3.3.0 255.255.255.0 Serial0.1  (route to business partner network)
```

This is an example of a route that would be added on the business partner's router:

```
route 6.6.6.0 255.255.255.0 Serial0.1  (route to customer network)
```

**Router Access Control Lists**

As a second line of defense, access lists are implemented on the serial and ethernet interfaces of each of the remote routers. These access lists first block any direct connectivity to the router, other than from the communications company. Second, they only allow inbound access from the proxy server at the hub and outbound access from the customer's internal networks. For example:

```
interface Serial0.1
access-group in_serial

access-list extended in_serial
  deny ip any host 0.0.0.0
  deny ip any 127.0.0.0 0.255.255.255
```

```
  deny ip any host 255.255.255.255
  deny ip 4.1.1.0 0.0.0.255  (deny LAN addresses on serial intf. - anti-spoofing)
  permit ip 1.1.2.50 255.255.255.255 1.1.1.254  (management access into router)
  deny ip any host 1.1.1.254  (deny all other access into router)
  permit ip host 2.2.2.10 any  (allow access from proxy server)
  permit tcp any any established  (allow packets from established connections)
  deny ip any any  (deny all other traffic)

interface ethernet0
access-group in_ethernet

access-list extended in_ethernet
  deny ip any host 0.0.0.0
  deny ip any 127.0.0.0 0.255.255.255
  deny ip any host 255.255.255.255
  deny ip any host 4.4.4.1  (deny all access into router)
  permit ip 4.4.4.0 255.255.255.0 any  (allow access from customer NAT range)
  deny ip any any  (deny all other traffic)
```

If a client requests connectivity with another business partner, a permit statement may be added to the serial access list, if the business partner requires inbound connectivity to the customer's internal networks or systems.

```
  permit ip 3.3.3.0 255.255.255.0 any  (allow access from business partner)
```

The frame relay router at the hub also has access lists to only allow outbound access from the proxy server, and to only allow inbound access to the proxy server. In addition, it has filters to prevent spoofing, direct router access (except from the communications company) and also blocks private, RFC 1918 addresses.

```
access-list extended in_serial
  deny ip any host 0.0.0.0
  deny ip any 127.0.0.0 0.255.255.255
  deny ip any host 255.255.255.255
  deny ip 10.0.0.0 0.255.255.255
  deny ip 172.16.0.0 0.15.255.255
  deny ip 192.168.0.0 0.0.255.255
  deny ip 2.2.2.0 0.0.0.255  (deny LAN addresses on serial intf. - anti-spoofing)
  permit ip 1.1.2.0 0.0.0.255 1.1.1.25  (management access into router)
  deny ip any host 1.1.1.25  (deny all other access into router)
  permit ip any host 2.2.2.10  (allow access to proxy server)
  deny ip any any

access-list extended in_lan
```

```
permit ip host 2.2.2.10 any
deny ip any any
```

**Stateful Firewall**

To protect the internal resources of the corporation from other systems on the extranet, a Cisco PIX firewall was put in place between the internal networks and the connection to the extranet. The PIX firewall was chosen over other stateful firewalls for a couple reasons. First, we already had several Cisco PIX firewalls within our infrastructure, so the support engineers were familiar with the user interface and configuration commands for the PIX firewall. This minimizes the chance of a configuration error on the firewall because someone is unfamiliar with the proper command or syntax to use when creating an access rule. Second, based on experience gained from the existing firewalls, the PIX had proven to be able to handle large traffic loads with minimal impact on CPU and interface utilization. A Cisco PIX 525 was already being used for our Internet connection, which is a full T3 running at 30 to 35 Mbps utilization, and the CPU utilization on the firewall was averaging around five percent. The PIX 525 has an advertised maximum throughput of 360 Mbps. Additional information and specifications about the PIX firewall can be found at the following link:

http://www.cisco.com/warp/public/cc/pd/fw/sqfw500/

The PIX firewall between corporate and the extranet has been configured to only allow TCP port 1080 traffic (Socks protocol) out from the corporate network to the extranet application gateway. Because the firewall is stateful, only established connections are allowed back into the corporate network. This firewall does not need routes to the remote networks, because communication between the hub and remote locations happens through the application gateway. Therefore, the PIX can have a relatively simple routing table and access lists. The PIX only has routes to the internal corporate networks. The application gateway is directly attached to the same subnet as the external interface on the PIX, therefore the PIX firewall does not need a route defined for this, and most importantly, the PIX does not have routes to any networks on the extranet.

