GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Auditing Systems, Applications, and the Cloud (Audit 507)"
at http://www.giac.org/registration/gsna

Testing a SonicWALL Plus DMZ firewall

GSNA Practical v4.0 Option 1, Topic 1 – Testing

Jeff Richeson
March 16, 2005

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

Abstract

Firewalls are used as a front-line defense to protect networked computers from external attack. How does one really know if the firewall is doing the job it is supposed to? It is too late to wait to be attacked to find out that the firewall did not perform as expected. This paper outlines three risks and impacts to a software development company with respect to the SonicWALL Plus DMZ firewall. Detailed test procedures are created to test for the existence of firewall vulnerabilities. Although the test procedures are specific to the SonicWALL firewall, they can easily be adapted for other firewalls. Finally, the test procedures are run and actual output from each test is shown. The results and potential solutions to vulnerabilities are discussed.

© SANS Institute 2000 - 2005, Author retains full rights.

Table of Contents

Abstract ii
Table of Contents iii
Task 1: Identification 1
  Company Overview 1
  Device to be Tested 2
Task 2: Risk Analysis 3
  Risk 1: Misconfigured Ruleset 3
  Risk 2: Firewall Weakness 4
  Risk 3: Denial of Service 6
Task 3: Test Procedures 7
  Configuration 7
  Test 1 8
  Test 2 9
  Test 3 12
Task 4: Audit 15
  Results of Test 1 15
  Results of Test 2 18
  Results of Test 3 20
  Conclusion 21
References 23

© SANS Institute 2000 - 2005, Author retains full rights.

Task 1: Identification

Company Overview

Great Software Development[1] (GSD) employs approximately 40 people in the Houston, Texas area. GSD develops custom software solutions for a variety of clients. Although GSD has some commercial work, the majority of the software development is for government contracts.

GSD does not employ a dedicated security administrator. Two of the software developers employed by GSD have the additional privileges and duties of being the information technology (IT) administrators for the company. These two employees are primarily developers and become "the administrator" when changes are needed or when something is not functioning correctly. The responsibility of the administrators is to ensure that the computing resources are operating properly so that the employees can complete their work.

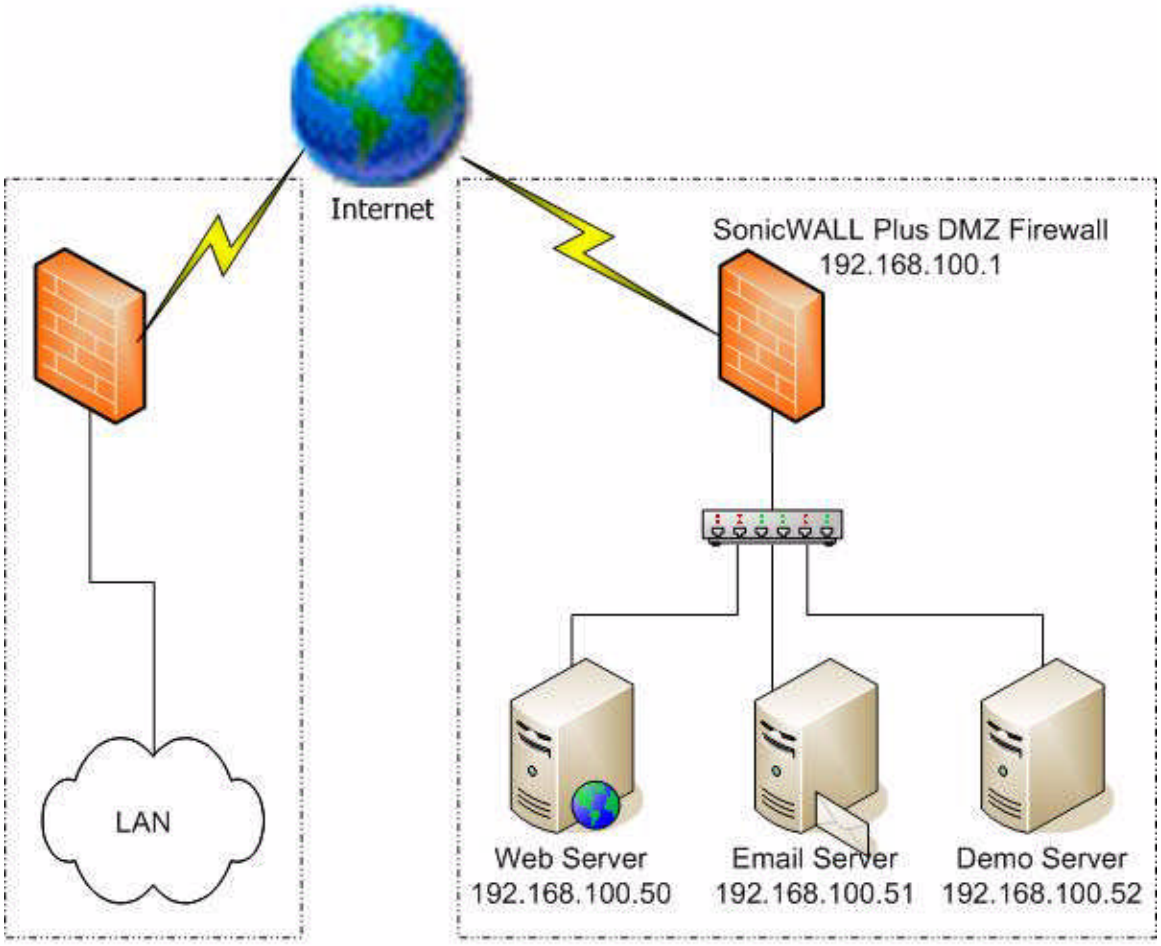


Figure 1: GSD Network Architecture

[1] The company name has been changed for this report.

Figure 1 depicts the overall network architecture for the company. Private IP addresses have been substituted throughout this document to hide the true IP addresses. GSD maintains two Internet connections. One connection is used solely for connecting all the desktop machines in the company LAN to the Internet. There are no externally accessible machines on this connection. The second connection is used exclusively for three publicly accessible servers: a web server, an email server, and a client demonstration server. The web server is used to communicate general information about the company to potential customers. Because software download and support is not offered through the web, the content of the site does not change often. The email server is used for the majority of the communications among employees, clients, and vendors. The other publicly accessible server is a client demonstration computer. This server houses password protected web and ftp sites used to demonstrate software currently in development to the clients.

Device to be Tested

The scope of this audit only includes the SonicWALL Plus DMZ firewall that is protecting the public servers. Vulnerabilities of the servers are not within the scope. However, since the servers are the assets being protected by the firewall, server vulnerabilities are important to the discussion of impacts of the firewall vulnerabilities.

