

Passive Vulnerability Scanner 3.6
Linux User Guide

September 20, 2013

(Revision 4)

The newest version of this document is available at the following URL:
http://static.tenable.com/prod_docs/PVS_3.6_Linux_user_guide.pdf

Copyright © 2002-2012 Tenable Network Security, Inc. Tenable Network Security, Nessus and ProfessionalFeed are registered trademarks of Tenable Network Security, Inc. Tenable, the Tenable logo, the Nessus logo, and/or other Tenable products referenced herein are trademarks of Tenable Network Security, Inc., and may be registered in certain jurisdictions. All other product names, company names, marks, logos, and symbols may be the trademarks of their respective owners.

Table of Contents

Introduction **5**
Standards and Conventions 5
Passive Vulnerability Scanner Background and Theory 5

Pre-Installation **6**
Hardware requirements 6
Software and Licensing 7
Download or Obtain the Software *7*
Obtain a Demo or Commercial Key *7*
Move the Key and Software to the Server *7*

Installation and Operations **8**
Upgrading from PVS 3.x 8
Red Hat *8*
Initial Installation 8
Run the Installation Script *9*
Post Installation *13*
File Locations *13*
Operations 13
Starting the Passive Vulnerability Scanner with the SecurityCenter *13*
Stopping the Passive Vulnerability Scanner *14*
Starting the Passive Vulnerability Scanner Manually *14*
Starting the Passive Vulnerability Scanner Manually as a Background Process *16*
The PVS Watchdog *17*
Using the Passive Vulnerability Scanner to Analyze TCPDUMP Capture Files *17*
Generating a Report *17*
Downloading New Vulnerability Plugins *18*
Configuration 18
pvs.conf File Format *18*
Specifying Focus Networks *23*
Using Multiple Interfaces *23*
Removing PVS 24
Red Hat *24*

PVS Real-Time Traffic Analysis Configuration Theory **24**
Focus Network 24
Detecting Server and Client Ports 25
Detecting Specific Server and Client Port Usage 26
What this Means for Firewall Rules 27
Working with the SecurityCenter 27
Selecting Rule Libraries and Filtering Rules 27
Detecting Encrypted and Interactive Sessions 27
Routes and Hop Distance 28
Alerting 28
New Host Alerting 28

Internal Passive Vulnerability Scanner IDs **29**

Copyright © 2002-2012 Tenable Network Security, Inc.

What is a Passive Vulnerability Scanner ID? 29
Internal Passive Vulnerability Scanner IDs 29

Working with Passive Vulnerability Scanner Plugins 30
Vulnerability and Passive Fingerprint Overview 30
Downloading New Vulnerability Plugins 30
Writing Custom Plugin Libraries 30
Restarting the Passive Vulnerability Scanner 31
Writing Passive Vulnerability Scanner Plugins 31
Plugin Keywords 31
Plugin Libraries 34
Basic Passive Vulnerability Scanner Example 34
More Complex Passive Vulnerability Scanner Example 35
Case Insensitive Example 36
Passive Vulnerability Scanner Network Client Detection 36
The Passive Vulnerability Scanner can Match "Previous" Packets 37
The Passive Vulnerability Scanner can Match Binary Data 38
Negative Matches 38
Time Dependent Plugins 39
Writing Passive Vulnerability Scanner Real-Time Plugins 40
Real-Time Plugin Model 40
New Keywords 40
Example Failed Telnet Login Plugin 41
Example Finger User List Enumeration Plugin 41
Example Unix Password File Download Web Server Plugin 42
Example Generic Buffer Overflow Detection on Windows Plugin 42
Passive Vulnerability Scanner Corporate Policy Plugins 44
Detecting Custom Activity Prohibited by Policy 44
Detecting Confidential Data in Motion 46

Passive Vulnerability Scanner Operating System Fingerprints 48
Passive Operating System Fingerprinting 48
Understanding the Fingerprint Language 48
Adding New Fingerprints 49

For Further Information 49

About Tenable Network Security 50

Appendix 1: Example pvs.conf Configuration File 51

Appendix 2: Working with SecurityCenter 56
Architecture 56
Managing Vulnerabilities 56
Managing IDS Events 56
The Passive Vulnerability Scanner is Real-Time 57

Appendix 3: Working with Nessus 58
Tuning for Nessus Only Reporting 58

Copyright © 2002-2012 Tenable Network Security, Inc.

Appendix 4: Non-Tenable License Declarations and Patent 60
Related 3rd Party and Open-Source Licenses 60

Copyright © 2002-2012 Tenable Network Security, Inc.

INTRODUCTION

This document describes the **Passive Vulnerability Scanner 3.6** architecture, installation, operation, integration with the SecurityCenter and export of data to third parties. Please email any comments and suggestions to support@tenable.com.

The Passive Vulnerability Scanner 3.6 is available for the following platforms:

- Red Hat Linux ES 4
- Red Hat Linux ES 5 / CentOS 5
- Red Hat Linux ES 5 / CentOS 5 64-bit
- Red Hat Linux ES 6 / CentOS 6
- Red Hat Linux ES 6 / CentOS 6 64-bit

This document describes the Passive Vulnerability Scanner installation and operation on the Linux platform and includes the theory and operational details related to the implementation.

STANDARDS AND CONVENTIONS

Throughout the documentation, filenames, daemons, and executables are indicated with a **courier bold** font such as **gunzip**, **httpd**, and **/etc/passwd**.

Command line options and keywords are also indicated with the **courier bold** font. Command line examples may or may not include the command line prompt and output text from the results of the command. Command line examples will display the command being run in **courier bold** to indicate what the user typed while the sample output generated by the system will be indicated in courier (not bold). Following is an example running of the Unix **pwd** command:

```
# pwd
/opt/pvs
#
```



Important notes and considerations are highlighted with this symbol and grey text boxes.



Tips, examples, and best practices are highlighted with this symbol and white on blue text.

PASSIVE VULNERABILITY SCANNER BACKGROUND AND THEORY

Passive vulnerability scanning is the process of monitoring network traffic at the packet layer to determine topology, clients, applications, and related security issues. Tenable has expanded the functionality of the Passive Vulnerable Scanner (PVS) to include traffic profiling and system compromise detection. The PVS can not only keep track of the vulnerabilities for 20,000 to 50,000 systems at a time, it can also:

- detect when they are compromised based on application intrusion detection

Copyright © 2002-2012 Tenable Network Security, Inc.

- highlight all interactive and encrypted network sessions
- detect when new hosts are added to a network
- track exactly which systems communicate with other systems and on what ports
- detect what ports are served and what ports are browsed by each system
- detect how many hops away each monitored host is

This document provides directions for deploying, configuring and operating the PVS. Please see the PVS Overview whitepaper in the Expert Resources section of the Tenable web site for a description of the PVS technology and example use cases.

PRE-INSTALLATION

To ensure a streamlined installation process, it is important to ensure that the appropriate hardware, software and licensing requirements are in place prior to installation.

HARDWARE REQUIREMENTS

Enterprise networks can vary in performance, capacity, protocols and overall activity. PVS deployments should consider raw network speed, the size of the network being monitored and the configuration of the PVS application when determining resource requirements.

The following chart outlines some basic hardware requirements for operating PVS:

Scenario	Recommended Hardware
Passive Vulnerability Scanner managing up to 20,000 – 50,000 hosts *	CPU: 1 dual-core 2 GHz CPU Memory: 2 GB RAM (4 GB RAM recommended)
Passive Vulnerability Scanner managing in excess of 50,000 hosts	CPU: 1 dual-core 3 GHz CPU (2 dual-core recommended) Memory: 2 GB RAM (4 GB RAM recommended) HDD: 72 GB at 10,000 rpm (72 GB at 15,000 rpm recommended)



* Note: The ability to monitor a given number of hosts rests heavily on the bandwidth, memory, and processor power available to the system running PVS.



Please research your VM software vendor for recommendations as VMs typically see a 12 to 30% loss in efficiency compared with dedicated servers.

Processor requirements will increase with greater throughput and number of network interfaces. Memory requirements will increase for networks with more hosts. The requirements for both of these components are affected by options such as a long report-lifetime and enabling some or all of the PVS optional services in the configuration file.

Copyright © 2002-2012 Tenable Network Security, Inc.

Software and Licensing

Download or Obtain the Software

To install the PVS, obtain the correct version for your desired operating system from the Tenable Support Portal. Confirm the integrity of the installation package by comparing the download MD5 checksum with the one listed in the product release notes.



It is important to ensure that the correct build for your operating environment is downloaded since they are not binary compatible with each other.

Obtain a Demo or Commercial Key

To obtain a 30 day demo key for the PVS, contact sales@tenable.com. Commercial keys and demo keys are handled the same way by the PVS, except that a demo key will expire. During a demo evaluation of the PVS, all of the features are available. The PVS uses two different licensing modes:

- **Host-Based** – A separate license is needed for each host where a PVS is deployed.
- **Network-Based** – This licensing method is based on the network that the PVS will monitor. Choose from either **Class C** (up to 255 contiguous hosts) or **Class B** (up to 65,535 contiguous hosts) options based on the size of the monitored network.



These licenses cannot cross class boundaries. For example, specifying 128 hosts from 10.32.4.0/24 and 128 hosts from 10.32.5.0/24 is not permitted.

As many PVS hosts as required can be deployed per licensed subnet and some or the entire targeted network can be monitored at any point. If an IP range greater than the license range is specified in the PVS configuration, it will be truncated to the license range. If a narrower range than the licensed range is specified, the narrower range will be used. For example, a university with a license of 10.32.0.0/16 may deploy a PVS with that license with a network range of 10.32.66.0/24. Specifying a monitored range of 10.32.0.0/8 would not increase the IP base from the licensed base of 10.32.0.0/16.



Note that if this is an upgrade, a new license must be requested from Tenable Support. Licenses acquired prior to PVS 3.6 are not compatible with newer versions of the software.

Move the Key and Software to the Server

The key file location and name will be required during the installation process. You may wish to make a note of this before beginning the installation such as:

```
/tmp/tenable-15-47.key
```

Copyright © 2002-2012 Tenable Network Security, Inc.