The outside (extranet) interface of the PIX is assigned a security of 0, which means traffic that originates from the outside of the firewall will not be allowed to the inside of the firewall. The internal interface of the PIX has a security of 100, which means by default anything on the inside can originate a connection to anything on the outside. In our case, we want to limit outbound connections to only the application gateway's address, and only to TCP port 1080 for Socks. We can achieve this with the following access list on the internal interface:

```
access-list inside_out permit tcp any 2.2.2.10 eq 1080
access-list inside_out deny ip any any
```

access-group inside_out in interface inside

**Application Gateway**

The application proxy gateway is the key piece of equipment that provides authentication and authorization for extranet access. Using a Socks-based proxy application, the gateway authenticates and authorizes all connections from the corporate network out to customer systems. The gateway checks the user's login credentials against a RADIUS server, which relays the authentication request to a corporate Windows NT domain controller. This allows the user to use the same login ID and password to authenticate for the extranet, as they use to log into the corporate domain.

The application gateway was placed outside of the PIX firewall (described above). This was done for a couple reasons. First, the Cisco PIX firewall is owned and maintained by a different group within the corporation, which means someone else would need to be involved in the extranet security decisions. Second, if the application gateway were placed on a DMZ interface of the firewall, the firewall would then need routes to all the business partner and client networks. With the application gateway located outside of the firewall, the firewall only needs routes to the internal networks and to the proxy server.

This required that the operating system on the proxy server be hardened and that the proxy server had some type of firewall software to protect it. The next few sections will discuss how the OS was hardened, what was done to protect it from external connections, and how the proxy software was configured to provide authentication, authorization and accountability.

**Installing Red Hat Linux 7.2**

The server used for the application gateway is an Intel PIII 800MHz processor with 512 MB RAM and two, mirrored 9.1 GB hard drives. This hardware configuration is based on recommendations from the proxy software vendor. For the Red Hat Linux 7.2 install, a "Server" installation was chosen. After the initial setup was started, the following partitions and options were configured.

**Partitioning:**

After specifying the installation class, you are asked to partition the space on the hard drives. Since the hard drives on this server are being mirrored with the hardware RAID controller (RAID 0), Linux will see it as one 9.1 GB drive.

Below is how the hard drives were partitioned:

| | |
|---|---|
| / | 1 GB |
| /chroot | 256 MB |
| /tmp | 256 MB |
| /var | 5 GB |
| <swap> | 512 MB |
| /home | 256 MB |
| /usr | 1.4 GB |

Most of the space was assigned to /var, which is where the log files are maintained. In a large environment, with detailed logging enabled, the log partition should be the largest partition. The remaining partitions will typically grow only slightly over time.

**Firewall Option:**

When prompted to enter the security level of the firewall, select "Medium" and then choose to allow only incoming SSH traffic. We will customize the Iptables file later to allow additional services.

