Application Gateway Firewalls:
How to Implement a Portus v.4.0 Application Gateway Firewall

GIAC Practical v.1.4 (revised June 2002)
Option 1: Research on a Topic in Information Security

Kyle A Smith
Sans ID#: SD595874
June 30, 2002

Table of Contents

Configuring a Portus v4.0 Application Gateway

Introduction/Abstract

A few years ago, the devices installed to provide security for an enterprise consisted primarily of a single firewall. During that time, processor power and throughput were maximized by a single style of firewall, the packet-inspection firewall. While this firewall style is still the most widely used in the market, another type of firewall is now being utilized effectively with little performance impact to an enterprise: the Application Gateway Firewall, more commonly known as the proxy firewall.

This document will give a brief description of the styles of firewalls that are available on the market to provide a comparison for later modeling, and it will be particularly focused on the Application Gateway firewall style. Finally, this paper will describe how the Portus v4.0 Application Gateway Firewall can be configured and deployed to provide an advanced level of protection for an enterprise. A brief description of configuration of the inbound and outbound web based proxies along with the email proxy will be described. The overall proxy management process will be described and shown for only the Telnet proxy.

For the purposes of this paper, the software will be Portus v.4.0 on an IBM RS/6000 running AIX version 4.3.3. Portus v.4.0 is also available on AIX 4.2.2, Red Hat Linux 7.0, 7.1, and 7.2 as well as Solaris 7 and 8.

A quick overview of Firewall Types:

The firewall has become an integral part of any basic network security design. Every design engineer has a decision to make when embarking on a network design in order to increase the security around an enterprise network. The firewall type must fit the enterprise's requirements and deliver the abilities necessary to protect it.
(space here)

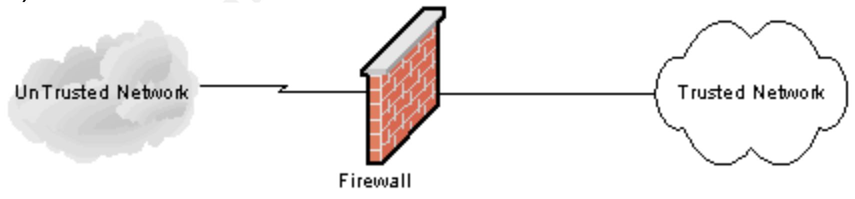

Figure 1: Basic Firewall Design

Packet-Filter Firewall:

The first basic type of firewall is commonly referred to as a packet-filter firewall. These firewalls work on the principal that every packet should adhere to a static rulebase that can be defined by the user. The payload of the packet is not inspected for validity. In most cases, each packet will be treated as an isolated item. These types of firewalls tend to be fairly fast and are transparent to end-users. They are susceptible to upper OSI level attacks and IP spoofing.

Pros: *Speed*. When the firewall doesn't have to recreate a conversation, it can more quickly verify the packets to allow or deny them.

Configuration. Because a large number of packet-filter firewalls are available, the difference between some vendors is how straightforward the rulebase and setup is for the enterprise.
Transparency. Most applications will not be aware of a firewall monitoring the network communications.

Cons: Com*plete Conversation.* Most pure packet-filter firewalls do not inspect the conversation to verify the type of traffic being transported.
Malicious Code. Within the payload of the packets, malicious data can be hidden that will be executed on the target machine.

Circuit-Level firewall:

The second type of firewall is the Circuit-Level firewall. These firewalls allow and deny packets based upon an administrator's defined rulebase. This type of firewall can inspect multiple levels of the packet, and some contain rudimentary packet reconstruction. Many circuit-level firewalls inspect a few packets of the conversation against the rulebase before documenting the 'circuit' or 'state' of the communication and allowing continued the data to pass without inspection. These types are also susceptible to upper OSI level attacks and IP spoofing.

Pros: *Speed*. These firewalls are usually faster than the static packet-filter firewalls because they don't keep re-inspecting the communication streams.
Ease of Setup. The rulebase tends to be as straightforward as your network setup.
Transparency. Most applications will not be aware of a firewall monitoring the network communications.

Cons: *Communication Consistency :* The firewall does not inspect the complete conversation and does not keep a watch on the conversation once it is been 'approved' as valid, so communication session hi-jacking could be an issue.
Standard Port Usage and Data Hiding. The firewall doesn't validate communication occurring across the port.