INSTALLATION AND OPERATIONS

UPGRADING FROM PVS 3.X

Red Hat

If you have used a PVS RPM to install PVS previously, an upgrade is simple and retains configuration settings.

Transfer the PVS RPM package to the system it is being installed on. Confirm the integrity of the installation package by comparing the download md5 checksum with the one listed in the product release notes.

Before upgrading you will first need to stop the PVS service:

```
# service pvs stop
```

Install the PVS software with the following command. Note that the specific filename will vary, depending on your version:

```
# rpm -Uvh pvs-x.x.x-esx.i386.rpm

Preparing...                ########################################[100%]
   1:pvs                    ########################################[100%]

[*] PVS and PVS Proxy have been installed with default configuration files
[*] Any configuration files from previous installations have been retained
[*] You may now run the post-install script at /opt/pvs/bin/pvs-post-
       install.sh
```

Run the installation script **/opt/pvs/bin/pvs-post-install.sh** and when prompted, use the upgrade option "**u**". Choosing the upgrade option will not affect your current configurations.

Once the upgrade is complete, restart the PVS service with the following command:

```
# service pvs start
```

INITIAL INSTALLATION



In order to ensure audit record time stamp consistency between PVS and SecurityCenter, make sure that the underlying OS makes use of NTP as described in:

http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Deployment_Guide/sect-Date_and_Time_Configuration-Command_Line_Configuration-Network_Time_Protocol.html

Copyright © 2002-2012 Tenable Network Security, Inc.



Unless otherwise noted, perform all commands as the system's root user.

Install the PVS software for Red Hat with the following command. Note that the specific filename will vary, depending on your version:

```
# rpm -ivh pvs-x.x-es3.i386.rpm

Preparing...                ########################################[100%]
   1:pvs                    ########################################[100%]

[*] PVS and PVS Proxy have been installed with default configuration files
[*] Any configuration files from previous installations have been retained
[*] You may now run the post-install script at /opt/pvs/bin/pvs-post-
      install.sh
```

The installation will create the directory **/opt/pvs** that contains the PVS software.

Run the Installation Script

The following is an example of running the installation script that configures the PVS. The commands entered by the user are shown in boldface and are example answers to the prompts. Your actual responses may vary. Additional configuration tuning may be performed after the initial configuration by editing the **/opt/pvs/etc/pvs.conf** file. A summary of the configuration options may be found in the "Configuration" section later in this document.

```
# /opt/pvs/bin/pvs-post-install.sh
```

The screen will clear and display the following:

```
--------------------------------------------------------------------------------
                          PVS INSTALLATION SCRIPT
--------------------------------------------------------------------------------

                         TENABLE NETWORK SECURITY
                          http://www.tenable.com
                            support@tenable.com
                            Copyright 2003-2012


Welcome to the PVS configuration script !

This script will help you configure your PVS installation on your system.

Press ENTER to continue
```

After pressing "ENTER", the following is displayed:

```
Was the installation you performed with the RPM a fresh install (i) or an
      upgrade (u) of PVS ? [i] i
```

Copyright © 2002-2012 Tenable Network Security, Inc.

Entering "**i**" for a new installation initiates prompts for configuration options.

The first screen displayed permits setting of the network devices.

All Unix servers have network devices that can be placed into "promiscuous" mode to sniff network packets. The PVS needs to know which device to use for sniffing. On Linux systems, this is typically "eth0", with the 0 varying depending on the configuration of the server. On Red Hat and other Unix systems, the **ifconfig -a** command can be used to list all of the active interfaces. The PVS can be configured to read from more than one interface, but to do this requires manual modification of the **/opt/pvs/etc/pvs.conf** file once installation is complete. An example of the screen displayed follows:

```
--------------------------------------------------------------------------------
                    PVS CONFIGURATION : Network Interface
--------------------------------------------------------------------------------


PVS listens for vulnerabilities by "sniffing" the data flowing through
       your
network. As a result, it should be attributed a specific network interface
to listen data on. This interface should be linked spanned port of a
       switch, or should be linked to a hub.

On which network interface will PVS be listening on ? (ie: eth0, fxp0)
       eth0
```

Next, the configuration script prompts for networks to monitor.

The PVS builds a model of hosts with ports, applications, and vulnerabilities based on the observed network traffic. The PVS needs to be configured to specifically focus on one or more networks. When the PVS focuses on a network, it will only consider the hosts within that network for client and server vulnerabilities. For example, your web server may have thousands of visitors each day. Unless you want the PVS to conduct passive vulnerability analysis of each visitor, specify the network of the web server.

When entering these networks for the PVS to focus on, IP addresses and CIDR notation is supported. There is no limit to the number of networks the PVS can focus on. When you are finished entering networks to focus on, press "Ctrl+D" to finish.

An example of the screen displayed is as follows:

```
--------------------------------------------------------------------------------
                   PVS CONFIGURATION : Networks to monitor
--------------------------------------------------------------------------------

PVS can perform passive vulnerability detection on network packets. It
       needs
to be configured with the information for which networks it is protecting.

For example, your network may have a web server, but you do not wish to
perform passive vulnerability analysis on the web clients visiting your
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
       web
server.  To do this, only list the network CIDR blocks for which you want
vulnerability reports for.

PVS will analyze any client or server vulnerabilities on only those
       networks.

Valid inputs are :

        192.168.1.0/24
        10.0.0.0/255.0.0.0

If no CIDR blocks are specified, the default of 0.0.0.0/0 will be chosen,
causing all network traffic to be monitored.

Press [ENTER] after every CIDR block.
Press Ctrl+D when you are done.

192.168.10.0/24
You entered :
192.168.10.0/24

Is this correct? (y/n) [y]y
```

The PVS configuration script next asks if the PVS is to be used with a SecurityCenter. Answering yes to this question causes the PVS installation to create a set of start scripts and configurations that allow PVS data to be sent to a SecurityCenter and to also receive new passive vulnerability plugins from a SecurityCenter as well.

This is accomplished by the **pvs-proxy** daemon, which performs a similar function as the SecurityCenter's **lightning-proxy** daemon, but is specifically tuned to upload PVS data to the SecurityCenter and to download new PVS passive plugins.



Plugins are pushed to the PVS from SecurityCenter or downloaded directly from Tenable via a provided script. PVS verifies the digital signatures of all plugin downloads to ensure file integrity.

To communicate with the SecurityCenter, this PVS installation needs to establish its own unique username and password. This credential information needs to be configured on the SecurityCenter through the administration interface so the SecurityCenter can log into this PVS and retrieve vulnerability data. For more information on this, please refer to SecurityCenter Administration Guide.

If the answer to this question is "y", the user will be prompted for a username and password. Otherwise, the **pvs-proxy** will not be configured. An example of this is as follows:

```
--------------------------------------------------------------------------------
                PVS CONFIGURATION : SecurityCenter Uplink
--------------------------------------------------------------------------------
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
PVS can report its data to the SecurityCenter console for centralised
       management.

If you enable SecurityCenter support, PVS will run a daemon (pvs-proxy)
       which will be polled by SecurityCenter regularly to fetch all the
       new reports. To do so, you will need to set up a username and
       password for pvs-proxy and give these credentials to
       SecurityCenter.

Do you want to enable SecurityCenter support ? (y/n) [y]y

If you reply "y", you will be prompted for a username and password:
Username : admin
Password :
Password (confirm) :
```

Finally, the installation script will ask for the location of the PVS license key. It will attempt to copy the key into the PVS installation directory. The entire path and file name of the key when it was copied to the system is required. An example of this step is as follows:

```
------------------------------------------------------------------------------
                        PVS CONFIGURATION : License
------------------------------------------------------------------------------



What is the location of the PVS license key ? /tmp/tenable-pvs-xx-xx.key

Now generating the configuration files...
Now copying the configuration files to /opt/pvs/etc...
Copying the key to /opt/pvs/var/pvs/tenable.key...
Generating the proxy SSL keys...
```

If this step is successful, the following screen is displayed:

```
------------------------------------------------------------------------------
                        PVS Configuration : Finished
------------------------------------------------------------------------------


Congratulations ! PVS has been configured properly on this host.


To start PVS, run :

        service pvs start

This will run both PVS and the PVS proxy, and the reports will be picked
up by the SecurityCenter.
```

The following command will start PVS for Red Hat and CentOS:

```
# service pvs start
```

Copyright © 2002-2012 Tenable Network Security, Inc.



Starting PVS via the "RC" script will also lint the `pvs.conf` script for syntax errors before starting. If errors are detected, PVS will direct you to address the errors and will not start.

Post Installation

Once installed, the original RPM file or unpacked files are not needed and may be deleted if desired.

If multiple Ethernet devices should be monitored, simply edit the **`pvs.conf`** file located in **`/opt/pvs/etc`** and add multiple lines which contain "**`interface`**" keywords and the specified Ethernet devices. For example, a **`pvs.conf`** file edited to support multiple Ethernet devices on Linux may include a section like this:

```
# which interface are we listening on ?
interface "eth0";
interface "eth1";
interface "eth2";
```

File Locations

The PVS installs its files into the following locations:

Path	Purpose
`/opt/pvs`	Base directory
`/opt/pvs/etc`	Configuration files for PVS and the PVS Proxy
`/opt/pvs/bin`	Location of the PVS and PVS Proxy executables, plus several helper tools for the PVS Proxy daemon
`/opt/pvs/var`	Plugins, discovered vulnerabilities, log files, keys, software license agreement and other miscellaneous items

The software license agreement for PVS is located in PDF format in the directory **`/opt/pvs/var/pvs`**. It is also available online at:

http://cgi.tenable.com/Tenable_Passive_Vulnerability_Scanner_3.x_Software_License_Agreement.pdf

Operations

Starting the Passive Vulnerability Scanner with the SecurityCenter

If PVS for Red Hat is to be managed by the SecurityCenter, use the **`service`** command to launch the **`pvs`** and **`pvs-proxy`** binaries:

```
# service pvs start
Starting PVS Proxy[  OK  ]
Starting PVS[  OK  ]
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
# ps aux | grep pvs
root     25191 22.7 26.9 384388 274704 pts/2    Sl    15:26    0:05
       /opt/pvs//bin/pvs
root     25194  0.1  0.6  13312  6380 pts/2     S     15:26    0:00
       /opt/pvs//bin/pvs-proxy
root     25199  0.0  0.0 103300   844 pts/2     S+    15:26    0:00 grep pvs#
```

Once running, the **pvs** binary will be managed completely by the **pvs-proxy** daemon.