**Pre-installation Configuration:**

The "Server" installation will by default install many services that aren't needed. These options should be unchecked during the install, so as to minimize the risk of someone being able to hack into a running service, or an open vulnerability in a piece of software. To remove these services, select the option "Select individual packages" and *uncheck* the items below:

| | | | |
|---|---|---|---|
| X-Window | GNOME | KDE | finger |
| lynx | whois | ncftp | rsh |
| rsync | talk | telnet | ghostscript |
| ghostscript-fonts | groff-perl | Mpage | pnm2ppa |
| Arpwatch | Procinfo | Rdate | Rdist |
| Screen | Ucd-snmp-utils | Indexhtml | Chkfontpath |
| Yp-tools | Lprng | Xfree86-xfs | Finger-server |
| Nfs-utils | Pidentd | Portmap | Ppp |
| Printconf | Rsh-server | Rusers | Rp-pppoe |
| Rusers-server | Rwall-server | Rwho | Talk-server |
| telnet-server | Tftp-server | Ucd-snmp | Wvdial |
| Ypbind | Ypserv | Xfree86-libs | Libjpeg |
| Libpng | Libtiff | Libjpeg-devel | Libtiff-devel |
| Libpng-devel | Gd-devel | Vflib2 | Gd |
| Wu-ftpd | Xfree86-75dpi-fonts | Urw-fonts | Xtt-fonts |

**Post-installation configuration:**

Even though we unchecked a lot of unneeded services in the step above, there are still a few services we can remove that were installed as part of the base "Server" installation. These options cannot be removed during the pre-installation phase. Therefore, they must be removed after the installation is complete by using the RPM (Red Hat Package Manager) command. Use the following syntax to remove packages with the RPM command:

rpm -e *packagename*

To view a list of all packages currently installed on the system, type **rpm -qa**. Below are the packages that aren't needed by the proxy server and can be removed using the **rpm -e** command:

| | | | |
|---|---|---|---|
| Pump | Metamail | Apmd* | Linuxconf |
| Setserial | Kudzu* | Raidtools | Redhat-logos |
| Redhat-release | Pciutils | rmt | |

***Note**: Since programs like apmd and kudzu run as a process, it is better stop those processes before uninstalling them by using the following commands:

/etc/rc.d/init.d/apmd stop
/etc/rc.d/init.d/kudzu stop

Many of the items removed in this section and the previous section were removed based on suggestions in the book "Securing and Optimizing Linux: RedHat Edition". For additional Red Hat Linux hardening procedures described in this book, please refer to the following web page:

http://www.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/index.html

**Red Hat Patches:**

To make sure the server is up-to-date with the latest patches and security fixes, you should reference the following web page and download and install any pertinent updates.

http://www.redhat.com/apps/support/errata/ (for all Red Hat versions)
http://rhn.redhat.com/errata/rh72-errata.html (updates specific to version 7.2)

**Configuring Iptables:**

Iptables is similar to the popular firewall for Linux called ipchains. The main difference is that iptables will only check traffic against one chain rather than multiple chains. For example, with ipchains a packet coming into the system and being forwarded out to another system would be processed by the INPUT, FORWARD and OUTPUT chains. With iptables, the INPUT chain is the only chain that would process this packet. Good references for iptables can be found at the following locations:

http://www.linuxnewbie.org/nhf/intel/security/iptables_basics.html
http://www.redhat.com/docs/manuals/RHL-7.2-Manual/ref-guide/ch-iptables.html

Since the proxy server only needs to allow SSH traffic inbound for management, and Socks traffic (TCP port 1080) inbound for application connectivity to extranet sites, the iptables INPUT chain can be very restrictive. The OUTPUT chains need to be more open, because there are various applications being used by people on the corporate network that use various ports to communicate with hosts systems at the customer sites. Therefore, we modified the iptables file, located in /etc/sysconfig, to look like this (comments are preceded by a # sign):

```
# By default, drop any traffic that's not specifically allowed
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT DROP [0:0]
# Allow return packets from established outbound connections
[0:0] -A INPUT -m state --state ESTABLISHED -j ACCEPT
# Allow inbound Socks connections on TCP port 1080
[0:0] -A INPUT -p tcp -m tcp --dport 1080 -j ACCEPT
# Allow UDP 32,000 to 35,000 packets (required for Socks software)
[0:0] -A INPUT -p udp -m udp --dport 32000:35000 -j ACCEPT
# Accept traffic within the proxy server on the loopback interface
[0:0] -A INPUT -i lo -j ACCEPT
# Accept ICMP (ping) packets, used to test connectivity to proxy server
[0:0] -A INPUT -p icmp -j ACCEPT
# Allow SSH connections from management PCs
[0:0] -A INPUT -s 10.10.10.52/255.255.255.252 -p tcp -m tcp --dport 22 -j ACCEPT
# Allow all outbound connections
[0:0] -A OUTPUT -o lo -j ACCEPT
[0:0] -A OUTPUT -o eth0 -j ACCEPT
```

After making modifications to the file, you must restart the daemon, causing it to re-read the iptables file, by issuing the following command:

```
/sbin/service iptables restart
```

**Socks-based Application Proxy Software**

Finally getting down to the heart of what the application gateway is designed to do, which is to provide "accountability" for our corporate connections out to extranet connected systems. User accountability is the ability to bind critical data functions to a single user. It holds users responsible for their actions (Tipton, p. 106). This was achieved by implementing a Socks-based Proxy server as a "gateway" between internal users and external systems.