Application Gateway Firewall:

The third type of firewall is the Application Gateway firewall, or proxy firewall. These firewalls sever any communication between networks and act as an intermediary to deliver packets. Every communication attempting to go *through* the firewall is first relayed *to* the firewall, which will then communicate to the end-station if not rejected. There should be no connection between an un-trusted and a trusted network for a secure network. The application gateway firewall is usually aware of certain protocols and inspects every level of the OSI layer to determine the validity of the traffic. These are not susceptible to application level OSI attacks for well-defined protocols and applications.

Pros: *Layered Security.* Complete Packet Inspection across all OSI layers.

Communication Segmentation. Communication is only allowed when it is passed to a gateway. No communication is allowed from an un-trusted network zone to a trusted network without using a gateway.

Cons: *Speed.* The complete network layer inspection requires significant processor power to complete inspections.
Gateway Availability. For the most part, gateways are only provided for standard and well-used ports. Specialized applications might not have a defined proxy available for implementation.
Application Issues. Applications might not be designed to use a proxy.
Configuration Complexity. The proxy setup can be complicated.

Hybrid Firewalls:

The majority of firewalls on the market today should be more commonly known as Hybrid firewalls. These firewalls are combinations of packet-level and circuit-level firewalls, along with some application gateway firewall abilities for well-known applications. For example, the Checkpoint firewall is most commonly billed as a stateful packet-inspection firewall, while it also contains most of the requirements for a circuit-level firewall and some proxy abilities for well-defined applications. This combination of technologies allows a firewall to be a portion of the best protection an enterprise can have installed while still meeting the basic requirements for an enterprise.

Stateful Packet-Inspection Firewall:

An enhancement joining the standard packet-inspection and circuit-level firewall is the stateful packet-inspection firewall. Internet.com's Webopedia defines it as:

Stateful inspection is a firewall architecture that works at the network layer. Unlike static packet filtering, which examines a packet based on the information in its header, stateful inspection tracks each connection traversing all interfaces of the firewall and makes sure they are valid. An example of a stateful firewall may examine not just the header information but also the contents of the packet up through the application layer in order to determine more about the packet than just information about its source and destination. A stateful inspection firewall also monitors the state of the connection and compiles the information in a state table. Because of this, filtering decisions are based not only on administrator-defined rules (as in static packet filtering) but also on context that has been established by prior packets that have passed through the firewall.

Source: http://www.webopedia.com/TERM/S/stateful_inspection.html

Portus Application Gateway v4.0

History:

The Portus Application Gateway has been in use since 1994 securing all sizes of enterprises. "Developed originally at IBM Thomas J. Watson Research Centre, PORTUS has been enhanced by both IBM and LSLI over several years"[1]. Portus has been continuously developed and has released up to 10 separate versions enhancing the functionality of the overall product. The Portus Application Gateway is an overlying product that secures the server's operating system in addition to incorporating proxies and other secure processes for the firewall functionality. Portus incorporates proxies for most of the well-known applications available and an overall generic proxy for other accesses.

The Portus design is implemented with proxies and management processes listening on well-defined ports. The management proxy process called *fwcop* manages the standard network services of SMTP, Telnet, and FTP. It serves as a replacement for the standard UNIX services process provided by inetd with increasing security controls Stand-alone proxies are available separately for inbound and outbound port 80 and port 443 traffic. UDP and RPC traffic-types are handle by a separate sub-process called the *rpcproxy*. For RealAudio streaming media applications a stand-alone proxy called *raproxy* exists. The Real-Time streaming proxy, also known as rtsproxy, handles other types of streaming media. If no other proxy is specifically defined for an application, the generic application proxy called *aproxy* can be setup to provide proxy services for the enterprise.