In this mode, the PVS will store its vulnerability (**.nsr** files) reports in a directory monitored by the PVS Proxy, **/opt/pvs/var/pvs-proxy/scans**. When the SecurityCenter connects to the PVS Proxy, it will copy any report found in this directory to the SecurityCenter and import it into the relevant customer databases.

Also, if the SecurityCenter has received new passive vulnerability plugins or real-time plugins from Tenable Network Security, Inc., it will install them in the PVS plugin directory and restart the PVS.

PVS data will be communicated to the SecurityCenter as often as it is reported. If a PVS is configured to log its current vulnerabilities every 30 minutes, the SecurityCenter will receive this data as soon as the SecurityCenter connects to it. If the PVS is only logging its data once every six hours, then the SecurityCenter will only get updates every six hours.

Stopping the Passive Vulnerability Scanner

If you need to stop the PVS service for any reason, use the following command for PVS for Red Hat and CentOS:

```
# service pvs stop
Stopping PVS Proxy[  OK  ]
Stopping PVS[  OK  ]
```

Or if you wish to restart the service use the command:

```
# service pvs restart
```

Starting the Passive Vulnerability Scanner Manually

This section describes how to start the PVS when it is used independently of the SecurityCenter. Starting the PVS with the SecurityCenter was described previously in this document.



Running PVS without specifying the full path will result in the host launching a tool that displays information about local physical volumes, and not Tenable's Passive Vulnerability Scanner. Ensure that you include the full path to Tenable's PVS as shown below.

To start the PVS binary manually, type the following command:

```
# /opt/pvs/bin/pvs -c /opt/pvs/etc/pvs.conf -r report.nsr
```

This will print out a message that will look similar to this:

Copyright © 2002-2012 Tenable Network Security, Inc.

```
Starting Passive Vulnerability Scanner
cb67c871206b18d743a5e070276bf13d  /opt/pvs/bin/pvs
Valid license detected for host pvshostname
Initializing PVS. Version: 3.6.0 Build ID: 201201182
Load of file /opt/pvs/var/pvs/disabled-scripts.txt is complete.
md5 checksum : 3e25b1d5076fa924393ad55fe1a95390
       /opt/pvs/var/pvs/plugins/tenable_plugins.prmx
Loaded include file: byte_func.inc
Loaded include file: cpe.inc
Loaded include file: cpe_db.inc
…
…
Loaded 4929 plugins
Filter expression: (net 0.0.0.0/0)
main - initialization completed
```

This indicates the PVS build, a valid license is installed, has 4229 plugins loaded, and the monitored network(s) are listed.

When running the PVS manually, the following command line options are available:

Command Line Option	Description
-c conf	path to the configuration file (default: **/opt/pvs/etc/pvs.conf**)
-r report	where to write the report
-l	show the list of loaded plugins
-p <packet dump file>	dump packet payload to file
-t <tree file>	dump tree to file
-f <dumpfile>	read packets from file
-m	print memory statistics to stdout
-L	lint the configuration file specified by **-c** for syntax errors
-h	the help screen
-v	show version information
-k <key file>	check the validity of a specified license file

Copyright © 2002-2012 Tenable Network Security, Inc.

The PVS requires the "**-c**" and "**-r**" options each time it runs. These options tell the PVS where to obtain its configuration information and where to write out its vulnerability report. The other options are available for diagnostics and to analyze TCPDUMP trace files.

By default, the current content of the reports will be written to disk in the file **pvs-report.nsr** every sixty minutes. Changing the **"report-frequency"** setting in the **pvs.conf** file will allow the report time to be modified. If a full path is not specified, **pvs-report.nsr** will be created in the location as specified by the **nsr-report-file** directive in the **pvs.conf** file. If you want the **pvs-report.nsr** file to be created in a specific directory, specify the full path, such as "**-r /home/jsmith/myreport.nsr**" when PVS is launched, or configure the location in **pvs.conf**.



Changing the name or location of the **nsr-report-file** directive will cause SecurityCenter to not be able to get the report file. Only change the directive if the file does not need to be accessed by SecurityCenter.



In addition, PVS automatically generates HTML reports for web browsers. These reports are located in the **/opt/pvs/var/pvs/reports/html/** directory.

This particular command line operates the PVS in the foreground. To stop it, enter "Ctrl+C" to kill the process, or send a **kill -9** command to the PVS process ID from another shell session.



The PVS does not save any learned network data. Therefore, the vulnerability data that is collected is not persistent. If the PVS is restarted, it will re-learn the underlying network very quickly.

Starting the Passive Vulnerability Scanner Manually as a Background Process



Running the PVS command without specifying the full path will result in the host launching a tool that displays information about local physical volumes, and not Tenable's Passive Vulnerability Scanner. Ensure that you include the full path to Tenable's PVS as shown below.

It may be preferable to start the PVS as a background process. To do this, type the following command:

```
# /opt/pvs/bin/pvs -c /opt/pvs/etc/pvs.conf -r report.nsr &
```

This will still print out some messages to the screen, but once the PVS has started, it is silent except for updating the report file. If you do not want to see these messages, invoke it with the following command:

```
# /opt/pvs/bin/pvs -c /opt/pvs/etc/pvs.conf -r report.nsr &> /dev/null 2>&1 &
```

The current content of the reports will be written to disk in the file **report.nsr** every sixty minutes. If a full path is not specified, report.nsr will be created in the directory from which

Copyright © 2002-2012 Tenable Network Security, Inc.

the **pvs** command was run. For example, if you were in **/home/userx** when you issued the above command, the report file would be located in **/home/userx/report.nsr**. If you want the **report.nsr** file to be created in a specific directory, specify the full path, such as "**-r /home/jsmith/myreport.nsr**".

The PVS Watchdog

The PVS Proxy uses a watchdog service to check the running state of the PVS and to restart the PVS service if it is detected to be down at any point. These settings determine the number of restart attempts and the interval at which PVS attempts to start. Once the maximum number of attempts has been made, the proxy will abandon restart attempts for a period of 30 minutes.

The settings for this watchdog are configured in **pvs-proxy.conf**:

```
watchdog {
          # How many times should the proxy attempt to start
          # PVS over a configurable interval. Once the maximum
          # number of attempts has been made, the proxy will
          # abandon restart attempts for a period of 30 minutes.
          # 'pvs-interval' is measured in minutes.
          pvs-restart-attempts 3;
          pvs-restart-interval 10;
          }
```

Using the Passive Vulnerability Scanner to Analyze TCPDUMP Capture Files

The PVS can read in TCP packets from a TCPDUMP binary packet capture file. Specify the file to read with the "**-f**" option, and the PVS will analyze the contents of the file and then produce a report specified with the "**-r**" option.

Generating a Report

The installation script will install the PVS with a default reporting cycle of 60 minutes. This means that the PVS will overwrite the report file with the latest vulnerability information every 60 minutes. In our previous example, the "**-r**" option was used to specify a report file of **report.nsr**. This report file is a compatible format with the Nessus vulnerability scanner. Examples of manipulating the report are provided in "Appendix 3 Working with Nessus".

By default, simply waiting 60 minutes will cause the PVS to produce a report. However, sending a "**-1**" signal to the PVS's process ID will cause it to immediately produce a report. In this example, the PVS's process ID was determined to be 19211.

```
# ls -lt pvs-report.nsr
-rw-r--r--    1 root     root         4153 Aug 27 10:25 pvs-report.nsr
# ps aux | grep pvs
root 19211 1.4 17.2 21772 16296 pts/0S 10:41 0:01 /opt/pvs/bin/pvs
# date
Wed Aug 27 10:43:26 EDT 2003
# kill -1 19211
# ls -lt pvs-report.nsr
-rw-r--r--    1 root     root         4153 Aug 27 10:43 pvs-report.nsr
```

Notice that the report file **pvs-report.nsr** has a time stamp of the current time. When the PVS receives the "**-1**" signal, it immediately produces a report.

Copyright © 2002-2012 Tenable Network Security, Inc.



On high-speed networks with more than 20,000 systems, if the system running the PVS is CPU bound, it may take up to 30 seconds to generate the report.

Downloading New Vulnerability Plugins

To manually update the PVS plugins, run **`/opt/pvs/bin/pvs-update-plugins.sh`**. This will update the plugins located in **`/opt/pvs/var/pvs/plugins`**.

If the SecurityCenter is being used to manage a PVS, new plugins for the PVS will expect to be sent to **`/opt/pvs/var/pvs/plugins`** and the PVS Proxy will restart the PVS. SecurityCenter will fail to send plugins if this directory is not configured for use by by PVS.

CONFIGURATION

pvs.conf File Format

The **`pvs.conf`** file, located in **`/opt/pvs/etc`**, is split into two sections: "options" and "networks". The "options" section specifies the operating parameters of the PVS and the "networks" section specifies which networks the PVS is to focus on. An example Unix **`pvs.conf`** file is shown in "Appendix 1". Any file that is specified in the **`pvs.conf`** file must be enclosed in quotes. The following table lists the available options that can be configured:

Name	Description
`report-threshold`	PVS reports new ports, applications, or vulnerabilities after they are detected. The **`report-threshold`** variable specifies how many times a vulnerability will be reported before PVS stops reporting those items as it detects them. When adding new port, application, or vulnerability information to the PVS's model of the observed network, this threshold can be used to limit false positives and stray ports that open and close quickly. For example, during an FTP file transfer, a client may temporarily open a port. However, with the report threshold variable, a vulnerability will not be reported until it has occurred a specified number of times. This variable has a default of "3".
`nsr-report-file`	Specifies the location for PVS to generate the report in NSR format. If PVS is started with the **`-r`** option, it will override this configuration option.  Only change this setting in the **`pvs.conf`** file from **`/opt/pvs/var/pvs-proxy/scans/pvs-report.nsr`** if SecurityCenter is not being used.
`xml-report-file`	Specifies the location for PVS to generate the report in XML format.
`nessus-report-file`	Specifies the location for PVS to generate the report in **`.nessus`** format.