For our solution, we used Permeo Technologies' (formerly NEC) e-Border Server. The e-Border Server is a Socks5 compliant application proxy server that implements additional security features not found in a typical Socks proxy server. Netscape's iPlanet Proxy Server was also evaluated during the initial testing. But their product only offered LDAP as an external authentication database. At the time, there were already several Radius servers within our environment, and we didn't want to implement another technology just for authentication at the application gateway. Permeo was also willing to contract with us on customizing their software with features needed specifically for our environment, including adding an Access Description prompt (discussed below) to the authentication sequence. For additional information on Permeo's e-Border Server, including information on how to download an evaluation copy with documentation, please refer to the following web page:

http://www.permeo.com/products/eborder.htm

The e-Border Client, installed on the corporate PC, communicates with the server using the Socks protocol on TCP port 1080. The server then makes a connection out to the requested customer system on the extranet, using the protocol originally requested by the PC. This may be anything from a Web browser connection, to pcAnywhere, Citrix, SQL or even a custom application. As long as it is IP based, the Socks proxy software can relay the connection on behalf of the client.

From a security standpoint, this means the PC doesn't need routes or access lists in the firewall to allow it to talk directly to the customer system. From the customer's perspective, it means the customer's system doesn't need routes or permissions in place to communicate back to the PC originating the connection. The customer only needs to be concerned with allowing connections from the proxy server.

The proxy server also provides accountability by authenticating and authorizing users, prompting the user to enter a description of their access (i.e., Upgrading customer system), logging the connection session and providing this information to the reporting system which automatically creates and e-mails reports to customers on a daily basis. The paragraphs below will cover these topics in greater detail.

**RADIUS vs. NT Domain Authentication**

Synchronization of user lists on access servers and home gateways is one of the harder problems of access for extranets, because user lists on network access servers and home gateways tend to drift apart with the normal changes in staffing and in enterprise organization (Covill, p. 72). The proxy server in our environment solves this problem by authenticating the user to the corporate Windows NT domain.

The e-Border server has the ability to authenticate directly to a Windows NT domain controller, if the e-Border server software is running on an NT server acting as a domain controller. But because the application gateway is a Linux server, and not an NT domain controller, the authentication request must be routed to a RADIUS server that's running on an internal Windows NT domain controller.

The idea of running the application gateway directly on a Windows NT domain controller was considered during the initial planning phase. This would have allowed the application gateway to directly handle the authentication requests without the need of an intermediary RADIUS server. But after working through the design plan, we realized there were several problems with this.

The most significant issue was that because the application gateway resides outside of the corporate firewall, and would contain a copy of the corporate NT user database, it was even more critical that the server be secure. While there are many good resources available on how to harden an NT server, options for firewalling the server itself, like would be done with iptables on a Linux server, were sparse and typically costly. One option was to run a stateful firewall like Check Point on the application gateway, but that would have added another technology to the solution and would have increased the cost of the project. The bottom line was that we weren't willing to take the chance of someone being able to gain access to the corporate user database because the security level of the server was questionable.