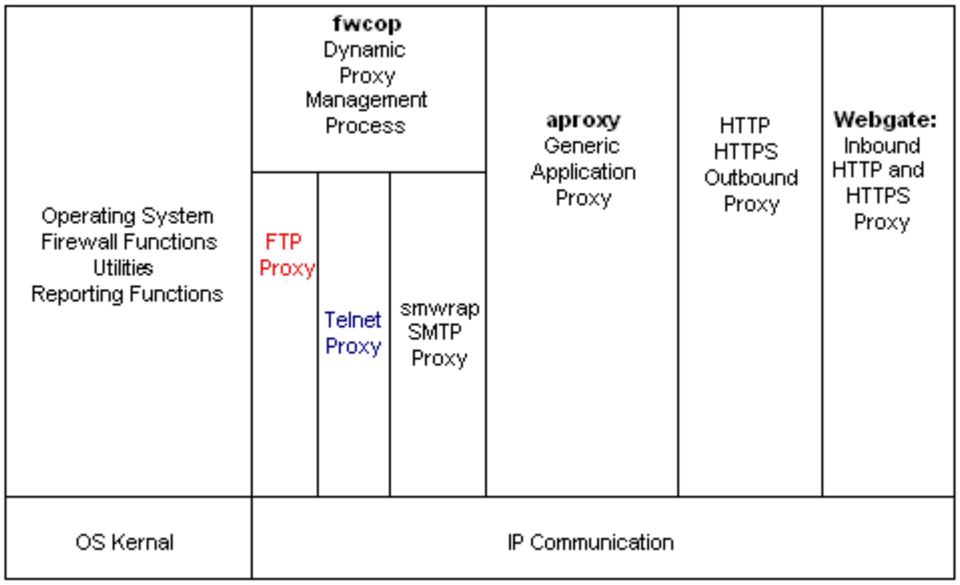

Figure 2: Logical Portus Configuration[2]

[1] "Details of Firewall Products", URL: http://secinf.net/info/fw/steph/sec_jump.html
[2] Portus Administration and Installation Guide

Setup of the System:

The installation of Portus onto a default IBM AIX 4.3.3 installation will harden several of the operating systems' abilities by disabling default processes and installing specially built processes like a re-rooted DNS process. The standard installation of Portus will also disable any IP forwarding built into the system. Certain other files must be configured before the Portus system is usable as a firewall. For this configuration, the basic system will be configured. Proxies for inbound web access, outbound web access, and email will be configured. The proxy management subsystem will also be designed to supervise the Telnet proxy.

Configure the local hosts file:

The fully qualified hostname and the IP address of the internal network interface must be entered correctly. If the Portus system is going to be a SMTP proxy, each internal mail server should be entered into the file. Only standard UNIX formatting for the /etc/hosts is required.

File: ***/etc/hosts***

```
127.0.0.1        loopback             localhost
333.444.555.2    hello1.sample.net    hello1   # Outside Address
10.100.10.2      hello1.sample.net             # Inside Address
10.100.10.100    mail.sample.net               # Inside Mail server
192.168.10.2     hello1.sample.net             # Inside 3rd Network Interface
```

Configure the server for internal name resolution:

The internal DNS server should be entered for the Portus proxies. The Portus system proxies will use this sever to resolve hostnames on the internal interface. The file should be in the standard UNIX formatting of an /etc/resolv.conf file.

File: ***/etc/firewall/resolv.inside.conf***

```
domain sample.net
nameserver      10.100.10.240
```

Configure the systems base DNS system:

Verify that the named file for DNS resolution matches the information entered previously for hostname and IP address.

File: ***/etc/named.local***

```
0.0.127 in-addr.arpa.   IN  SOA         hello1.sample.net.      root.hello1.sample.net. (
        96091401        ; serial
        36000           ; refresh every 10 hours
        3600            ; retry after 1 hour
        3600000         ; expire after 1000 hours
        36000           ; default ttl is 10 hours
        )
        IN  NS          127.0.0.1.
1       IN  PTR         localhost.
```

Configure the protected networks:

Specify the internal networks to be secured by the Portus system. Enter as many internal networks as are defined within your enterprise networks. Hosts within the secure networks are given transparent access to both the telnet and FTP proxies.

File: ***/etc/firewall/securenets***
```
10.100.
10.200.
192.168.
172.16.
```

Configure the protected local network addresses:

Specify the IP addresses to which the Portus system will listen for network traffic. The network interface itself should be defined in this file. Other IP addresses that are implemented as aliases for the outside network interface should also be defined in this file. The defined IP addresses are used to check for IP spoofing through the spoofmon security process. Spoofmon is an automatic process built into the Portus security monitoring subsystem. Spoofmon monitors the network traffic and watches for IP addresses on interfaces that should not be receiving traffic particular traffic. Protected network traffic on the unprotected interface will generate an alert and message for the system administrator as it could be the work of a hacker.

File: ***/etc/firewall/secureports***
```
10.100.10.2
10.100.50.20
10.100.50.30
10.100.50.40
10.100.50.50
```

Configure the outside interfaces addresses:

Each proxy can be configured to listen across as many interfaces as available on the system. On the AIX system, configure the interface aliases by editing the ***/etc/rc.net*** file.