The SonicWALL Plus DMZ (to be referred to as SonicWALL) is a network appliance built in 1999, that functions as a router and stateful packet inspection firewall. It also has capabilities as DHCP and NAT which are not used by GSD. As its name implies, instead of having the standard two network interfaces for the LAN and WAN, a third interface for the DMZ was added for protecting public servers. The only management is through a web-based user interface, which sacrifices some configuration flexibility for the sake of convenience (Jackson). "The major issue with graphic interfaces is configuration granularity. In many modern firewall platforms, there are options available in the firewall that cannot be configured using the graphic interface" (Wack). The SonicWALL has built-in logging capabilities and also provides the ability to log to an external syslog server. The ruleset blocks or allows packets based on network interface, source and destination address, port number, and protocol (tcp, udp, or icmp only). In order to protect itself and the machines connected to it, the SonicWALL can detect and defend against various attacks such as IP spoofing, blind spoofing, syn flood, Ping-of-Death, Land Attack, FTP bound, smurf, and others (Ranum).

GSD uses the SonicWALL as the border router and firewall for the public servers. To reduce the exposure to the Internet, the incoming traffic is filtered to allow only necessary ports for business. All outbound traffic is permitted. Although there is no written firewall policy, an informal policy is followed of only opening ports for which there is a business need. Overall, the firewall is generally ignored until an application does not function as expected. In that case, the firewall logs are reviewed to discover what is being blocked and then

© SANS Institute 2000 - 2005, Author retains full rights.

© SANS Institute 2000 - 2005 Author retains full rights.

the necessary ports will be permitted.

Task 2: Risk Analysis

Risk 1: Misconfigured Ruleset

Explanation

The firewall ruleset determines which packets are allowed or blocked by the firewall. A misconfiguration of the ruleset may accidentally allow packets that should be blocked. Although a misconfiguration could also block packets that should be allowed, it is not generally a security concern. This risk is that a change in the ruleset by the administrator could have unintended security effects. The results of the misconfiguration are that ports that were expected to be blocked by the firewall are now open.

Importance to organization

The misconfiguration of the firewall ruleset could be a contributing factor to a compromise of the servers that are supposed to be protected by the firewall. As with most compromises, there is not a single factor but a set of contributing factors. For GSD, a server compromise could lead to email tampering, webpage defacement, and modification to demonstration software. The potential impact to GSD is loss of integrity of data stored on the public servers. This could lead to loss of revenue for the company since some clients may go elsewhere for software development.

Primary vulnerabilities

According to documentation from the National Institute of Standards and Technology (NIST), a vulnerability is “a flaw or weakness in system security procedures, design, implementation, or internal controls that could be exercised (accidentally triggered or intentionally exploited), resulting in a security breach or a violation of the system's security policy” (Stoneburner). The primary vulnerabilities are: an easy-to-use graphical interface, lack of testing after ruleset updates, lack of a written policy, and lack of firewall configuration management and documentation. The majority of the primary vulnerabilities are not with the firewall itself but rather with the internal controls of how it is used.

The easy-to-use graphical interface allows inexperienced administrators to make incorrect changes. Although the interface is touted as an improvement over the command-line, it has the downside of enabling anyone with privileges to make updates.

When the ruleset is updated to allow additional ports, there is no testing performed to ensure that the change is not overly broad. The testing that is performed only ensures that the requested change is allowed by the firewall. In fact, the entire firewall could be disabled and no user would complain (until the

© SANS Institute 2000 - 2005 Author retains full rights.

servers were compromised.)

The lack of a written firewall policy forces the firewall administrators to use their best judgment when deciding to implement firewall change requests. Some change requests should be denied, however there is not policy to back the decision of the administrators.

The lack of firewall configuration management and documentation leaves the administrators with a lack of understanding of the firewall ruleset. Rules that were supposed to be temporary stay forever. Without a good baseline or configuration management, there is no way of knowing if unauthorized changes were made.

Scenario of exposure and means of exploitation

The vulnerabilities discussed above can lead to a firewall that is not as secure as it is expected to be. Since the firewall is connected to the Internet, it is constantly exposed to a hostile environment. Malicious automated scanners probe the firewall every day looking for open ports. Although open ports in themselves are not bad, it can be a problem if they are open when the company thinks they are closed. This is akin to accidentally leaving a door unlocked when it is expected to be locked. Open ports give dedicated hackers avenues into the machines, which is why the ports should be restricted to those that are necessary.

Risk 2: Firewall Weakness

Explanation

The primary role of the firewall is to block all but specific traffic by properly implementing the ruleset. The firewall vendor may accidentally or purposely leave an undocumented global exception for specific addresses, ports, or protocols. The firewall administrators depend on the firewall operating as expected in a hostile environment. That means it should withstand intentionally malformed packets. These packets are not generated by standard applications but are hand-crafted explicitly not to follow the standard. The risk is that the firewall fails to perform properly by allowing packets that should be blocked or by blocking packets that should be allowed when presented with malicious packets. It is assumed that the firewall performs properly under normal circumstances.

Importance to organization

GSD relies on the firewall to perform as advertised. Flaws in the firewall design can leave especially large vulnerabilities because the expectation of the firewall is that it properly implements the ruleset and does not provide any hidden backdoors. The impact to GSD is the potential loss of confidential information since private company email and unreleased software is stored on computers protected by the firewall. For a server compromise, the firewall would be a

© SANS Institute 2000 - 2005

Author retains full rights.

contributing factor and would require the presence of additional vulnerabilities on the servers.

Primary vulnerabilities

The SonicWALL has no known vulnerabilities that would cause it to improperly block or allow packets. The firewall specifications claim to be able to protect against several well-known attacks (Ranum). In order to properly test, the assumption is made that it has the following vulnerabilities: spoofed address, ruleset implementation deficiencies, and malformed packet traversal. These vulnerabilities are similar to each other in that they all render the firewall ineffective. The vulnerabilities are differentiated by how the exploit is accomplished.

Spoofed or falsified source addresses are used by attackers for a variety of reasons. One objective is to trick the firewall into believing an external packet is really from the internal network. Most firewalls examine both the source address and network interface to determine if a packet claiming to be internal is really coming in from the external interface. The firewall is vulnerable to spoofed addresses if it allows external packets by treating the external packets as internal.

The firewall should properly implement the ruleset allowing only the designated packets, however, the vendor may have placed backdoors for administration and maintenance. The firewall is vulnerable to a ruleset implementation deficiency when standard, well-formed packets are allowed through when they should have been blocked.

The firewall is vulnerable to malformed packet traversal if it allows non-standard packets or it halts processing all packets. Non-standard packets are usually ones that are hand-crafted by attackers and do not conform to standards. The common non-standard packets that are encountered by the firewall have incorrect TCP flags set; these flags are used to maintain the state of the connection.

Scenario of exposure and means of exploitation

Since the firewall is connected to the Internet, it is constantly exposed to a hostile environment. A spoofed address exploit is attempted whenever the source address is modified to match the subnet of an internal host. The Land attack was an IP denial-of-service (DOS) attack discovered in early 1998. This attack caused a vulnerable machine to talk to itself by setting the source and destination address to the same value. The firewall may encounter illegal or unexpected combinations of the TCP flags when attackers are attempting to exploit a vulnerability in the firewall. For example, setting the SYN and ACK flag may trick the firewall into believing this packet is in response to a connection attempt.

© SANS Institute 2000 - 2005, Author retains full rights.