Copyright © 2002-2012 Tenable Network Security, Inc.

nessus-report-version	Specifies the Nessus report file version to save the file. The default is "2".
log-directory	The **log-directory** keyword specifies a location for the **pvs** binary to write system logs. Log files of the format "**YYYYMM.log**" (e.g., **200502.log**) will be created. These detail PVS operation and can be useful for debugging.
interface	The **interface** keyword specifies the network device to use for sniffing packets. Examples include: eth0 and fxp1. There is no default for this and it is set by the PVS installation script.
key-file	Specifies the PVS license key file. The license key can be any file name in any directory. The default **key-file** is **/opt/pvs/var/pvs/tenable.key**.
disabled-plugins	This keyword points to a file that contains a list of PVS IDs that should be disabled. The file contains a list of IDs, one per line, followed by a new-line. If the file exists, the PVS will disable the specified plugins when it starts.  Plugins 0-14 may not be disabled and will be ignored if added to the disabled plugins list. Plugins provided by Tenable will be numbered from 13-6999.
plugins-directory	This keyword specifies the directory containing the passive vulnerability plugins used by the PVS. By default, the plugins are located in **/opt/pvs/var/pvs/plugins**. SecurityCenter will fail to send plugins if this directory is not configured for use by by PVS.
scripts-directory	Points to the location of PASL scripts used by PVS. By default, the scripts are located in **/opt/pvs/var/pvs/scripts**.
disabled-scripts	By default, all PASL scripts present in either the **.prmx** feed or in the **scripts-directory** will be executed. The **disabled-scripts** keyword specifies a file containing names of PASL scripts that should be disabled.  Scripts provided by Tenable will be numbered from 7000-7999.
realtime-plugins	Points to a file that specifies a list of plugins that should be run in real-time mode.
max-packet-cache-size	The **max-packet-cache-size** keyword specifies the maximum size in megabytes of the cache that will be used to store the contents of the packets collected. By default it is set to 1024 MB. When the cache is full, any subsequent packets captured will be dropped until space in the cache

Copyright © 2002-2012 Tenable Network Security, Inc.

	becomes available. The size of the cache will change dynamically and can be expected to grow to this maximum number only when needed.
memory	When reconstructing network sessions, the PVS will pre-allocate as many megabytes of memory as specified by this variable. By default, the PVS is installed with a **memory** value of "50" megabytes. Networks with sustained speeds larger than 100 Mb/s or more than 5,000 unique IP addresses should modify this value to "100" MB. For customers running in front of multiple Class B networks, values of "400" MB should be used if the system has enough spare memory. However, if you have a large network (such as a university network with 10,000 nodes or more) a setting of 500 should be used. In addition to the session table, the PVS also will use another 200 to 300 MB to store the host vulnerability information and port-scan information.
report-lifetime	Reports can be cached for a specified number of days. After the configured day count is met, the PVS's entire model of a discovered network is completely removed. The PVS starts over again learning about the hosts that are involved on the network. This value can be set extremely high, such as 365 days, if this behavior is not desired. However, it is very useful to have fresh reports on a weekly or monthly basis. The default value is 30 days.
save-knowledge-base	When this option is enabled, the PVS knowledge base will be saved on disk for recovery after the program is restarted. The KB is stored in **/opt/pvs/var/pvs/kb**.
kb-max-age	The maximum length of time in seconds that a knowledge-base entry remains valid after its addition.
report-frequency	This variable specifies in minutes (default 60) how often the PVS will write a report. The PVS can be configured to write its current model of the network into various file formats including HTML, XML, "**.nsr**", and "**.nessus**". If the PVS is being managed by a SecurityCenter, SC will retrieve the PVS report every 12 hours by default.
detect-encryption	This keyword block specifies a set of "**dependency**" and "**exclude**" statements that the PVS uses to analyze sessions containing encrypted traffic. The **dependency** keywords identify the specific PVS IDs that have been detected on a host before an analysis of a session occurs. The **exclude** keyword specifies a list of protocol filters (please refer to "Appendix 1") for which the PVS should avoid performing encryption detection. When an encrypted session is detected, an alert is generated showing source, destination, ports, and session type. The session type may be one of the following: > internal-interactive-session (4)

Copyright © 2002-2012 Tenable Network Security, Inc.

	> outbound-interactive-session (5) > inbound-interactive-session (6) > internal-encrypted-session (7) > outbound-encrypted-session (8) > inbound-encrypted-session (9) The number in parentheses represents the corresponding plugin ID field.
detect-interactive-sessions	This keyword block specifies a set of "**dependency**" and "**exclude**" statements that the PVS uses to analyze sessions that contain interactive traffic. The **dependency** keywords identify the specific PVS IDs that have been detected on a host before an analysis of a session occurs. The **exclude** keyword specifies a list of protocol filters (please refer to "Appendix 1") for which the PVS should avoid performing interactive detection. When an encrypted session is detected, an alert is generated showing source, destination, ports, and session type. For a list of session types, refer to the **detect-encryption** option above.
high-speed	The PVS is designed to expect to find various protocols on non-standard ports. For example, the PVS can easily find an Apache server running on a port other than 80. However, on a high-speed network, the PVS can be placed into a "high-speed" mode that allows it to focus certain plugins on specific ports. When the **high-speed** keyword is specified, any plugin that has the keywords **hs_dport** or **hs_sport** defined will be executed only on traffic traversing the specified ports. The **high-speed** keyword takes no arguments.
strip-vlan-tags	If your network contains one or more VLANs you will need to enable or disable this setting to monitor the desired network.  With the strip VLAN tags setting enabled, PVS will only be able to process traffic that contains VLAN tags. Likewise, with the setting disabled, PVS will only be able to process traffic that does not contain VLAN tags. Multiple PVS servers will need to be deployed in mixed environments.
realtime-file	This file is used to record PVS detected real-time events with a time-stamped entry.
max-realtime-file	The **max-realtime-file** option specifies the maximum amount of data from real-time IDS events that will be stored in one text file. The option must be specified in kilobytes, megabytes, or gigabytes by appending a "K", "M", or "G" to the value. After this value is reached, a numbered file with the same base name as above will be used to store additional events. When the option is disabled, there is no limit on real-

Copyright © 2002-2012 Tenable Network Security, Inc.

	time file size.
realtime-syslog	Specifies the IP address of a SYSLOG server to receive real-time events from the PVS. Up to sixteen SYSLOG servers can be specified for alerting. A local SYSLOG daemon is not required. Multiple **realtime-syslog** keywords can be used to specific more than one SYSLOG server.
vulndata-syslog	Specifies the IP address of a SYSLOG server to receive vulnerability data from the PVS. Up to sixteen SYSLOG servers can be specified for alerting. A local SYSLOG daemon is not required.  While PVS may display multiple log events related to one connection, it will only send a single event to the remote SYSLOG server(s).
connections-to-services	When enabled, this keyword causes PVS to log which clients are attempting to connect to servers on the network and what port they are attempting to connect to. They do not indicate if the connection was successful, but only indicate that an attempt to connect was made. Events detected by the PVS of this type are logged as Nessus ID "00002".
show-connections	When enabled, PVS will record clients in the focus network that attempt to connect to a server IP address and port and receive a positive response from the server. The record will contain the client IP address, the server IP address and the server port that the client was attempting to connect to. For example, if four different hosts within the focus network attempted to connect with a server IP over port 80 and received a positive response, then a list of those hosts would be reported under event "00003" and port 80. By default, this feature is not enabled and it only reports on client IP addresses that reside within the focused networks.
new-host-alert	The PVS listens to network traffic and attempts to discover when a new host has been added. To do this, the PVS constantly compares a list of hosts that have generated traffic in the past to those currently generating traffic. If it finds a new host generating traffic, it will issue a "new host alert" via the real-time log. For large networks, the PVS can be configured to run for several days to gain knowledge about which hosts are active. This prevents the PVS from issuing an alert for hosts that already exist. The number of days the PVS should monitor traffic to learn which hosts are active is specified by this setting. For large networks, Tenable recommends that the PVS operate for at least one day before detecting new hosts.
backup-interval	The PVS constantly compares its list of active hosts to the list of hosts generating traffic to discover newly added or missing

Copyright © 2002-2012 Tenable Network Security, Inc.

	hosts. To prevent rediscovery of the entire network, the PVS can frequently write the list of active hosts to a file so that the information is available to PVS across restarts. Tenable recommends that this file be updated every 120 minutes.
Backup-file	The location on the disk to save the backup file.
networks	Specifies the networks to be monitored. This is set by the PVS installation script in Unix.
excluded-networks	Specifies any networks that should be excluded from PVS monitoring. Networks should be specified using CIDR notation and placed between the brackets after this directive. If left blank, no addresses will be excluded.

Specifying Focus Networks

The "**networks**" keyword is used to indicate which networks the PVS will focus on while performing passive vulnerability analysis. While the PVS runs, it evaluates a network session only if one of the IP addresses is within a list of specific networks. These networks are known to the PVS as the "focus networks". Without the specification of a list of networks for the PVS to monitor, it will monitor the entire Internet and report not only on your vulnerabilities, but on the vulnerabilities of systems communicating with your network. Tenable recommends configuring your PVS to focus only on networks you are responsible for.

The "**networks**" keyword can specify hosts or networks using CIDR notation. Here are several examples of valid entries:

- 10.10.10.22
- 10.10.10.22/32
- 10.10.10.0/24
- 10.163.155.0/255.255.255.0
- 0.0.0.0/0

The last example is used to conduct passive vulnerability analysis on every network session and host observed. For busy networks with large numbers of unique visitors and active IP addresses, the PVS will consume large amounts of memory and produce very large reports. Care should be used when monitoring an active network without a specific focus address.



Tenable has deployed PVS sensors in front of popular web sites and performed passive vulnerability analysis on more than 75,000 unique IP addresses at one time. The PVS is much more efficient if it is focused on one particular range of addresses.