File: ***/etc/rc.net***
```
ifconfig en0 333.444.555.10 alias
ifconfig en0 333.444.555.20 alias
```

Proxy Configuration:

Firewall Cop proxy:

FWCop is the firewall traffic process that regulates several of the underlying sub-proxies. FWCop supervises the FTP, telnet, SMTP, and Real-Audio sub-proxies. Prior to version 4.0, each of the sub-process were configured as standalone processes. With the introduction of FWCop, each sub-proxy is regulated and only available through a specialized configuration. FWCop serves as a replacement for the inetd process.

Since fwcop regulates the amount of connections to a proxy, limitations can be set through the configuration. A properly segmented network can be defined within the maxconnections parameter to allow outbound connections. Using fwcop, the regulated proxies can be defined to listen only on particular defined IP address associated with the Portus system.

File: ***/etc/firewall/fwcop.conf***

```
# Telnet Proxy Configuration Lines
                proxy {
                        maxprocs 999
                        path /usr/local/etc/tnproxy tnproxy
                        listen 23
                        listen 10.100.10.2:23
                        maxconn 10.0.0.0/255.0.0.0 10 15
                        maxconn 0.0.0.0/0 1 2
                }
```

Configuration Parameters:

Path <full-qualified path>

Define here the path to the executable for the proxy. At least one parameter, usually the proxy executable itself, must be defined.

Maxprocs <number>

The maximum number of process fwcop will spawn to handle the requests to this proxy.

Listen <port>

Defines what TCP port the fwcop process will listen for inbound connections across all interfaces setup on the system

Listen <IP Address>:<port>

Define what port and address the fwcop process will listen for connections for access.

maxconn < network/netmask> <max single IP connects> <max connects>

Define the connection configuration fwcop will manage per a network. The fwcop process will regulate the number of connections per IP and the maximum of connections for the proxy in general.

Secure Mail Wrapper Proxy:

The Secure Mail Wrapper (smwrap) process is used to regulate SMTP traffic by the Portus system. The smwrap process consists of several utility programs along with two proxies to deliver mail securely and report on the functions. These files include the smwrap program, smwrapd program, adam program, smrpt program, and the smrptx program.

The smwrap program is the process that received mail from both the secure and un-secure networks. It will only receive the emails by writing them to a standard directory. The files are written as an 'smwrap' file with an extension of a random number. Once the complete email has been received, the Smwrap will check the file against its security configurations to determine the next set. The emails could either be blocked immediately and destroyed or sequestered for further examination by an administrator. The program then renames the remaining files with an upper case 'Smwrap' file name. Smwrapd "wakes up" periodically and checks for available 'Smwrap' files to forward on to the system mail program, like 'sendmail.' This process doesn't allow sendmail to receive emails directly from an outside source thereby avoiding any possible sendmail exploits.

File: ***/etc/firewall/smwrap.conf***

```
Group          = staff
Userid         = hermes
Maxbytes       = 1500000
Maxreceipts    = 100
Maxchildren    = 10
Spoolpath      = /home/hermes
Timeout        = 1200
Undelivpath    = /home/hermes/sequestered
Wakeup         = 60
Smwpdpath      = /usr/local/etc
Smtppath       = /usr/lib/sendmail
####
Deny from @hotmail.com to *
Deny from * to george@sample.net
Denyx from * to * ext = exe sequester
```

Configuration Parameters:

Timeout = <time>
Defines the timeout for smwrap before it determines that the connection is not a complete message.

Userid = <UNIX userID>
Defines the UNIX userID that the smwrap processes should run under.

Group = <UNIX group>
Defines the UNIX group that the smwrap processes should run under.

SpoolPath = <full path to a directory>
The smwrap subprogram will deposit the incoming mail to this directory before processing its rules. The smwrap subprogram will rewrite the mail messages to this directory for smwrapd to forward to the system mail program.

Maxbytes = <count>
Defines the maximum size per individual email that the process will accept. The smwrap subprogram will automatically destroy the messages once they have passed the limit. *No reply is sent to the sending mail system stating that the message was received or rejected.*

Maxchildren = <count>
Defines the maximum number of email addresses on the “To” or “CC” mail fields. The smwrap subprogram will automatically destroy the messages once they have passed the limit. *No reply is sent to the sending mail system stating that the message was received or rejected.*

Smwrpdpath= <full path to the directory housing the subprograms>
The smwrap and smwrapd subprograms are located at this location.