In addition to the problem of securing the server, there was also the issue having to open up the restrictions on the corporate firewall to allow the domain controller to communicate and replicate with the internal domain controllers. This means the secure and relatively simple access list on the external interface of the corporate firewall would need to become more complex and less secure.

There was also the concern of the Windows NT domain controller broadcasting domain information across the extranet. This probably could have been addressed with additional packet filters on the extranet router, but as with the stateful firewall, would have added additional complexity to the environment.

Going with the Linux server and RADIUS combination, we were not only able to firewall the server with iptables, but we were also able to keep the corporate NT domain user database internal, behind the corporate firewall. This solution simplified the access lists on the corporate firewall and the extranet router, and meant that the application gateway would not contain any corporate usernames or passwords that could have been compromised.

**Authentication**

When a corporate user attempts to connect to an extranet resource, the e-Border client on their PC prompts them for their credentials. The e-Border server receives this information from the client and then forwards it to a RADIUS server that's running on an internal Windows NT domain controller. The RADIUS software on the domain controller submits the authentication request to the domain controller. If the authentication is successful, the domain controller returns this status to the RADIUS software along with any group memberships that have also been defined within RADIUS and NT. The RADIUS software then sends this information back to the e-Border server, which uses the information to perform the authorization described below.

Also aiding in maintaining user accountability is a feature of the e-Border client that requires the user to type in a description of their access, or a "reason", for connecting to the customer's system. This "reason" is captured in the server log file and integrated into the Customer Access reports, which will be described in more detail below.

**Authorization**

If the user's authentication attempt is successful, the RADIUS server returns a Filter-ID (IETF RADIUS attribute 011) which is equal to the Windows NT group the user is a member of. This Filter-ID is used in the proxy access lists (located in the e-Border server.conf file) to determine which destinations the user is authorized to access. For example, the following permit statement would allow a user with an IP address of 10.10.20.200, who is a member of the "support" group, to access systems at Customer X, who has a network range of 4.1.1.0 (with a Class C subnet mask), after they have authenticated via RADIUS to the corporate domain:

permit r - 10.10.20. 4.1.1. - - support@corpdomain #Radius access for support group

The text after the # sign is simply a comment that is saved to the log file. It can be used as an administrative note, or integrated into the reports described in the following section. The designation "@corpdomain" tells the proxy server which RADIUS server it should send the authentication attempt to. The proxy server uses a file called radius.conf to

define the relationship between the "domain" the user enters at the authentication prompt, and the RADIUS server that handles authentication for this "domain". For this particular scenario, the following configuration is used:

corpdomain 10.10.1.180 1645 - - password

Which means if a user types in the domain "corpdomain", then send the request to the RADIUS server 10.10.1.180 or port 1645 (standard RADIUS port) with a RADIUS shared secret of "password". The shared secret is simply used to allow the proxy server to authenticate to the RADIUS server, and has nothing to do with the password the user types in for their credentials.