Using Multiple Interfaces

The PVS can use multiple interfaces. To specify this, simply use multiple lines to specify multiple interfaces. For example, the **pvs.conf** file shown below will cause the PVS to look at both the eth0 and eth1 interfaces:

Copyright © 2002-2012 Tenable Network Security, Inc.

```
options {
report-threshold 3;
failure-threshold 10;
interface "eth0";
interface "eth1";
key-file "/opt/pvs/var/tenable.key";
plugins-directory "/opt/pvs/var/pvs/plugins";
fingerprints "/opt/pvs/var/pvs/osfingerprints.txt";
memory 50;
report-lifetime 30;
report-frequency 60;
}
networks {
192.168.0.0/24;
}
```

REMOVING PVS

Red Hat

To remove PVS for Red Hat versions, you need to know what name it is registered as within the RPM database. This name will not be the same as the filename used for installation. To determine what name the PVS RPM is registered as, run the following command:

```
# rpm -qa | grep pvs
```

This will produce output similar to the following:

```
pvs-3.6.0-1.x86_64
```

For this example, the command to remove the PVS RPM would be:

```
# rpm -e pvs-3.6.0-1.x86_64
```

This will not remove configuration files or files that were not part of the original installation. Files that were part of the original package but have changed since installation also will not be removed. To completely remove the remaining files, run the following command:

```
# rm –rf /opt/pvs
```

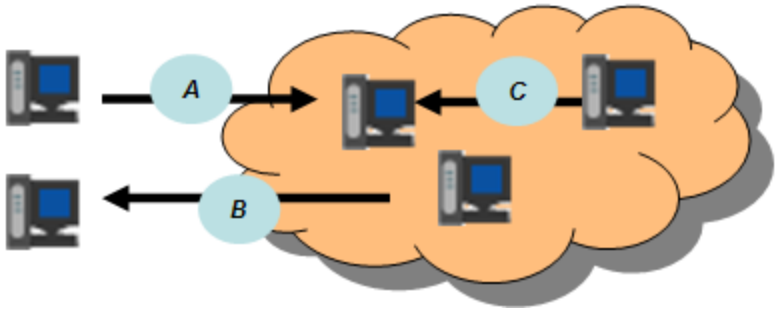
PVS REAL-TIME TRAFFIC ANALYSIS CONFIGURATION THEORY

This section describes how configuration options affect PVS operation and provides details on PVS architecture.

FOCUS NETWORK

When a focus network is specified via the "networks" keyword, only one side of a session needs to be matched on the list. For example, if you have a DMZ that is part of the focus network list, the PVS will report on vulnerabilities of the web server there, but not on web clients visiting from outside the network. However, a web browser within the DMZ visiting the same web server would be reported.

Copyright © 2002-2012 Tenable Network Security, Inc.



In the above picture, three sessions labeled A, B, and C are shown communicating to, from, and inside a focus network. In session A, the PVS only analyzes vulnerabilities observed on the server inside the focus network and does not report client side vulnerabilities. In session B, the PVS ignores vulnerabilities on the destination server, but reports client side vulnerabilities. In session C, both client and server vulnerabilities are reported.

There is one more filter that the PVS uses while looking for unique sessions. This is a dependency that requires the host to be running a major service. These dependencies are defined by a list of PVS plugin IDs that identify SSL, FTP and several dozen other services.

Finally, the entire process of detecting these sessions can be filtered by specific network ranges and ports. For example, if a University ran a public FTP server that had thousands of downloads each hour, it would make sense to disable interactive sessions on port 21 on that FTP server. Similarly, disabling encryption detection on ports such as 22 and 443 will also eliminate some noise for the PVS.

Detecting Server and Client Ports

The method used by TCP connections to initiate communication is known as the "three-way handshake". This method can be compared to how a common telephone conversation is initiated. If Bob calls Alice, he has effectively sent her a "SYN" packet, in TCP terms. She may or may not answer. If Alice answers, she has effectively sent a "SYN-ACK" packet. The communication is still not established, since Bob may have hung up as she was answering. The communication is established when Bob replies to Alice, sending her an "ACK".

The PVS configuration interface (**`pvs.conf`** file) enables PVS to log network client activity. On Unix systems, this is accomplished via the "**`connections-to-services`**" keyword in the **`pvs.conf`** file.

Whenever a system within the monitored network range tries to connect to a server over TCP, the connecting system will emit a TCP "SYN" packet. If the port the client is connecting to is open, then the server will respond with a TCP "SYN/ACK" packet. At this point, PVS will record both the client address and the server port the client is connecting to. If the port on the server is not open, then the server will not respond with a TCP "SYN/ACK" packet. In this case, since PVS never sees a TCP "SYN/ACK" response from the server, PVS will not record the fact that the client tried to connect to the server port, since the port is not available to that client.

The Unix "**`connections-to-services`**" option does not track how many times the connection was made. If the same host browses the same web server a million times, or browses a million different web servers once, the host will still be marked as having browsed on port 80. This data is logged as Nessus ID #00002.

Copyright © 2002-2012 Tenable Network Security, Inc.



By default, these settings are disabled and must be manually edited in the **pvs.conf** file.

The PVS detects many applications through plugin and protocol analysis. At a lower level, the PVS also detects open ports and outbound ports in use on the monitored networks. By default, the PVS will detect any TCP server on the protected network if it sees a TCP "SYN-ACK" packet.

In combination, the detection of server ports and client destination ports allows a network administrator to see who on their network is serving a particular protocol and who on their network is speaking that protocol.

Detecting Specific Server and Client Port Usage

Another PVS configuration option provides more specific details about server and client port usage. On Unix systems, this is the "**show-connections**" keyword in the **pvs.conf** file. This setting keeps track of host communication within the focus network. When the "**show-connections**" option is enabled, every time a host connects to another host, PVS records the client, server, and server port, if one of the hosts is in the defined focus network. It does not track the frequency or time stamp of the connections – just that a connection was made.

The "**show-connections**" option provides a greater level of detail than the "**connections-to-services**" option. For example, if your IP address is 1.1.1.1 and you use the SSH service to connect to "some_company.com", use of these options would record the following:

```
show-connections:
1.1.1.1  → some_company.com:SSH

connections-to-services
1.1.1.1  → SSH
```

Using the "**connections-to-services**" option lets you know that the system at 1.1.1.1 uses the SSH protocol. This information may be useful to know regardless of where the service is being used.

The PVS does not log a session-by-session list of communications. Instead, it logs the relationship between the systems. For example, if system A is detected using the SSH protocol on port 22 connecting to system B, and both systems are within the focus network, the PVS would log:

- System A browses on port 22
- System B offers a service (listens) on port 22
- System A communicates with System B on port 22

If system B were outside of the focus network, the PVS would not record anything about the service System B offers, and would also log that System A browses outside of the focus network on port 22. The PVS does not log how often a connection occurs, only that it

Copyright © 2002-2012 Tenable Network Security, Inc.

occurred at least once. For connections outside of the focus network, the PVS will only log what ports are browsed, not the actual destinations.



If logging session-by-session network events is a requirement for your network analysis, Tenable offers the Log Correlation Engine product, which can be used to log firewall, web server, router, and sniffer logs. For more information, please visit http://www.tenable.com/products/lce/.

What this Means for Firewall Rules

If the PVS is placed immediately behind a firewall, such that all of the traffic presented to the PVS is flowing through the firewall, then the list of served ports and client side ports and the respective IP addresses of the users is readily available. By using tools such as the SecurityCenter's "browse vulnerabilities" interface, information about these ports (both client and server) can be browsed, sorted, and reported on. Lists of IP addresses and networks using these client and server ports can also be viewed.

Working with the SecurityCenter

When multiple PVS sensors are managed by a SecurityCenter, users of the SecurityCenter are able to analyze the aggregate types of open ports, browsed ports, and communication activity occurring on the focus network. Since the SecurityCenter has several different types of users and privileges, many different IT and network engineering accounts can be created across an enterprise so they can share and benefit from the information detected by the PVS.

Selecting Rule Libraries and Filtering Rules

Tenable ships the PVS with two distinct sets of rule libraries.

The first is a library of passive vulnerability detection rules. This file is encrypted and cannot be modified by the end users of the PVS. However, if certain vulnerability rules should be disabled, they can be specified through the PVS configuration file, `pvs.conf` for Unix systems.

The second set of rules are the PVS real-time plugins. With these plugins enabled, the PVS can look for a wide variety of successful application and system compromises. They are not encrypted and can be modified by PVS users for customization.



If the PVS is being managed by the SecurityCenter, it will automatically update the libraries shipped. In this case, any changes to PVS plugins should be made by disabling specific plugins or by creating new libraries to augment the plugin set delivered by Tenable.

Detecting Encrypted and Interactive Sessions

The PVS can be configured to detect both encrypted and interactive sessions. An encrypted session is a TCP or UDP session that contains sufficiently random payloads. An interactive session uses timing and statistical profiling of the packets in a session to determine if the session involves a human typing at a command line prompt.

In both cases, the PVS will identify these sessions for the given port and IP protocol. It will

Copyright © 2002-2012 Tenable Network Security, Inc.

then list the detected interactive or encrypted session as a vulnerability.

The PVS has a variety of plugins to recognize telnet, Secure Shell (SSH), Secure Socket Layer and other protocols. In combination with the detection of the interactive and encryption algorithms, it is likely that the PVS will log multiple forms of identification for the detected sessions.

For example, with a SSH service running on a high port, it is likely that the PVS would not only recognize this as an encrypted session, it would also recognize the version of SSH and determine if there were any vulnerabilities associated with it.

Routes and Hop Distance

For active scans, one host can find the default route and an actual list of all routers between it and a target platform. To do this, it sends one packet after another with a slightly larger TTL (time to live) value. Each time a router receives a packet, it decrements the TTL value and sends it on. If a router receives a packet with a TTL value of one, it sends a message back to the originating server that the TTL has expired. The server simply sends packets to the target host with greater and greater TTL values, and collects the IP addresses of the routers in-between when they send their expiration messages.