Risk 3: Denial of Service

Explanation

The SonicWALL does stateful packet inspection in order to simplify the ruleset administration. The initial SYN packet is checked against the ruleset and, if allowed, a cache entry is created for the connection. Except for the initial SYN packet, all packets must match to an existing connection in the firewall's connection cache. Although UDP does not use connections, a virtual connection record of the ports and addresses is kept in the cache. The size of the cache is limited and thus can be filled by an excessive number of connections. Once the cache is full, no more connections are allowed until there is cache space available. The risk is that a full firewall cache is effectively a denial of service (DoS) for all new connections coming into the servers.

Importance to organization

GSD uses the email system to communicate to employees, clients, and vendors. The web server is used to provide company information to potential clients. The demonstration computer provides current clients access to software. The impact to GSD of a firewall cache DoS is a loss of availability of all server resources. When the cache is full the firewall continue functioning, but it is not able to process any new connections because it has no more storage space. The interruption in a critical communication service causes delays, wasted time, and loss of revenue to GSD.

Primary vulnerabilities

The primary vulnerability for the SonicWALL firewall that leads to this impact is a limited amount of memory to track the connections in the cache. This vulnerability is not unique to the SonicWALL because any technology that depends on a limited resource is vulnerable to resource depletion.

Scenario of exposure and means of exploitation

Attackers have indirect control over connection cache since the amount of memory used is proportional to the number of connections. By creating many simultaneous connections, an attacker can fill the cache. A mitigating factor is that this cache only tracks actual connections that have been allowed by the ruleset and it does not need to track connection attempts. By filtering packets through the firewall ruleset, the cache is somewhat protected from a deliberate DoS by an external attacker. The cache could more easily be exhausted by an internal machine connecting to an external machine since the current ruleset does not block any internally initiated connections.

GSD has experienced this denial of service at least two times, however, neither time was caused by an intentional DoS. Once it was caused by a worm-infected machine being plugged in to the DMZ network. The second occurrence was caused by a test of a bulk email program. Both of these were caused by

© SANS Institute 2000 - 2005

Author retains full rights.

applications making many outbound connections through the firewall. The vulnerability can be exploited by any application (or group of applications) creating and maintaining connections until the cache is full. Since inbound connections are limited by the firewall, it might not be easy to create the condition from outside the firewall.

Task 3: Test Procedures

Configuration

Each test will require the same basic hardware and software configuration. Two machines are needed to test the firewall: one inside and one outside the firewall. The internal machine is connected to the hub on the DMZ interface of the firewall. This machine is assigned an unused IP address in the same subnet as the servers. The external machine is connected to a hub on the WAN interface of the firewall, although it could be any machine connected to the Internet. Connecting it in this fashion avoids problems with other firewalls and packet

© SANS Institute 2000 - 2005, Author retains full rights.

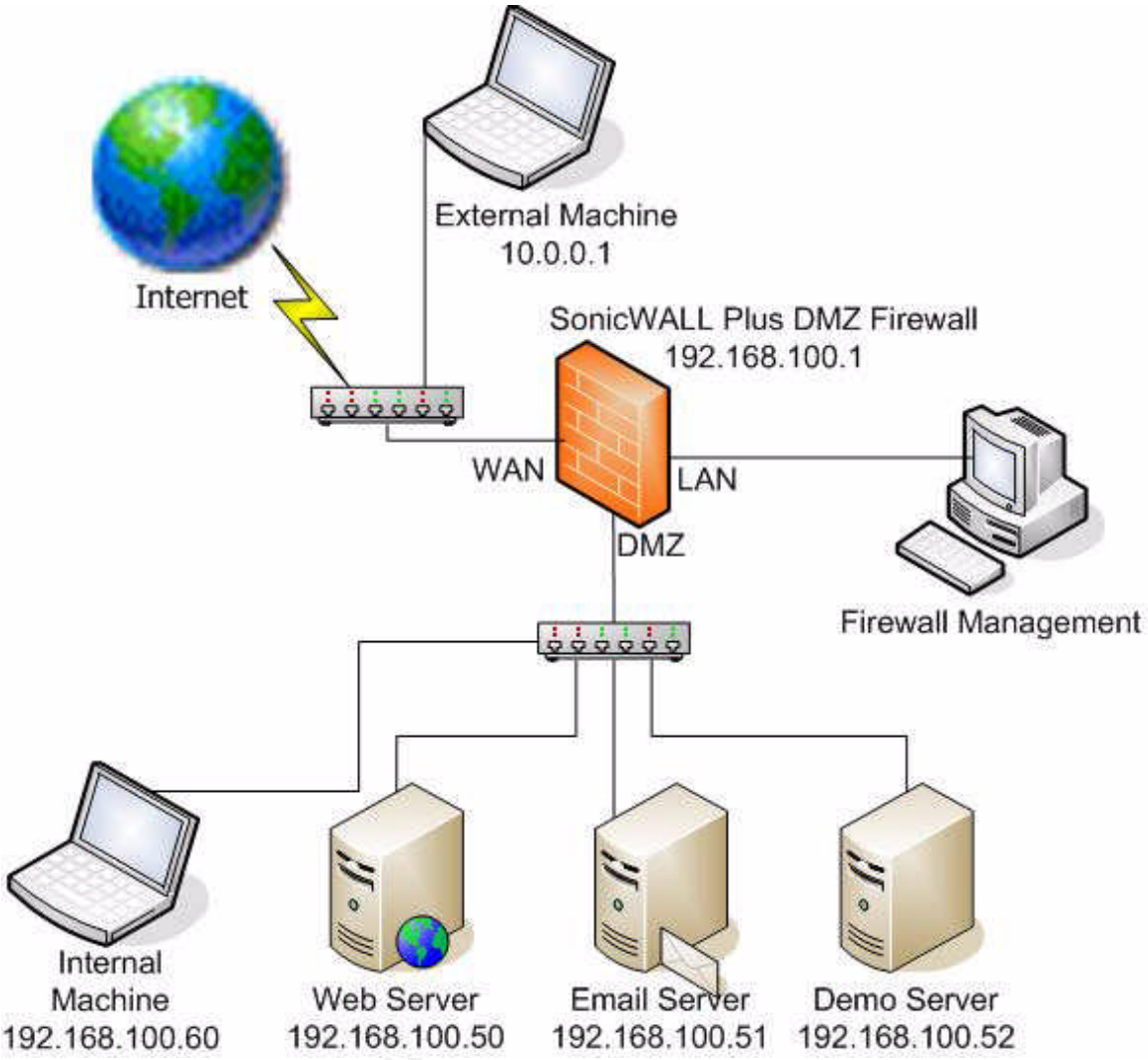


Figure 2: Test Configuration

filters in the route. In addition, having the internal and external machines physically located together facilitates the testing. A firewall management machine is needed to administer the firewall and view the log files. This machine is connected to the LAN interface of the firewall. Figure 2 depicts the overall configuration.

Both the internal and external machines have the same software loaded. The machines are running a customized version of Knoppix 3.7. Knoppix was used because it can be run completely from CD without loading any software on the hard drive. All the commands require root access. For completeness, each software package is listed below with the version number and description.

- Ethereal 0.10.7: a graphical protocol analyzer that provides a real-time display of packets seen on the network interface
- Hping2 2.0.0-rc3: a command-line packet assembler
- Linux 2.4.27: an open-source operating system
- Nmap 3.75: a command-line network scanner

© SANS Institute 2000 - 2005

Author retains full rights.