Smtppath = <full path to the system mail program>
Defines the location for the system mailing application. Includes the executable filename within the configuration parameter.

Undelivpath = <full path to sequester mail>
Defines the directory location of the messages to be examined later by an administrator.

Wakeup = <count>

Defines the number of seconds that the smwrapd subprogram will wait before it checks for new messages deposited by the smwrap subprogram.

Deny from <address | * > to <addresss | * >

Defines what mail the smwrap system should deny from the outside or the inside. *If the system is configured to block emails from a particular system, then smwrap will only block mail that is defined fully after the "@" within an email's "From" line. A definition of block *@*.hotmail.com will not block mail from "@mail3.hotmail.com". A generic "*" will block all emails from an address or to an address, depending upon implementation.*

Denyx from <address |*>to <address |*>ext = <extension> <result>

Defines what email addresses can send or receive particular email attachments. The smwrap system can be configured to sequester the emails for an administrator to manually strip the attachments at a later point.

HTTP/HTTPs Proxies:

The number one usage of external network links for most enterprises is web-based access to the overall Internet. Portus contains two different proxies specifically designed for inbound and outbound HTTP/HTTPS connections. The outbound proxy is commonly referred to as the http proxy, while the inbound referred to as the webgate proxy.

File: ***/etc/portus/webgate.conf***

```
Gproxpath       = /home/hermes
Checkprocs      = 5
DNS             = no
Httpdlog        = webgate.access.log
Logformat       = WebTrendsMultiHomed
Maxprocs        = 600
Nprocs          = 30 30 155
Maxuse          = 600
Timeout         = 1800
###
deny from * to 333.444.555.1 80
permit from * to 333.444.555.10 80 redirect 10.100.45.10 80
permit from * to 333.444.555.10 443 redirect 10.100.45.10 443 VLOG
permit from 777.666.555.444 to 333.444.555.20 80 redirect 10.100.45.20 80
```

Configuration Parameters:

Gproxpath = <path>

Required. Defines the full path that the location webgate will run under.

DNS = <Yes | No>

Enable this value if the webgate proxy is to use DNS lookups when resolving outside and inside IP addresses.

Httpdlog = <filename>

Defines the name of the log file to document transitions. If no value is defined, it will use httpd.access.log as the default.

Checkprocs = <number>

Configures the number of seconds before the parent webgate process checks the number of webgate children processes to determine if more processes are needed to meet demand.

LogFormat = <WebTrendsMultiHomed | CLFExt>

Defines the format of the log file to document the webgate processes activity.

Maxprocs = <number>

Defines the maximum number of webgate parent and children process to spawn.

Maxuse = <number>

Defines the number of times a particular webgate process can be reused before it is reaped and replaced with a new process. The reaping of the process returns all system resources to the system.

Nprocs = <number> <number> <number>

Defines the values for the number of process to start, the minimum number of spare webgate processes to keep alive, and the maximum number of idle process to keep alive. Upon starting the webgate parent process, it will spawn the first value defined.

Timeout = <number>

Defines the amount of time a process should wait for a message before it is considered an idle webgate process.

Deny from <address|*> to <addresss|*> <port|*>

Defines an explicit deny rule for denial of a particular IP address to access an external-to-internal redirect. Denies all IP addresses from using a specific webgate proxy.

Permit from <address|*> to <addresss|*> <port|*> redirect <address1, address2, etc.> <port|*> <VLOG>

Defines an explicit allow rule for redirection to an inside server. Webgate allows a single redirect to be sent to multiple servers in a round-robin fashion, enabling the "VLOG" parameter to document HTTPS connections to the system syslog file for later analysis. Otherwise, the HTTPS connections are not documented to the standard log file.

The outbound HTTP proxy is a modified apache server configured to be a web-proxy. The majority of the configurations of the Apache proxy server are standard secure configurations. Portus includes other abilities that tie into the Apache http daemon. It can also be used as a content filter to define sites an enterprise user can access. Multiple files can be used to define what sites are to be blocked. A specialized webpage can be used to redirect users from filtered sites.