Since the PVS is entirely passive, it cannot send or elicit packets from the routers or target computers. It can however, record the TTL value of a target machine. The TTL value is an 8-bit field, meaning it can contain a value between 0 and 255. Most machines use an initial TTL value of 32, 64, 128, or 255. Since there is a maximum of 16 hops between your host and any other host on the internet, it is a simple algorithm that the PVS uses to map any TTL to the number of hops.

For example, if the PVS sniffed a server sending a packet with a TTL of 126, this is closest to 128 and two hops away. The PVS does not know the IP address of the in-between routers.



Modern networks have many devices such as NAT firewalls, proxies, load balances, intrusion prevention, routers, and VPNs that will rewrite or reset the TTL value. In these cases, the PVS can report some very odd hop counts.

Alerting

When the PVS detects a real-time event, it can send the event to a local log file or send it via SYSLOG to a log aggregator such as Tenable's Log Correlation Engine and the SecurityCenter, as well as internal log aggregation servers and third party security event management vendors.

New Host Alerting

The PVS can be configured to detect when a new host has been added to the network. This is not as simple as it sounds, and several parameters can be configured within the PVS to increase or decrease the accuracy of detecting true change.

Initially, the PVS has no knowledge of your network's active hosts. The first packets that the PVS sniffs would send an alert. To avoid this, the PVS can be configured to learn the network over a period of days. Once this period is over, any "new" traffic would be from a host that has not communicated during the initial training.

Copyright © 2002-2012 Tenable Network Security, Inc.

To prevent the PVS from having to relearn the network each time it starts, a file can be specified to save the active host information. This file contains a list of all the current active hosts for the PVS. The scanner also requires that an interval to update this file be specified. Tenable recommends an update time of at least one day (1440 minutes).



When the PVS logs a new host, the Ethernet address is saved in the message. When the PVS is more than one hop away from the sniffed traffic, the Ethernet address will be that of the local switch, not the actual host. If the scanner is deployed in the same collision domain as the sniffed server, the Ethernet address will be accurate.



For DHCP networks, the PVS will detect a "new" host very often. Tenable recommends deploying this feature on non-volatile networks such as demilitarized zone (DMZ). Users should also consider analyzing PVS "new" host alerts with Tenable's SecurityCenter, which can sort real-time PVS events by networks.

INTERNAL PASSIVE VULNERABILITY SCANNER IDS

WHAT IS A PASSIVE VULNERABILITY SCANNER ID?

This section describes the PVS's advanced signature language for each plugin. Each vulnerability and real-time check that the PVS performs has a unique associated ID. Since Tenable manages the Nessus vulnerability scanner, we have added the IDs used by the PVS into the overall Nessus architecture. PVS IDs start from #00000 and go through #10000. Nessus IDs start from #10001 and extend upward.

INTERNAL PASSIVE VULNERABILITY SCANNER IDS

Some of the PVS's checks, such as detecting open ports, are built in. The following chart lists each of the internal checks and describes what they mean:

PVS ID	Name	Description
00000	Detection of open port	The PVS has observed a SYN-ACK leave from a server.
00001	Passive OS Fingerprint	The PVS has observed enough traffic about a server to perform a guess of the operating system.
00002	Client Side Port Usage	The PVS has observed browsing traffic from a host.
00003	Show Connections	The PVS has logged a unique network session of source IP, destination IP and destination port.
00004	Internal Interactive Sessions	The PVS has detected one or more interactive network sessions between two hosts within your focus network.
00005	Outbound Interactive Sessions	The PVS has detected one or more interactive network sessions originating from within your focus network and

Copyright © 2002-2012 Tenable Network Security, Inc.

		destined for one or more addresses on the Internet.
00006	Inbound Interactive Sessions	The PVS has detected one or more interactive network sessions originating from one or more addresses on the Internet to this address within your focus network.
00007	Internal Encrypted Session	The PVS has detected one or more encrypted network sessions between two hosts within your focus network.
00008	Outbound Encrypted Session	The PVS has detected one or more encrypted network sessions originating from within your focus network and destined for one or more addresses on the Internet.
00009	Inbound Encrypted Session	The PVS has detected one or more encrypted network sessions originating from one or more addresses on the Internet to this address within your focus network.
00012	Host TTL Discovered	The PVS logs the number of hops away each host is located.

WORKING WITH PASSIVE VULNERABILITY SCANNER PLUGINS

VULNERABILITY AND PASSIVE FINGERPRINT OVERVIEW

The PVS has two sources of "plugin" information: the **.prmx** and **.prm** plugin libraries in the **plugins** directory and the operating system fingerprints in the **osfingerprints.txt** file.

Tenable distributes its passive vulnerability plugin database in an encrypted format. This file is known as **tenable_plugins.prmx** and can be updated on a daily basis, if necessary. PVS plugins that are written by the customer or third parties have the extension of **.prm**.

Tenable has also implemented passive fingerprinting technology based on the open-source **SinFP** tool. With permission from the author, Tenable has also included the database of passive operating system fingerprints for the fingerprinting technology in this distribution of the PVS. New passive operating system fingerprints can be added by PVS users through by manually editing the **os_fingerprint_sinfp.pasl** file and restarting the PVS.

Downloading New Vulnerability Plugins

Tenable ships the PVS with a utility that downloads the most recent plugins. To manually update the PVS plugins on Unix systems, run **/opt/pvs/bin/pvs-update-plugins.sh**. This will update the plugins, located in **/opt/pvs/var/pvs/plugins**. If the SecurityCenter is being used to manage a PVS, new plugins for the PVS will automatically be sent and the PVS Proxy will restart the PVS.

Writing Custom Plugin Libraries

PVS customers can write their own passive plugin libraries. These plugins are added into the **plugins** directory in the PVS's installation directory. The plugin library must end with a **.prm** extension for the PVS to see it. The next section details exactly how to write PVS plugins.

Copyright © 2002-2012 Tenable Network Security, Inc.

Restarting the Passive Vulnerability Scanner

Once new passive plugins or operating system fingerprints are available to the PVS, it must be stopped and started again.

WRITING PASSIVE VULNERABILITY SCANNER PLUGINS

Plugin Keywords

There are several keywords available for writing passive vulnerability plugins for PVS. Some of these keywords are mandatory and some are optional. The mandatory keywords are highlighted in blue.

Name	Description
bid	Tenable assigns SecurityFocus Bugtraq IDs (BID) to PVS plugins. This allows a user reading a report generated by the PVS to link to more information available at http://www.securityfocus.com/bid. Multiple Bugtraq entries can be entered on one line separated by commas.
bmatch	This is the same as "**match**" but can look for any type of data.
clientissue	If a vulnerability is determined in a network client such as a web browser or an email tool, a server "**port**" will be associated with the reported vulnerability.
cve	Tenable also assigns Common Vulnerability and Exposure (CVE) tags to each PVS plugin. This allows a user reading a report generated by the PVS to link to more information available at http://cve.mitre.org/. Multiple CVE entries can be entered on one line separated by commas.
dependency	This is the opposite of "**noplugin**". Instead of specifying another plugin that has failed, this keyword specifies which plugin has to have succeeded. This keyword specifies a PVS ID that should exist in order for the plugin to be evaluated. In addition, this plugin can take the form of "**dependency=ephemeral-server-port**", which means that the server being evaluated must have an open port above port 1024.
description	This field describes on one line the nature of the detected vulnerability. This data is printed out by the PVS when printing the vulnerability report. Macros are available that allow for the printing of matched network traffic such as banner information and are discussed in the examples below. For line breaks, the characters " " can be used to invoke a new line.
Exploitability: **canvas** **core**	Displays exploitability factors for the selected vulnerability. For example, if the vulnerability is exploitable via both Canvas and Core and has a unique CVSS temporal score, the following tags might be displayed in the plugin output:

Copyright © 2002-2012 Tenable Network Security, Inc.

cvsstemporal metasploit	CANVAS : D2ExploitPack CORE : true CVSSTEMPORAL : CVSS2#E:F/RL:OF/RC:C  These keywords are displayed only in vulnerabilities detected by PVS 3.4 and greater.
dport	Same as "**sport**", but for destination ports.
family	Each Tenable plugin for the PVS is included in a family. This designation allows Tenable to group PVS plugins into easily managed sets that can be reported on individually.
hs_dport	Same as "**hs_sport**" except for destination ports.
hs_sport	Normally, when the PVS runs its plugins, they are either free ranging looking for matches on any port, or fixed to specific ports with the "**sport**" or "**dport**" keywords. In very high speed networks, many plugins have a fall-back port, known as a high-speed port, which focuses the plugin only on one specific port. In high speed more, the performance of a PVS plugin with an "**hs_sport**" keyword is exactly the same as if the plugin was written with the "**sport**" keyword.
id	Each PVS plugin needs a unique rule ID. Tenable assigns these 16 bit numbers within the overall Nessus range of valid entries. Current plugin IDs can be listed at Tenable's website for the PVS.
match	This keyword specifies a set of one or more simple ASCII patterns that must be present in order for the more complex pattern analysis to take place. The "**match**" keyword gives the PVS a lot of its performance and functionality. With this keyword, if it does not see a simple pattern, the entire plugin will not match.
max_syslog	This option will affect the output sent to servers specified with **realtime-syslog**, as well as the data in the file specified with **realtime-file**. By specifying a maximum number of bytes, the syslog message generated from the current sniffed packet will be truncated accordingly.
name	This is the name of the vulnerability the PVS has detected. Multiple PVS plugins can have the same name, but this is not encouraged.
nid	To track compatibility with the Nessus vulnerability scanner, Tenable has attempted to associate PVS vulnerability checks with relevant Nessus vulnerability checks. Multiple Nessus IDs can be listed under one "**nid**" entry such as "**nid=10222,10223**".