- Ndiff 0.05beta1: a utility to display differences in two nmap output files
- Perl 5.4.8: a powerful scripting language

Test 1

It is assumed that the firewall is vulnerable to misconfiguration because the administration is done by a human. The following procedures will determine through test and inspection whether the ruleset is configured properly with respect to corporate policy and expectations.

This part of the test procedure attempts to derive a majority of the firewall ruleset from a standard portscan of the servers. The scan inside the firewall is compared to the scan outside the firewall. The differences in the two scans are attributed to the firewall blocking certain ports. The following nmap command is used for the port scanning.

On the internal machine run:

```
nmap –sT –P0 –n –v –m internal –T polite 192.168.100.50-52
```

On the external machine run:

```
nmap –sT –P0 –n –v –m external –T polite 192.168.100.50-52
```

This command does a full TCP connect (`-sT`) scan without first pinging the targets (`-P0`) and without doing name resolution (`-n`). Verbose out will be sent to the screen (`-v`) and a machine-readable file (`-m internal`) will be created with the name of "internal" or "external". The slow scan speed is used (`-T polite`) to help avoid detection by the firewall. Three hosts are scanned: 192.168.100.50, .51, and .52. Although specific ports are not listed in the parameter list, the default behavior is to scan ports 1024 and below as well as ports listed in the "nmap-services" file. The default port listing is sufficient for this test.

Next, the two scans will be compared to find the differences caused by the firewall. The ndiff utility is used to produce the differences between the two files created by nmap. The two files should be on the same machine and the following command will produced the differences.

```
ndiff –b internal –o external –oh c > changes
```

This command take a baseline file (`-b internal`) and shows only the machines that have changes (`-oh c`) with respect to the observed file (`-o external`) and redirects the output to the file named “changes” (`> changes`).

The output from the external nmap scan contains all the scanned ports that are available from outside the firewall. The “changes” file contains all the scanned ports that are opened on the servers but are blocked by the firewall. These two files are the inferred firewall ruleset. The blocked ports may not solely be

© SANS Institute 2000 - 2005, Author retains full rights.

attributed to the ruleset since the firewall may have detected the scan and temporarily blocked all access from the scanning address. Use the firewall logs to determine if an external scan was detected. The built-in firewall logs are viewed through the management computer by using a web browser to login to the firewall. Clicking on the "Log" button along the left side brings up the logs. If the logs show a detected scan then the external scan should be run again with a slower scan speed. The –T parameter in nmap with options of paranoid or sneaky will reduce the scan speed.

Once the scanning is complete, the inferred ruleset needs to be compared to the actual firewall ruleset. The actual firewall ruleset can be obtained by using a web browser on the management computer and clicking on the "Access" button on the left side then selecting the "Rules" tab. Since the SonicWALL does not provide the ability to dump the ruleset to a file, a printout, screen capture, or cut and paste from the web browser will have to suffice. The inferred ruleset will be matched to the actual ruleset. Also, if the rule function is not obvious, each rule on the printout will be annotated with a useful description. Comparing the scan results to the firewall ruleset gives the firewall administrator a fresh look at a ruleset that may have become all too familiar.

After all the entries on the scans have been matched with ruleset entries, the ruleset needs to be reviewed for compliance with the written corporate firewall policy. If the current policy to too vague or no policy exists, then the ruleset should be compared to industry best practices. Basically, the ruleset should only allow in packets for ports that are necessary for business. All others should be blocked. Any discrepancies between the actual ruleset and the corporate policies should be noted in the results for this test.

Test 2

The first test should have demonstrated, in part, that the SonicWALL operates as expected when it receives expected input. Test 2 will determine if the firewall is vulnerable to failing when it receives unexpected input. The firewall will be vulnerable if it allows packets that should be denied or it stops responding to all packets. The intent is not to stress test the firewall, but rather to ensure it is properly hardened to handle malicious data.

To test that the firewall actually blocks malicious packets, the internal computer will be running a protocol analyzer, Ethereal, which will be listening for any packets directed to that computer. There should be no packets detected since the firewall should already be configured to block all access to this "unknown" (to the firewall) computer. The external computer is used to create a variety of malicious packets in an attempt to circumvent the firewall protections.

On the internal computer run:

```
ethereal –i eth0 –k –p –S -l –n –f ”host 192.168.100.60”
```

© SANS Institute 2000 - 2005

Author retains full rights.

`This command starts Ethereal,` a graphical protocol analyzer, immediately capturing (`-k`) packets on the eth0 network interface (`-i eth0`) without using promiscuous mode (`-p`). Packets are displayed as they are seen (`-S`) and the window is automatically scrolled so that the latest packet is always seen (`-l`). Hostname resolution is disabled (`-n`) so that unnecessary DNS traffic is avoided. A capture filter is applied so that only packets going to or coming from the 192.168.100.60 address are captured (`-f "host 192.168.100.60"`). The Ethereal display should be reviewed for activity after each packet is sent from the external machine. Additionally, the firewall logs should be reviewed from the management computer to see what the firewall logged for the packet.

First, the external computer will attempt a stealthy fin scan of the internal computer. In this case, the results of the scan are not as important as the information gathered by Ethereal on the internal computer.

```
nmap –sF -f –P0 –n –v –T polite 192.168.100.60
```

Some of the parameters are not explained since this command is similar to the nmap commands run in Test 1 (see page 8 for the explanation of those parameters). Since the output is not all that important, it is just displayed to the screen and not saved in files. A fin scan is used (`-sF`) to avoid detection by the firewall. Fragmentation is used (`-f`) in an attempt to confuse the firewall. After the scan, the firewall logs should be reviewed to see what was detected by the firewall. Also, the packets shown on Ethereal should be reviewed to see if any reached the internal computer.

Next, the external computer will send a variety of malformed packets to see which, if any, are seen by the internal computer. Hping2 is the tool used to create most of the malformed packets. Hping2 allows most aspects of a packet to be modified via command-line parameters

The following command should be typed on a single line:

```
hping2 --udp --spoof 10.0.0.1 --baseport 53 --destport 53
   --numeric --count 1 192.168.1.60
```

This command sends a packet that looks like UDP DNS traffic to see if the firewall permits it. The command sends a single (`--count 1`) udp (`--upd`) packet from the source port 53 (`--baseport 53`) to the destination port 53 (`--destport 53`) and destination address of the internal machine (`192.168.1.60`). Since the reply is not needed, a spoofed, non-routable source address is used (`--spoof 10.0.0.1`). No name resolution is done (`--numeric`) since it not needed. The source address will be varied in future commands to prevent the firewall from blacklisting any single address.

Again, on a single line:

```
hping2 --spoof 10.0.0.2 --syn --count 1 --baseport 20 --destport 20
   --numeric 192.168.1.60
```

© SANS Institute 2000 - 2005 Author retains full rights.

This command is very similar to the first. This command sends a packet that looks like TCP FTP return data connection. Some firewalls may allow the return FTP data connection. The parameters are the same as the first command with the exception of the port numbers.

The next few commands attempt to fool the firewall by setting unexpected TCP state flags. The parameters have already been explained previously. The relevant parameters for each command are bolded.