The functionality provided by the radius.conf file will allow us to expand the application gateway to be used as a tool by business partners that need access to specific customer systems. This will keep the application gateway within our security infrastructure, while utilizing the existing authentication database of the business partner. As an example, the following configuration line would redirect authentication request for Partner A to the RADIUS server at their location:

partnerdomain 5.5.5.100 1645 - - partnerpassword

Couple the line above with the following configuration line in the server.conf file, and the server is able to authenticate and authorize access from a specific source (5.5.5.20) to a particular customer system (4.1.1.5):

permit r - 5.5.5.20. 4.1.1.5 - - group@partnerdomain #Radius access for Partner A

**Logging and Reporting**

Accountability wouldn't be possible if the connection and user information gathered weren't saved and made available for future reference. The proxy server saves this information to a central log file on the server. A typical log entry looks like this:

User=jsmith@corpdomain, Auth=171, Time=05/22/2002 10:53:54, Duration=06:22:41, Transfer=1141626, Source=10.10.20.200:1173, Destination=4.1.1.21:1433, Connection=TCP Proxy, ACL=32 [Radius access for support group], Reason=SQL Database query

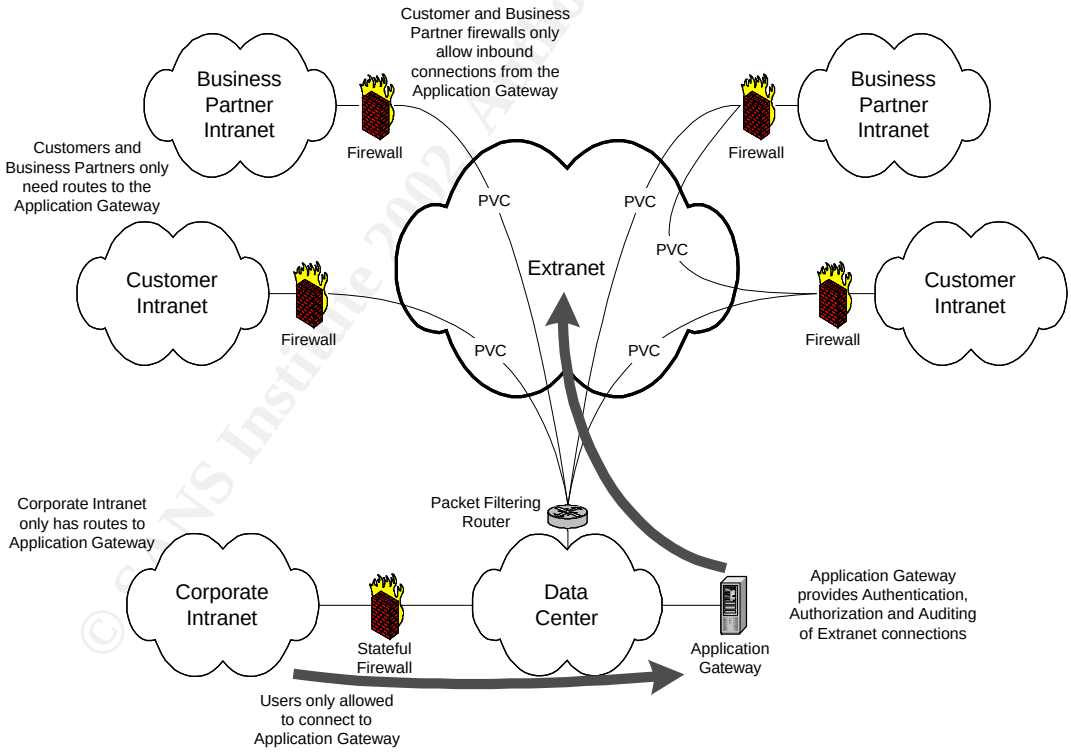
This information is used by our custom reporting system to create "Customer Access" reports that are automatically mailed to customers on a daily basis. The design and functionality of the reporting system will not be covered in great detail in this paper. The design is basically a combination of Perl scripts that collect log data from several systems including the application gateway, parse the data for the respective customer network

range(s) and then extract that information into a Customer Access report that is sent to the customer.

The report documents who accessed their systems, which systems they accesses, the time and duration of the access, protocols used, bytes of data transferred and the "reason" the system was accessed. This completes the accountability cycle because the customer is only a phone call or e-mail away from asking "Why did jsmith access ServerA yesterday at 10:02 AM via Telnet with a reason of 'Because I feel like it'". The user jsmith can quickly be contacted and asked why he "felt like" accessing ServerA at Customer X yesterday.

## The Extranet Today

With all of the above solutions in place, the extranet has become a much more secure environment for performing critical and confidential business functions. Comparing the current diagram below with the original diagram of the extranet before the changes, you can see security issues were addressed and resolved at all access points of the extranet.




Both customers and business partners feel secure in being part of the extranet. Their firewalls can be secured to only allow access from the application gateway, instead of our

entire corporate intranet. In addition, the routers connecting them to the extranet are configured with routes and access lists that only allow communications from the application gateway and any other customers or business partners they've requested access to.