File: ***/etc/apache/httpd.conf***

```
User                hermes
Group               staff
HostnameLookups     Off
MinSpareServers     20
MaxSpareServers     30
StartServers        250
MaxClients          500
```

```
MaxRequestsPerChild   1000
ProxyBlockList  /etc/firewall/lists/hacker  http://www.sample1.net/nono/index.html
ProxyBlockContent       Java
Listen                  10.100.10.1:80
Listen                  192.168.1.1:80
<Directory Proxy>
        allow from 10.100.0.0/255.255.0.0
        allow from 192.168.0.0/255.255.0.0
</Directory Proxy>
```

Configuration Parameters:

User = <userid>

Defines the system userid to execute the outbound http process.

Group = <group id>

Defines the system groupid to execute the outbound http process.

HostnameLookups = <on|off>

Determines if the daemon process is to perform DNS lookups for every query during transition. Disable this for quicker system performance.

MinSpareServers = <count>

Defines the minimum number of spare children processes for the parent daemon process to keep alive awaiting user connections.

MaxSpareServers= <count>

Defines the maximum number of spare children processes for the parent daemon process to keep alive before reaping unused processes and reclaiming system resources.

StartServers= <count>

Defines the number of processes for the parent daemon process to start when it is httpd process is started by the system.

MaxClients= <count>

Defines the maximum number of clients that can use a child process before it is over-used and reclaimed by the parent daemon.

MaxRequestsPerChild= <count>

Defines the number of requests that a child process will serve before the child process is reclaimed by the parent daemon process.

ProxyBlockList <path to file> <message URL>

Documents the location of the file that contains the list of websites to block. The users will be redirected to the site defined in the message URL when a blocked site is encountered.

ProxyBlockContent <content type>

Defines the type of http traffic to block. The proxy can be configured to block Java, Javascript, cookies, or ActiveX components.

<Directory Proxy> allow from <network>/<netmask>

Defines the networks that can access the proxy instances.

Conclusion:

The Portus Application Gateway Firewall is a secure proxy firewall that can be deployed in a relatively short amount of time. Using the configurations described above, a simple network can be protected and given access to outside resources. The majority of the configurations described are standard configurations that can be modified as an enterprise grows. Since the system is highly configurable, if an enterprise does not want the full set of proxy abilities, the system components can be de-activated as necessary.

With advances in system performance and network speed over the past several years, Application Gateway Firewalls have finally begun to match the industry standard packet-filter firewalls in ability. With the increasing amount of exploits and vulnerabilities of higher-level applications, an Application Gateway firewall can be a valuable addition to an enterprises security stance.

Appendix A: References

URL Sources Verified: June 30, 2002

Grennan, M.; "Firewall and Proxy Server HowTo"; Feb. 26, 2000; URL: http://www.tldp.org/HOWTO/Firewall-HOWTO.html

Internet.com Webopedia; "Stateful Inspection"; URL: http://www.webopedia.com/TERM/S/stateful_inspection.html

King, C., Dalton, C, and Osmanoglu, T.; "Security Architecture: Design, Deployment, and Operations"; 2001, Osborne/McGraw-Hill; ISBN 0-07-213385-6

Kovacich, G.; "Information Systems Security Officer's Guide", 1998 Butterworth-Heinemann; ISBN 0-7506-9896-9

Livermore Software Laboratories, Intl.; "Portus-ES Product Brief"; March 12, 2002; URL: http://www.lsli.com/portus_es_product_brief.html

Livermore Software Laboratories, Intl.; "Portus Installation and Administration Guide, version 4.0"; January 2001

National Institute of Standards and Technology, Computer Security Resource Center; "Application Gateways", Feb. 5, 1996; URL: http://csrc.nist.gov/publications/nistpubs/800-10/node52.html

Network Security Library, "Details of Firewall Products", URL: http://secinf.net/info/fw/steph/sec_jump.html

pchelp@pc-help.org; "What is a Firewall?", URL: http://www.nwinternet.com/~pchelp/security/firewalls.htm

Pulsar Systems; "Functions in PORTUS 4.0 from Pulsar Systems"; July 2, 2001; URL: http://www.pulsarsystems.com/portus2.html

Spafford, G. and Garfinkel, S.; "Practical Unix and Internet Security, 2nd Edition"; April 1996; O'Reilly & Associates, Inc.; ISBN 1-56592-148-8

Zwicky, E.D., Cooper, S., and Chapman, D.; "Building Internet Firewalls, 2nd Edition"; June 2000, O'Reilly & Associates, Inc., ISBN 1-56592-871-1