Spoof an internal address:
```
hping2 --spoof 192.168.1.50 --syn --count 1 --destport 23 --numeric
  192.168.1.60
```
Spoof the external address of firewall:
```
hping2 --spoof 192.168.75.1 --syn --count 1 --destport 23 --numeric
   192.168.1.60
```
Spoof an external address of company's other firewall:
```
hping2 --spoof 192.168.99.1 --syn --count 1 --destport 23 --numeric
   192.168.1.60
```
Set no flags:
```
hping2 --spoof 10.0.0.1 --count 1 --destport 23 --numeric
192.168.1.60
```
Set the syn flag for port 80:
```
hping2 --spoof 10.0.0.2 --syn --count 1 --destport 80 --numeric
   192.168.1.60
```
Set the syn flag for port 23:
```
hping2 --spoof 10.0.0.3 --syn --count 1 --destport 23 --numeric
   192.168.1.60
```
Set the fin flag:
```
hping2 --spoof 10.0.0.4 --fin --count 1 --destport 23 --numeric
   192.168.1.60
```
Set the ack flag:
```
hping2 --spoof 10.0.0.5 --ack --count 1 --destport 23 --numeric
192.168.1.60
```
Set the syn and ack flag:
```
hping2 --spoof 10.0.0.6 --ack --syn --count 1 --destport 23 --numeric
   192.168.1.60
```
Set the reset flag:
```
hping2 --spoof 10.0.0.7 --rst --count 1 --destport 23 --numeric
   192.168.1.60
```
Set the push flag:
```
hping2 --spoof 10.0.0.8 --push --count 1 --destport 23 --numeric
192.168.1.60
```
Set the urgent flag:
```
hping2 --spoof 10.0.0.9 --urg --count 1 --destport 23 --numeric
   192.168.1.60
```
Set the Xmas or Explicit Congestion Notification (ECN) flag:
```
hping2 --spoof 10.0.0.10 --xmas --count 1 --destport 23 --numeric
   192.168.1.60
```
Set the Ymas or Congestion Window Reaction (CWR) flag:
```
hping2 --spoof 10.0.0.11 --ymas --count 1 --destport 23 --numeric
   192.168.1.60
```
Set all the TCP flag:

© SANS Institute 2000 - 2005, Author retains full rights.

```
hping2 --spoof 10.0.0.12 --ack --syn --fin --rst --push --urg --xmas
   --ymas --count 1 --destport 23 --numeric 192.168.1.60
```

This is by no means an exhaustive test of all the TCP flag combinations or other odd and malformed packets that can be seen by the firewall. Observed results of the firewall logs and ethereal packet capture should be recorded after running each command. If packets make it through the firewall and are detected by the internal computer then the firewall ruleset should be reviewed to see if there is any justification for the packet being allowed.

Test 3

Test 3 will determine how many simultaneous connections the firewall can handle before the connection cache is saturated. It is possible that the firewall can handle more connections than can reasonably be created and maintained by one computer connecting to another computer. In this case, the test shows that the cache is "unlimited" even though this is known not to be true. The test is first run with the internal computer initiating the connections. Since the ruleset does not block outbound connections, this test will show if a single computer can attempt sufficient connections to fill the cache. If the initial test is successful in demonstrating the cache limit, then the client and server roles will be reversed with the external computer initiating connections.

At the end of this section is a Perl script written by the author to specifically test the number of connections supported by the SonicWALL connection cache. The term "connection" includes UDP traffic since the cache creates an entry for UDP traffic. The script is documented with comments throughout the program. The script, named "connections.pl", will need to be copied to both the internal and external computer.

Before the script is run, the limit of simultaneous open files needs to be adjusted. The script maintains an array of TCP socket handles. If the limit is not changed, the script will quit after reaching the maximum number of open handles at 1024. The following command should be run on both the internal and external computer to raise the limit to 100,000 open files:

```
ulimit -n 100000
```

On the external computer run:

```
connections.pl --mode=server --port=9999 --proto=tcp
```

On the internal computer run:

```
connections.pl --mode=client --ipaddr=10.0.0.1 --port=9999 --
   proto=tcp --maxConn=32000
```

While the test is running, the firewall logs should be reviewed on the management computer for signs of a full cache. Once the script on the internal computer finishes or dies, both the client and server should be rerun using UDP instead of TCP in the script parameters. If the test is able to fill the cache then

© SANS Institute 2000 - 2005

Author retains full rights.

the roles of the internal and external computer need to be reversed with the external computer becoming the client.

On the external computer run:

```
connections.pl --mode=client --ipaddr=192.168.1.60 --port=80
   --proto=tcp --maxConn=32000
```

On the internal computer run:

```
connections.pl --mode=server --port=80 --proto=tcp
```

Code listing for “connections.pl”

```
#!/usr/bin/perl
use Getopt::Long;
use Socket;
# This program is used to create and maintain many connections.
# The following are the available options:
#   --mode       can be "client" or "server"; defaults to "server"
#   --ipaddr     IP address or host name for the client to connect
to;
#                defaults to localhost
#   --port       Port number for the client to send to or the server
#                to bind to; defaults to 9999
#   --proto      can be "tcp" or "udp"; default to "tcp"
#   --maxconn    number of client sockets to create

$| = 1; #unbuffer stdout

# Process the options and set defaults
GetOptions(\%options, "mode=s", "ipaddr=s", "port=i",
           "proto=s", "maxconn=i");
my $mode    = $options{'mode'}   || "server";
my $ipaddr  = $options{'ipaddr'} || "127.0.0.1";
my $port    = $options{'port'}   || 9999;
my $proto   = $options{'proto'}  || "tcp";
my $maxConn = $options{'maxconn'}|| 10;

my @sockets;
my $type = ($proto =~ /udp/i) ? SOCK_DGRAM : SOCK_STREAM;

if ($mode =~ /client/i) # Client processing
{
   print "Client starting.\n";

   # Create the destination address structure
   my $dest = sockaddr_in($port, inet_aton($ipaddr)) || die $!;
   for (my $count=1; $count<=$maxConn; $count++)
   {
      local *SH;

      # Create the local socket
      socket(SH, PF_INET, $type, getprotobyname($proto)) || die $!;

      # Connect to the destination address
      connect(SH, $dest) || die $!;

      if ($proto =~ /tcp/i) # For TCP
```

© SANS Institute 2005, Author retains full rights.

```
      {
         # Push the socket on an array.  This is needed to keep the
         # socket from closing when the reference goes out of scope.
         # It is not needed for udp since closing the socket
         # has no effect.
         push(@sockets, *SH);
      }
      else # for UDP
      {
         # Since there is no connection, some traffic must be sent
         # to the server
         print SH "\n" || die $!;
      }
      print "$count\n"; # Print the number of sockets on the display
   }
}
else # Server processing
{
   # Create the local address structure.  All interfaces are used.
   my $local = sockaddr_in($port, INADDR_ANY) || die $!;

   # Create the local socket.  Bind the local socket to the
   # local address and port.  Listen on the socket for
   # incoming traffic.
   socket(SH, PF_INET, $type, getprotobyname($proto)) || die $!;
   bind(SH, $local) || die $!;
   listen(SH, SOMAXCONN);
   print ("Server listening on port $port.\n");

   while (TRUE) # Loop forever
   {
      if ($proto =~ /tcp/i) # For TCP
      {
         # Accept the remote connection from the client and push the
         # client socket on an array.  This is needed to prevent the
         # server from closing the connection when the reference goes
         # out of scope.
         local *CLIENT;
         accept(CLIENT, SH) || die $!;
         push(@sockets, *CLIENT);
      }
      else # For UDP
      {
         # Read on byte from the socket
         read SH, $dummy, 1;
      }
      # Print a trail of dots to know that server is responding.
      print ".";
   }
}
```