The corporate intranet is also more secure because of the routing changes that were put in place. Customers and business partners are not able to route directly to our intranet, and corporate users are not able to directly connect out to customer systems. The Cisco PIX firewall also provides an added layer of security. In the unlikely event someone is able to bypass the routing and access list controls, the filters on the PIX will block all traffic that didn't originate from the internal network. It also prohibits the application gateway from originating a connection into the corporate intranet.

The application gateway provides many important security features we didn't have before. Probably the most important are authentication, authorization, auditing and accountability. With the authentication and authorization tied into the corporate Windows NT domain, users don't have to remember a separate password to access an extranet resource, plus the user is required to change his/her password on a regular basis as set forth by the corporate policy. And because the gateway is accessing the corporate Windows NT domain via Radius, we don't have to be concerned with exposing a corporate Windows NT domain controller to the extranet.

The gateway also provides detailed logs that are used to create the customer access reports that are automatically e-mailed to the customers. The Red Hat Linux operating system on the gateway has been hardened by removing unnecessary services and secured using iptables. The iptables rule set will only allow Socks proxy connections from the corporate intranet, SSH connections from our management PCs, and any established TCP connections; all other traffic will be dropped. Finally, the application gateway provides a central connection point, which makes securing the corporate intranet, customer networks and business partner networks much easier.

**Conclusion**

An important key in the success factor of this extranet, is making customers and business partners feel secure in being part of the network. They must be comfortable with the level of security implemented throughout the extranet. If not, they will find other methods of communicating with their business partners.

In the case scenario presented here, multiple steps were taken to secure all access points to the extranet. At the network layer, we now have security measures that not only protect the corporate intranet from the extranet, but that also protect customers and business partners from each other.

In addition, there is now a stateful firewall between the corporate network and the extranet, which also protects internal, corporate resources from the extranet. This has made our own corporation feel more secure as being the hub for the extranet.

Finally, the application level proxy server is acting as a gateway between corporate users and resources on the extranet. The operating system on the gateway is hardened to minimize the risk of someone attacking an open service and it is also running iptables to limit inbound access. For corporate connections to extranet resources, it implements security features that provide authentication, authorization, auditing and user accountability.

Because of these measures, the extranet is continuing to grow. New customers and business partners are coming online, while existing customers are expanding their services by using the extranet to connect to other business partners. The security infrastructure has also brought us in-line with federal regulations for the industry, while still allowing for future expansion and introduction of new services to the extranet.

**References**

Covill, Randal Jorde. Implementing Extranets: The Internet as a Virtual Private Network. Boston, MA: Digital Press, 1998.

Loshin, Peter. Extranet Design and Implementation. Alameda, CA: Sybex, 1997.

Tipton, Harold F. and Krause, Micki. Information Security Management Handbook. Boca Raton, FL: Auerbach Publications, 2000.

Red Hat, Inc. "Red Hat Linux 7.2: The Official Red Hat Linux Reference Guide." Chapter 18: Firewalling with iptables. 2001. URL: http://www.redhat.com/docs/manuals/linux/RHL-7.2-Manual/ref-guide/ch-iptables.html (22 May 2002).

Kenshi, Prince. "Iptables Basics NHF." URL: http://www.linuxnewbie.org/nhf/intel/security/iptables_basics.html (15 May 2002).

Mourani, Gerhard. "Securing and Optimizing Linux: RedHat Edition - A Hands on Guide." 2000. URL: http://www.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/index.html (3 May 2002).