Copyright © 2002-2012 Tenable Network Security, Inc.

nooutput	For plugins that are written specifically to be used as part of a dependency with another plugin, the "**nooutput**" keyword will cause the PVS to not report anything for any plugin with this keyword enabled.
noplugin	This keyword will prevent a plugin from being evaluated if another plugin has already matched. For example, it may make sense to write a plugin that looks for a specific anonymous FTP vulnerability, but have it disabled if another plugin that checked for anonymous FTP had already failed.
pbmatch	Same as "**bmatch**" except for binary data on the previous side of the reconstructed network session.
pmatch	This keyword is the same as "**match**" but is applied against the previous packet on the other side of the reconstructed network session.
pmax_syslog	This option will affect the output sent to servers specified with **realtime-syslog**, as well as the data in the file specified with **realtime-file**. By specifying a maximum number of bytes, the syslog message generated from the **previous** sniffed packet will be truncated accordingly.
pregex	Same as "**regex**" except the regular expression is applied to the previous side of the reconstructed network session.
pregexi	Same as "**pregex**" except the pattern matching is case insensitive.
psyslog_cutoff	This option will affect the output sent to servers specified with **realtime-syslog**, as well as the data in the file specified with **realtime-file**. By specifying a token (e.g., "required"), the syslog message generated from the **previous** sniffed packet will be truncated from the token on.
regex	This keyword specifies a complex regular expression search rule that will be applied to the network session.
regexi	Same as "**regex**" except the pattern matching is case insensitive.
risk	All PVS plugins need a risk setting. Risks are classified as LOW, MEDIUM, or HIGH. A LOW risk is an informational vulnerability such as an active port or service. A MEDIUM risk is something that may be exploitable or discloses information and a HIGH risk is something that is easily exploitable.
rst-packet	When a specific plugin is matched, the PVS can send a TCP reset packet to the source IP address of the matching traffic. This can help organizations apply basic content and application filtering to their networks.
rst-packet-full	Same as "**rst-packet**" except that the TCP reset packets are sent to both the source IP and destination IP addresses.

Copyright © 2002-2012 Tenable Network Security, Inc.

seealso	If one or more URLs are available, this keyword can be used to display them. Multiple URLs can be specified on one line with commas. Example entries for this could include CERT advisories and vendor information web sites. Note: PVS 3.0.x will only display the last **seealso** defined in the PRM. PVS 3.2 and later will display multiple **seealso** directives.
solution	If a solution is available, it can be described here. The report section will highlight the solution with different text.
sport	This setting applies the PVS plugin to just one port. For example, it may make sense to write a SNMP plugin that just looks for activity on port 162. However, for detection of off-port services like a web server running on port 8080, a "**sport**" field would not be used in the plugin.
syslog_cutoff	This option will affect the output sent to servers specified with **realtime-syslog**, as well as the data in the file specified with **realtime-file**. By specifying a token (e.g., "required"), the syslog message generated from the sniffed packet will be truncated from the token on.
timed-dependency	With this keyword, the functionality of the "**noplugin**" and "**dependency**" keywords is slightly modified such that the evaluation must have occurred within the last "n" seconds.
udp	All plugins are assumed to be based on the TCP protocol unless this keyword is specified.



In addition to tcp or udp, the following protocols are supported: sctp, icmp, igmp, ipip, egp, pup, idp, tp, rsvp, gre, pim, esp, ah, mtp, encap, pim, comp, raw or other.



The **max_syslog**, **pmax_syslog**, **syslog_cutoff**, and **psyslog_cutoff** directives can be used in any combination. If multiple limits are reached, the smallest limit will take effect.

Plugin Libraries

When writing PVS plugins in a **.prm** library, spaces are allowed, as are comment fields that start with a number sign. Each plugin must be separated with the word "NEXT" on a single line.

Simply creating a **.prm** file in the **plugins** directory will make it available for use. PVS must be restarted for the new plugins to be used.

Basic Passive Vulnerability Scanner Example

This plugin illustrates the basic concepts of PVS plugin writing:

```
id=1001
nid=11414
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
hs_sport=143
name=IMAP Banner
description=An IMAP server is running on this port. Its banner is :<br> %L
risk=NONE
match=OK
match=IMAP
match=server ready
regex=^.*OK.*IMAP.*server ready
```

In this example, the following fields are used:

- **id** is a unique number assigned to this plugin
- **nid** is the Nessus ID of the corresponding Nessus NASL script
- **hs_sport** is the source port to key on if we have the high-speed mode enabled
- **name** is the name of the plugin
- **description** is a description of the problem or service
- **match** is the set of match patterns we must find in the payload of the packet before we evaluate the regular expression
- **regex** is the regular expression to apply to the packet payload

Notice that the description contains the %L macro. If this plugin evaluates successfully then the string pattern in the payload that matched the regular expression is stored in %L and is printed out at report time.

More Complex Passive Vulnerability Scanner Example

```
id=1004
nid=10382
cve=CVE-2000-0318
bid=1144
hs_sport=143
name=Atrium Mercur Mailserver
description=The remote imap server is Mercur Mailserver 3.20. There is a
       flaw in this server (present up to version 3.20.02) which allow any
       authenticated user to read any file on the system. This includes
       other users mailboxes, or any system file. Warning : this flaw has
       not been actually checked but was deduced from the server banner
solution=There was no solution ready when this vulnerability was written;
       Please contact the vendor for updates that address this
       vulnerability.
risk=HIGH
match=>* OK
match=MERCUR
match=IMAP4-Server
regex=^\* OK.*MERCUR IMAP4-Server.*v3\.20\..*$
```

Notice that the first match pattern makes use of the ">" symbol. The ">" symbol indicates that the subsequent string must be at the beginning of the packet payload. Use of the ">" symbol is encouraged where possible as it is an inexpensive operation.

Copyright © 2002-2012 Tenable Network Security, Inc.

Case Insensitive Example

There is a tool called SmartDownLoader that uploads and downloads large files. Unfortunately, versions 0.1 through 1.3 use the syntax "SmartDownloader", versions 1.4 through 2.7 use "smartdownloader" and versions 2.8 through current uses the syntax "SMARTdownloader". Searching for the various combinations of this text with purely the **regex** command would cause us to use a statement that looks like this:

```
regex=[sS][mM][aA][rR][tT][dD]own[lL]oader
```

However, with the **regexi** command, the search string is much less complex and less prone to creating an error:

```
regexi=smartdownloader
```

By using **regexi**, we can more quickly match on all three versions as well as future permutations of the string "smartdownloader". In a case such as this, **regexi** is the logical choice.

```
id=8800
dependency=1442
hs_sport=6789
name=SmartDownLoader Detection
description=The remote host is running SmartDownLoader, a tool for
       performing rudimentary uploads and downloads of large binary files.
solution=Ensure that this application is in keeping with Corporate
       policies and guidelines
risk=MEDIUM
family=PeerToPeer
match=ownloader
regexi=smartdownloader
```

A complete example PVS plugin using the **regexi** keyword is shown above. The use of the **match** keyword searching for the string "ownloader" is not a typo. By searching for network sessions that have this string in them first, the PVS can avoid invoking the expensive **regexi** search algorithm unless the "ownloader" pattern is present.

Passive Vulnerability Scanner Network Client Detection

```
id=1010
hs_dport=25
clientissue
name=Buffer overflow in multiple IMAP clients
description=The remote e-mail client is Mozilla 1.3 or 1.4a which is
       vulnerable to a boundary condition error whereby a malicious IMAP
       server may be able to crash or execute code on the client.
solution=Upgrade to either 1.3.1 or 1.4a
risk=HIGH
match=^From:
match=^To:
match=^Date:
match=^User-Agent: Mozilla
match=!^Received:
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
regex=^User-Agent: Mozilla/.* \(.*rv:(1\.3|1\.4a)
```

Match patterns that begin with the "^" symbol mean that at least one line in the packet payload must begin with the following pattern. Match patterns that begin with the "!" symbol indicate that the string must NOT match anything in the packet payload. In this case, the "!" and "^" symbols are combined to indicate that we should not evaluate any packet whose payload contains a line starting with the pattern "Received:".

The "^" is more expensive to evaluate than the ">" symbol. So, while both match patterns "^<pattern>" and "><pattern>" would find "<pattern>" at the beginning of a packet payload, the use of ">" is more desirable as it is less costly. Use "^" when looking for the occurrence of a string at the beginning of a line, but not at the beginning of the packet payload. In the latter case, use the ">" character instead.

The Passive Vulnerability Scanner can Match "Previous" Packets

The PVS allows matching on patterns in the current packet as well as patterns in the previous packet in the current session. This plugin shows how we can make use of this feature to determine if a Unix password file is sent by a web server:

```
id=1001
name=Password file obtained by HTTP (GET)
family=Generic
sport=80
description=It seems that a Unix password file was sent by the remote web
       server when the following request was made :<br>%P<br>We saw :
       <br>%L</br>
pmatch=>GET /
pmatch=HTTP/1.
match=root
match=daemon
match=bin
regex=root:.*:0:0:.*:.*
```

Here we see **match** patterns for a root entry in a Unix password file. We also see **pmatch** patterns that would match against a packet that makes an HTTP GET request to a web server. The **match** patterns apply the current packet in a session and the **pmatch** patterns apply to the packet that was captured immediately before the current one in the current session. To explain this visually, we are looking for occurrences of the following:

```
                    GET / HTTP/1.*
1) client     ------------------------> server:port 80

               Contents of password file:
               root:.*:0:0:.*:.*
2) client     <------------------------ server:port 80
```

Our **match** pattern would key on the contents in packet 2) and our **pmatch** pattern would key on packet 1) payload contents.

Copyright © 2002-2012 Tenable Network Security, Inc.

The Passive Vulnerability Scanner can Match Binary Data

The PVS also allows matching against binary patterns. Here is an example plugin that makes use of binary pattern matching to detect the usage of the well known community string "public" in SNMPv1 response packets (The "#" is used to denote a comment.):

```
###
# SNMPv1 response
#
# Matches on the following:
# 0x30              - ASN.1 header
# 0x02 0x01 0x00    - (integer) (byte length) (SNMP version - 1)
# 0x04 0x06 public  - (string)  (byte length) (community string - "public")
# 0xa2              - message type - RESPONSE
# 0x02 0x01 0x00    - (integer) (byte length) (error status - 0)
# 0x02 0x01 0x00    - (integer) (byte length) (error index - 0)
###
id=1001
udp
sport=161
name=SNMP public community string
description=The remote host is running an SNMPv1 server that uses a well-
       known community string - public
bmatch=>0:30
bmatch=>2:020100
bmatch=>5:04067075626c6963a2
bmatch=020100020100
```

Binary match patterns take the following form:

```
bmatch=[<>[off]:]<hex>
```

Binary match starts at <off>'th offset of the packet or at the last <offset> of the packet, depending on the use of > (start) or < (end). <hex> is a hex string we look for.

```
bmatch=<:ffffffff
```

This will match any packet whose last four bytes are set to 0xFFFFFFFF.

```
bmatch=>4:41414141
```

This will match any packet that contains the string "AAAA" (0x41414141 in hex) starting at its fourth byte.

```
bmatch=123456789ABCDEF
```

This will match any packet that contains the hex string above.