Task 4: Audit

Results of Test 1

The nmap scan of the servers from inside the firewall shows the ports that have services listening on them. Because not all ports are scanned, there may be

© SANS Institute 2000 - 2005, Author retains full rights.

ports that are open that are not represented in this scan. Parts of the verbose output are not shown here.

```
nmap -sT -P0 -n -v -m internal -T polite 192.168.100.50-52

Starting nmap 3.75 ( http://www.insecure.org/nmap/ ) at 2005-02-19
17:05 EST
Initiating Connect() Scan against 3 hosts [1663 ports/host] at 17:05

Host 192.168.100.50 appears to be up ... good.
Interesting ports on 192.168.100.50:
(The 1651 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
135/tcp   open  msrpc
443/tcp   open  https
445/tcp   open  microsoft-ds
1025/tcp  open  NFS-or-IIS
1433/tcp  open  ms-sql-s
3372/tcp  open  msdtc
3389/tcp  open  ms-term-serv
8000/tcp  open  http-alt
8443/tcp  open  https-alt
38292/tcp open  landesk-cba

Host 192.168.100.51 appears to be up ... good.
Interesting ports on192.168.100.51:
(The 1652 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE
25/tcp    open  smtp
110/tcp   open  pop3
135/tcp   open  msrpc
143/tcp   open  imap
443/tcp   open  https
1025/tcp  open  NFS-or-IIS
1058/tcp  open  nim
1059/tcp  open  nimreg
3372/tcp  open  msdtc
3389/tcp  open  ms-term-serv
10000/tcp open  snet-sensor-mgmt

Host 192.168.100.52 appears to be up ... good.
Interesting ports on 192.168.100.52:
(The 1650 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
21/tcp   open  ftp
80/tcp   open  http
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
1027/tcp open  IIS
1030/tcp open  iad1
1031/tcp open  iad2
1433/tcp open  ms-sql-s
```

© SANS Institute 2000 - 2005, Author retains full rights.

```
2105/tcp open  eklogin
3372/tcp open  msdtc
```

The following is the results of the nmap scan from outside the firewall. As expected, it shows a subset of the ports available from the internal scan. The ports shown are all available to the public through the firewall. The available ports are appropriate for the type of services offered on each server.

```
nmap -sT -P0 -n -v -m external -T polite 192.168.100.50-52

Starting nmap 3.75 ( http://www.insecure.org/nmap/ ) at 2005-02-19
17:35 EST
Initiating Connect() Scan against 3 hosts [1663 ports/host] at 17:35

Host 192.168.100.50 appears to be up ... good.
Interesting ports on 192.168.100.50:
(The 1659 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE
21/tcp   open  ftp
80/tcp   open  http
443/tcp  open  https
8443/tcp open  https-alt

Host 192.168.100.51 appears to be up ... good.
Interesting ports on 192.168.100.51:
(The 1659 ports scanned but not shown below are in state: closed)
PORT    STATE SERVICE
25/tcp  open  smtp
110/tcp open  pop3
143/tcp open  imap
443/tcp open  https

Host 192.168.100.52 appears to be up ... good.
Interesting ports on 192.168.100.52:
(The 1660 ports scanned but not shown below are in state: closed)
PORT    STATE SERVICE
21/tcp  open  ftp
80/tcp  open  http
443/tcp open  https
```

The comparison of the two nmap scans by the ndiff utility shows which of the scanned ports were blocked by the firewall.

```
-----------------------------------------------------
ndiff run Thu Feb 24 12:00:21 EST 2005

command line: -b internal -o external -oh c
baseline: internal
observed: external
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
changed hosts:

192.168.100.50
    135/tcp       open -> closed
    445/tcp       open -> closed
   1025/tcp       open -> closed
   1433/tcp       open -> closed
```

© SANS Institute 2000 - 2005, Author retains full rights.

```
   3372/tcp       open -> closed
   3389/tcp       open -> closed
   8000/tcp       open -> closed
  38292/tcp       open -> closed

192.168.100.51
    135/tcp       open -> closed
   1025/tcp       open -> closed
   1058/tcp       open -> closed
   1059/tcp       open -> closed
   3372/tcp       open -> closed
   3389/tcp       open -> closed
  10000/tcp       open -> closed

192.168.100.52
    135/tcp       open -> closed
    139/tcp       open -> closed
    445/tcp       open -> closed
   1025/tcp       open -> closed
   1027/tcp       open -> closed
   1030/tcp       open -> closed
   1031/tcp       open -> closed
   1433/tcp       open -> closed
   2105/tcp       open -> closed
   3372/tcp       open -> closed
-------------------------------------------------------
```

The ruleset from the firewall shows that the firewall is configured in a default deny stance. Specific rules must be written to allow certain ports. Any port not listed is denied. The scan results are easily mapped to the ruleset. Rules 4 through 10 were not seen in the scan results. This may be caused by the scan not hitting that range of ports or that there is no active service for those ports

© SANS Institute 2000 - 2005, Author retains full rights.

Action	Service	Source	Destination
1 Allow	SMTP(25)	WAN	192.168.100.51(DMZ)
2 Allow	POP3(110)	WAN	192.168.100.51(DMZ)
3 Allow	IMAP4(143)	WAN	192.168.100.51(DMZ)
4 Allow	EmailCalendar(8484)	WAN	192.168.100.51(DMZ)
5 Allow	EmailAdmin(8181)	WAN	192.168.100.51(DMZ)
6 Allow			

© SANS Institu

s full rights.

The test does not show any specific misconfiguration in the ruleset. Since corporate policy allows for ports to be open that are needed for business, the ruleset meets the policy. However, inspection of the ruleset reveals possible issues that are not uncovered by the test. The firewall administrator should determine if the ports specified in rules 4 through 10 are not need or were just not scanned for in the test. There may be an overly broad opening of certain ports. Rules 11 through 13 allow web and ftp traffic for all servers, even ones that are not serving those ports. This could be an issue for a new server that accidentally was configured to respond on those ports. Instead of globally allowing these ports for all servers, additional rules should be created so that the ports are only available for specific servers. Additionally, rules 15 through 17 allow three specific host addresses to bypass all firewall rules. These rules should be validated to ensure they are still necessary.

Results of Test 2

The firewall properly implemented the ruleset and did not let any packets through that it was not supposed to. Additionally, none of the malformed packets caused the firewall to stop processing legitimate packets. Figure 3 shows all the packets received by the internal computer that were sent from the external computer. The output from the external computer is not shown since it is not relevant to the test. The results of this test are the output displayed on the Ethereal packet capture.