Last Updated: August 20th, 2017

# Upcoming SANS Training

**Click Here for a full list of all Upcoming SANS Events by Location**

| | | | |
|---|---|---|---|
| SANS Chicago 2017 | Chicago, ILUS | Aug 21, 2017 - Aug 26, 2017 | Live Event |
| SANS Virginia Beach 2017 | Virginia Beach, VAUS | Aug 21, 2017 - Sep 01, 2017 | Live Event |
| SANS San Francisco Fall 2017 | San Francisco, CAUS | Sep 05, 2017 - Sep 10, 2017 | Live Event |
| SANS Tampa - Clearwater 2017 | Clearwater, FLUS | Sep 05, 2017 - Sep 10, 2017 | Live Event |
| SANS Network Security 2017 | Las Vegas, NVUS | Sep 10, 2017 - Sep 17, 2017 | Live Event |
| SANS Dublin 2017 | Dublin, IE | Sep 11, 2017 - Sep 16, 2017 | Live Event |
| SANS Baltimore Fall 2017 | Baltimore, MDUS | Sep 25, 2017 - Sep 30, 2017 | Live Event |
| Data Breach Summit & Training | Chicago, ILUS | Sep 25, 2017 - Oct 02, 2017 | Live Event |
| SANS Copenhagen 2017 | Copenhagen, DK | Sep 25, 2017 - Sep 30, 2017 | Live Event |
| SANS London September 2017 | London, GB | Sep 25, 2017 - Sep 30, 2017 | Live Event |
| Rocky Mountain Fall 2017 | Denver, COUS | Sep 25, 2017 - Sep 30, 2017 | Live Event |
| SANS SEC504 at Cyber Security Week 2017 | The Hague, NL | Sep 25, 2017 - Sep 30, 2017 | Live Event |
| SANS DFIR Prague 2017 | Prague, CZ | Oct 02, 2017 - Oct 08, 2017 | Live Event |
| SANS Oslo Autumn 2017 | Oslo, NO | Oct 02, 2017 - Oct 07, 2017 | Live Event |
| SANS October Singapore 2017 | Singapore, SG | Oct 09, 2017 - Oct 28, 2017 | Live Event |
| SANS AUD507 (GSNA) @ Canberra 2017 | Canberra, AU | Oct 09, 2017 - Oct 14, 2017 | Live Event |
| SANS Phoenix-Mesa 2017 | Mesa, AZUS | Oct 09, 2017 - Oct 14, 2017 | Live Event |
| Secure DevOps Summit & Training | Denver, COUS | Oct 10, 2017 - Oct 17, 2017 | Live Event |
| SANS Tysons Corner Fall 2017 | McLean, VAUS | Oct 14, 2017 - Oct 21, 2017 | Live Event |
| SANS Brussels Autumn 2017 | Brussels, BE | Oct 16, 2017 - Oct 21, 2017 | Live Event |
| SANS Tokyo Autumn 2017 | Tokyo, JP | Oct 16, 2017 - Oct 28, 2017 | Live Event |
| SANS Berlin 2017 | Berlin, DE | Oct 23, 2017 - Oct 28, 2017 | Live Event |
| SANS Seattle 2017 | Seattle, WAUS | Oct 30, 2017 - Nov 04, 2017 | Live Event |
| SANS San Diego 2017 | San Diego, CAUS | Oct 30, 2017 - Nov 04, 2017 | Live Event |
| SANS Gulf Region 2017 | Dubai, AE | Nov 04, 2017 - Nov 16, 2017 | Live Event |
| SANS Miami 2017 | Miami, FLUS | Nov 06, 2017 - Nov 11, 2017 | Live Event |
| SANS Amsterdam 2017 | Amsterdam, NL | Nov 06, 2017 - Nov 11, 2017 | Live Event |
| SANS Milan November 2017 | Milan, IT | Nov 06, 2017 - Nov 11, 2017 | Live Event |
| SANS Sydney 2017 | Sydney, AU | Nov 13, 2017 - Nov 25, 2017 | Live Event |
| Pen Test Hackfest Summit & Training 2017 | Bethesda, MDUS | Nov 13, 2017 - Nov 20, 2017 | Live Event |
| SANS Paris November 2017 | Paris, FR | Nov 13, 2017 - Nov 18, 2017 | Live Event |
| SANS Adelaide 2017 | OnlineAU | Aug 21, 2017 - Aug 26, 2017 | Live Event |
| SANS OnDemand | Books & MP3s OnlyUS | Anytime | Self Paced |