Although there were no specific rules configured on the firewall for this internal computer, the stealth nmap fin scan was able to communicate to port 80. The firewall is configured to allow access to port 80 for all servers.

© SANS Institute 2000 - 2005, Author retains full rights.

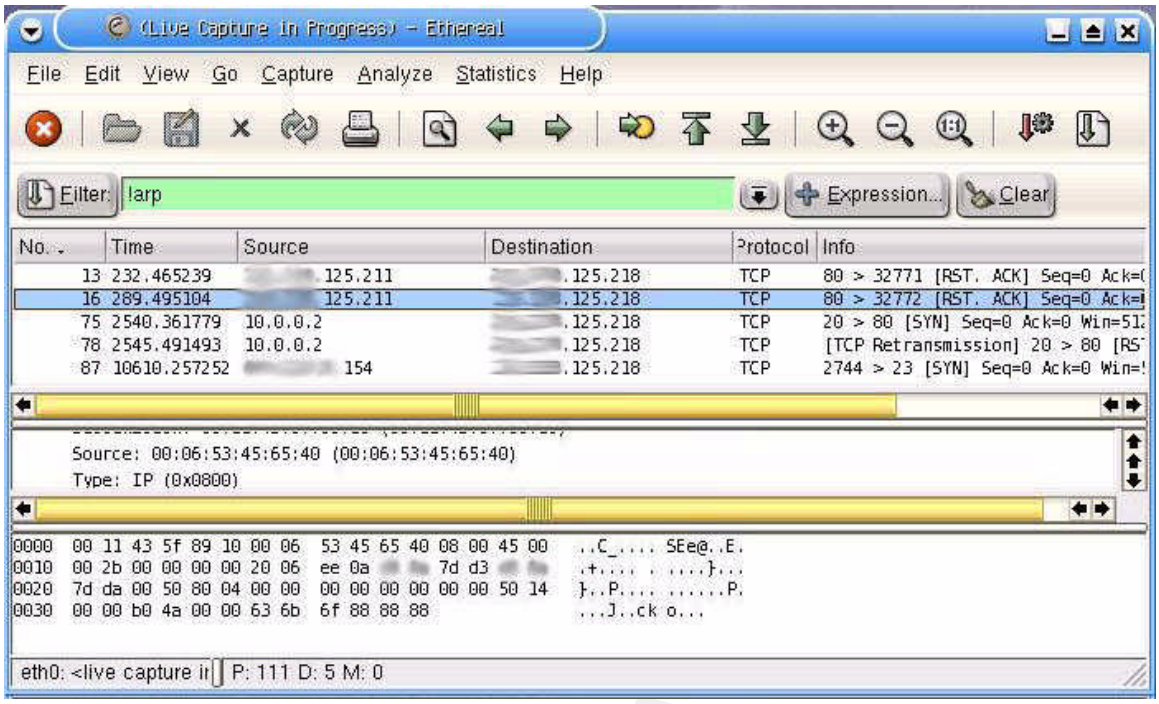


Figure 3: Ethereal Packet Capture

The hping2 command that attempted to spoof UDP DNS traffic was blocked and the firewall logged this message:

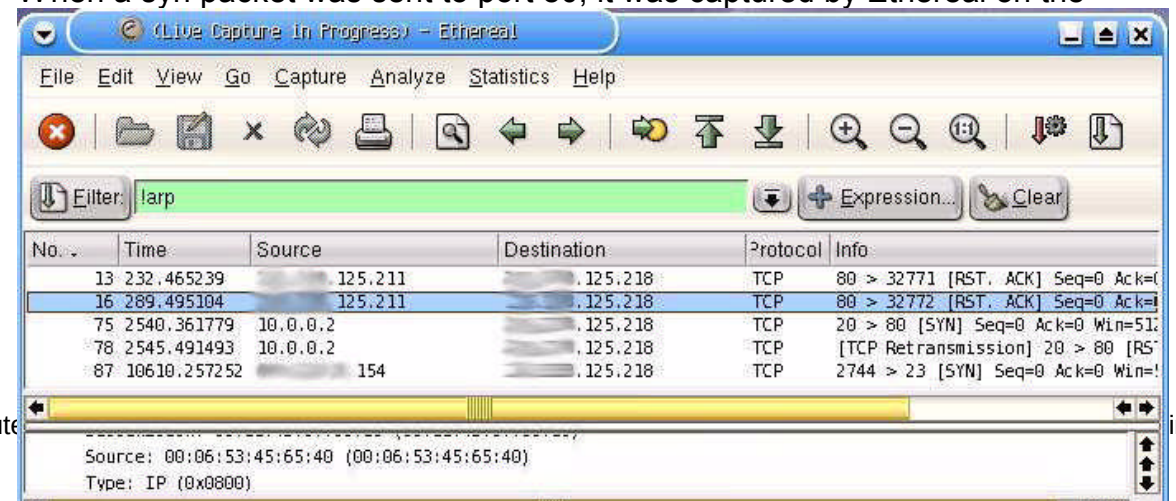
02/19/2005 18:35:34.400	UDP packet dropped	10.0.0.1, 53, WAN	192.168.100.60, 53, DMZ

When an internal or firewall IP address was used as the spoofed source address the firewall blocked access and logged this message:

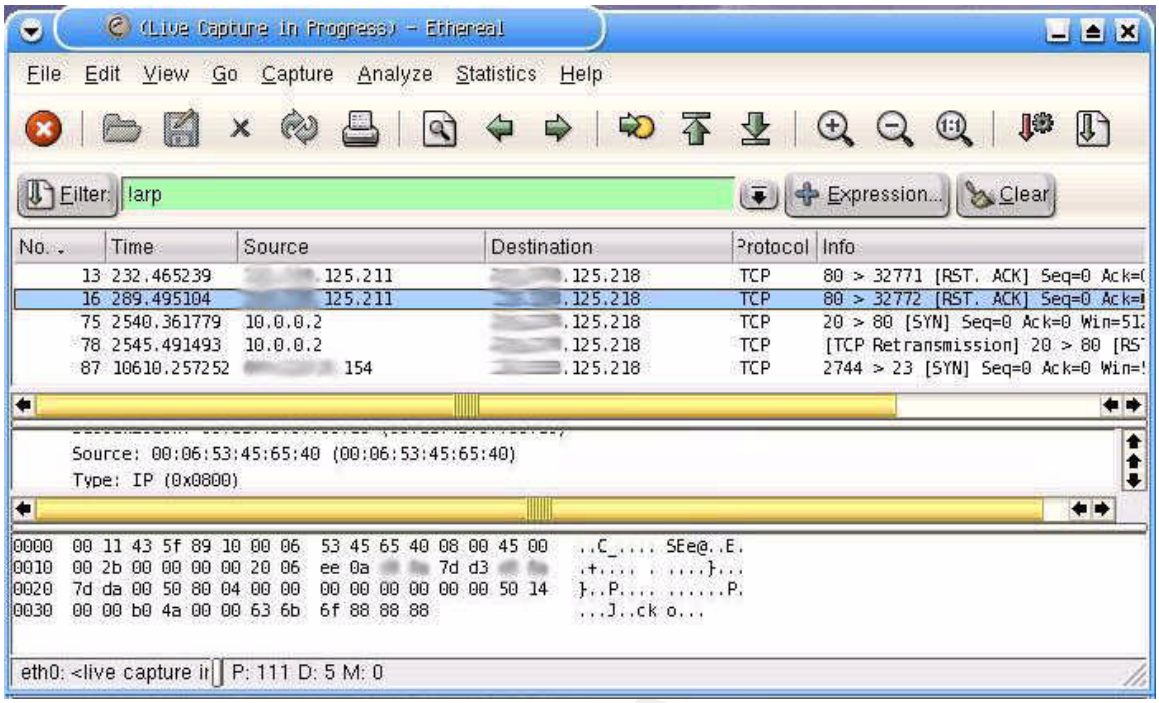
02/19/2005 20:43:31.032	IP spoof detected	192.168.100.50, 2548, WAN	192.168.100.60, 23, DMZ	MAC address: 00.11.43.3D.C4.34

In Figure 3, the line numbered as 87 shows that when the company's other external firewall's IP address was spoofed, the packet was permitted. Firewall rule 16 permits all TCP traffic from this IP to the servers. This rule allows the servers to be managed remotely from the company's other network. The rule also opens a large hole in the firewall that could be exploited with spoofed source addresses.

When a syn packet was sent to port 80, it was captured by Ethereal on the



© SANS Institute ins full rights.



internal computer because firewall rule 11 lets all servers communicate on port 80. When a syn packet was sent to port 23, it was blocked and logged by the firewall:

02/19/2005 20:46:32.768	TCP connection dropped	10.0.0.3, 1706, WAN	192.168.100.60, 23, DMZ	Telnet	18

The log entry shows that firewall rule 18, which is the default deny from the WAN, was used to block the syn packet. No combination of TCP flags that was used in the test caused the firewall to fail. With the exception of the log entry noted here, none of the blocked packets were logged by the firewall.

Results of Test 3

The Perl script to generate and maintain simultaneous connections worked as expected. The output from the server and client are shown below. The output has been edited to show the beginning and ending. The repetitive middle was removed.

Server output on the external computer:

```
./connections.pl --mode=server --port=9999 --proto=tcp
Server listening on port 9999.
.....................................................................
.....................................................................
.....................................................
```

Client output on the internal computer:

```
./connections.pl --mode=client --ipaddr=10.0.0.1 --port=9999
--proto=tcp --maxconn=32000
Client starting.
1
2
```

```
3
…
1922
1923
1924
Connection refused at ./connections.pl line 43.
```

The script on the client computer died after successfully creating 1924 TCP connections. The firewall logs had this entry:

02/19/2005 21:28:16.192	The cache is full; too many open connections; some will be dropped	192.168.100.60, 45828, DMZ	10.0.0.1, 9999, WAN

Repeating this test multiple times always caused the client script to die at approximately the same connection number. Similar results were also found when the external computer became the client and the internal computer was the server. This demonstrates that the firewall can handle approximately 2000 simultaneous connections before the connection cache is full. The firewall is vulnerable to a DoS condition once the cache becomes saturated since no new connections are permitted. This vulnerability can be exploited by external users. However there are a few mitigating factors that work to minimize the impact of malicious users.

1. There must be a server with a listening port that is permitted by the firewall. A blocked connection is not stored in the cache.
2. The service must be capable of handling large numbers of simultaneously connected clients. Queued connection requests are not stored in the cache.
3. The client must control when the connection is terminated. Terminated connections are removed from the cache.

In the current configuration for GSD, the web server is probably the only service that can sustain large numbers of multiple connections. Since the connection is terminated by the server after the request is fulfilled, it is unlikely that a malicious client could sustain too many connections. Although the firewall is vulnerable to this DoS attack, it would be difficult to exploit this vulnerability from outside the firewall.

Conclusion

As should be expected for any firewall product, the SonicWALL Plus DMZ firewall properly implements the firewall ruleset. This does not, however, prevent the administrator from making incorrect entries in the ruleset. All security infrastructure is vulnerable to failure due to misconfiguration. The procedures for test 1 demonstrate that the firewall is working as expected by running scans inside and outside the firewall. A comparison of the two scans shows the impact of the firewall. Test 1 concludes with a review of the firewall ruleset and a comparison to the corporate policy and industry best practice. The results of the test show that a review of the current ruleset is needed to ensure that unneeded entries are deactivated or removed.

© SANS Institute 2000 - 2005 Author retains full rights.

The test 2 procedures simulated hostile attacks on the firewall. Packets with non-standard and unexpected TCP flags were sent in an attempt to confuse the firewall. The results of test 2 demonstrate that the firewall can properly withstand an attack of malformed packets. However, the test did highlight possible issues with the ruleset. The entries for FTP and web access are overly broad, allowing those ports for any server. Also, the three rules with IP addresses that are allowed full access to the servers should be reviewed to ensure they are essential to proper operation.

The test 3 procedures were designed to determine the size limitations of the connection cache used for stateful packet inspection. The results of test 3 demonstrate that the firewall can handle approximately 2000 simultaneous connections. Additional connection attempts are rejected by the firewall until the cache has space, thus the firewall is vulnerable to this DoS condition. There are a few mitigating factors that reduce the ability for an external attacker to successfully implement this attack.

Although this firewall is about six years old, it still provides the necessary security for GSD. If future web and email traffic increases, it is possible a full connection cache condition may occur more frequently. If the 2000 simultaneous connection limit is not acceptable, then a higher capacity firewall will need to be used instead.

© SANS Institute 2000 - 2005 Author retains full rights.

References

Green, John. "SonicWALL Plus DMZ". Windows IT Pro. 1 May 1999. 19 Dec. 2004. <http://www.win2000mag.com/Articles/Index.cfm?ArticleID=5136>

Grosse, Paul. "Sonic Systems Inc. SonicWALL" Personal homepage. June 1999. 4 Jan. 2005. <http://ourworld.compuserve.com/homepages/pagrosse/j02f.htm>

Jackson, William. "Sonic security appliances do job with little management". Government Computer News. 12 Apr. 1999. 27 Dec. 2004. <http://www.gcn.com/18_9/news/34092-1.html>

Phifer, Lisa. "VPN RFP Lab Eval: SonicWALL". ISP-Planet. 25 Oct. 2001. 19 Dec. 2004. <http://www.isp-planet.com/technology/vpn/sonicwall_eval1.html>

Ranum, Marcus. "Firewall Product Functional Summary". Information Warehouse! Inc. 7 June 2000. 21 Dec. 2004 <http://www.icsalabs.com/html/communities/firewalls/certification/rxvendors/sonicwallpro/pfd.pdf>

Stoneburner, Gary, et al. "Risk Management Guide for Information Technology Systems". National Institute of Standards and Technology. July 2002. 15 Dec. 2004. <http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf>

Wack, John, et al. "Guidelines on Firewalls and Firewall Policy". National Institute of Standards and Technology. January 2002. 17 Dec. 2004. <http://csrc.nist.gov/publications/nistpubs/800-41/sp800-41.pdf>

© SANS Institute 2000 - 2005, Author retains full rights.