Negative Matches

PVS plugins can also be negated. Here are two examples:

```
pmatch=!pattern
pbmatch=>0:!414141
```

Copyright © 2002-2012 Tenable Network Security, Inc.

In each of these cases, the plugin would not match if the patterns contained in these "not" statements were present. For example, in the first **pmatch** statement, if the pattern "pattern" were present, then the plugin would not match. In the second statement, the binary pattern of "AAA" (the letter "A" in ASCII hex is 0x41) would match only if it were not presenting the first three characters.

Time Dependent Plugins

The last plugin example shows some more advanced features of the PVS plugin language that allows a plugin to be time dependent as well as make use of the evaluation of other plugins. The plugin shows how the PVS can detect an anonymous FTP server. The **NEXT** keyword is used to separate plugins the plugin file.

```
id=1018
nooutput
hs_sport=21
name=Anonymous FTP (login: ftp)
pmatch=^USER ftp
match=^331
NEXT #------------------------------------------------------------
id=1019
dependency=1018
timed-dependency=5
hs_sport=21
name=Anonymous FTP enabled
description=The remote FTP server has anonymous access enabled.
risk=LOW
pmatch=^PASS
match=^230
```

Since we are trying to detect an anonymous FTP server we are going to be looking for the following traffic pattern:

```
                          USER ftp
1) FTP client   ----------------------> FTP server

                 331 Guest login ok, ...
2) FTP client   <---------------------- FTP server

                    PASS joe@fake.com
3) FTP client   ----------------------> FTP server

                       230 Logged in
4) FTP client   <---------------------- FTP server
```

Here we cannot use a single plugin to detect this entire session. So, instead we use two plugins: the first plugin looks for packets 1) and 2) and the second plugin looks for packets 3) and 4).

A review of the above plugin shows that plugin 1018 matches 1) and 2) in the session by keying on the patterns "USER ftp" and the 331 return code. Plugin 1019 matches on 3) and 4) by keying on the patterns "PASS" and the 230 return code.

Copyright © 2002-2012 Tenable Network Security, Inc.

Notice that plugin 1019 has the following field: **dependency=1018**. This field indicates the plugin 1018 must first evaluate successfully before plugin 1019 may be evaluated (i.e., that plugin 1019 depends on plugin 1018's success before it can be evaluated).

One more step is needed to complete the plugin for the anonymous FTP session. We need to ensure that both plugins are actually evaluating the same FTP session. We can do this by attaching a time dependency to plugin 1019. The field **time-dependency=5** indicates that plugin 1018 must have evaluated successfully in the last five seconds for 1019 to be evaluated. In this way we can ensure that both plugins are evaluating the same FTP session.

WRITING PASSIVE VULNERABILITY SCANNER REAL-TIME PLUGINS

Real-Time Plugin Model

PVS real-time plugins are exactly the same as PVS vulnerability plugins with two exceptions:

- they can occur multiple times
- their occurrence may not be recorded as a vulnerability

For example, an attacker may attempt to retrieve the source code for a Perl script from an Apache web server. If the PVS observes this event, it would be logical to send a real-time alert. It would also be logical to mark that the Apache server is potentially vulnerable to some sort of Perl script source code download. In other cases, it may be more logical to just log the attempt as an event, but not a vulnerability. For example, a login failure over FTP is an event that may be worth logging, but does not indicate a vulnerability.

As the real-time plugins are written, there are two keywords that indicate to the PVS that these are not a regular vulnerability plugin. These are the "**realtime**" and "**realtimeonly**" keywords. All keywords will be covered more in-depth in the next session, but the basic difference of the "**realtime**" and "**realtimeonly**" keywords is that "**realtime**" events go into the vulnerability database and the "**realtimeonly**" events do not.

In the previous example, the FTP user login failure would be marked as a "**realtimeonly**" event because we would like real-time alerting, but not a new entry into the vulnerability database.

New Keywords

Name	Description
include	The PVS supports dependencies where one plugin depends on a list of other plugins. The "**include**" keyword specifies a file that contains a list of other PVS IDs to be dependent. Tenable includes a **services.inc** file with the PVS that lists the major applications such as SMTP, NTP, FTP, etc.
realtime	If a plugin has this keyword, then the PVS will generate a SYSLOG message or real-time log file entry the first time this plugin matches. This prevents vulnerabilities that are worm

Copyright © 2002-2012 Tenable Network Security, Inc.

	related from causing millions of events. For example, the plugins for the Sasser worm only generate one event. Output from plugins with this keyword will show up in the vulnerability report.
realtimeonly	If a plugin has this keyword, then the PVS will generate a SYSLOG message or real-time log file entry each time the plugin evaluates successfully. These plugins never show up in the report file.
track-session	This keyword will cause the contents of a session to be reported (via SYSLOG or the real-time log file) a specified number of times after the plugin containing this keyword was matched. This is an excellent way to discover what a hacker "did next" or possibly what the contents of a retrieved file were.
trigger-dependency	Normally if a plugin has multiple dependencies, then all of those dependencies must be successful for the current plugin to evaluate. However, the "**trigger-dependency**" keyword allows a plugin to be evaluated as long as at least one of its dependencies is successful.

Example Failed Telnet Login Plugin

The easiest way to learn about PVS real-time plugins is to evaluate some of those included by Tenable. Below is a plugin that detects a failed Telnet login to a FreeBSD server.

```
# Look for failed logins into an FreeBSD telnet server
id=0400
hs_sport=23
dependency=1903
realtimeonly
name=Failed login attempt
description=PVS detected a failed login attempt to a telnet server
risk=LOW
match=Login incorrect
```

This plugin has many of the same features as a vulnerability plugin. The ID of the plugin is 0400. The high-speed port is 23. We need to be dependent on plugin 1903 (which detects a Telnet service). The "**realtimeonly**" keyword tells the PVS that if it observes this pattern, that it should alert on the activity, but not record any vulnerability.

Under the SecurityCenter, events from the PVS are recorded alongside other IDS tools.

Example Finger User List Enumeration Plugin

The finger daemon is an older Internet protocol that allowed system users to query remote servers to get information about a user on that box. There have been several security holes in this protocol that allowed an attacker to elicit user and system information that could be useful to attackers.

```
id=0500
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
dependency=1277
hs_sport=79
track-session=10
realtimeonly
name=App Subversion - Successful finger query to multiple users
description=A response from a known finger daemon was observed which
       indicated that the attacker was able to retrieve a list of three or
       more valid user names.
risk=HIGH
match=Directory:
match=Directory:
match=Directory:
```

With this plugin, we are only looking for these patterns on systems where a working finger daemon has been identified (dependency #1277). In this plugin though, we see the use of the "**track-session**" keyword. If this plugin is launched with a value of 10, the session data from the next 10 packets is tracked and logged in either the SYSLOG or real-time log file.

During a normal finger query, if only one valid user is queried, then only one home directory will be returned. However, many of the exploits for finger involve querying for users such as "NULL", "0", or "..". This causes vulnerable finger daemons to return a listing of all users. In that case, this plugin would be activated because of the multiple "Directory:" matches.

Example Unix Password File Download Web Server Plugin

This plugin below looks for any download from a web server that does not look like HTML traffic, but does look like the contents of a generic Unix password file.

```
id=0300
dependency=1442
hs_sport=80
track-session=10
realtimeonly
name=Web Subversion - /etc/passwd file obtained
description=A file which looks like a Linux /etc/passwd file was
       downloaded from a web server.
risk=HIGH
match=!<HTML>
match=!<html>
match=^root:x:0:0:root:/root:/bin/bash
match=^bin:x:1:1:bin:
match=^daemon:x:2:2:daemon:
```

The plugin is dependent on PVS ID 1442, which detects web servers. In the match statements, we are attempting to ignore any traffic that contains valid HTML tags, but also has lines that start with common Unix password file entries.

Example Generic Buffer Overflow Detection on Windows Plugin

One of the PVS's strongest intrusion detection features is its ability to recognize specific services, and then to look for traffic occurring on those services that should never occur unless they have been compromised. Since the PVS can keep track of both sides of a conversation and make decisions based on the content of each, it is ideal to look for Unix

Copyright © 2002-2012 Tenable Network Security, Inc.

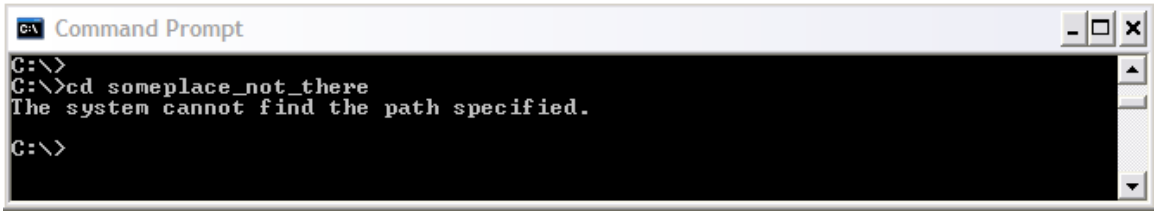
and Windows command shells occurring in services that should not have those command shells in them. Here is an example plugin:

```
# look for Windows error when a user tries to
# switch to a drive that doesn't exist
id=0201
include=services.inc
trigger-dependency
track-session=10
realtimeonly
name=Successful shell attack detected - Failed cd command
description=The results of an unsuccessful attempt to change drives on a
       Windows machine occurred in a TCP session normally used for a
       standard service. This may indicate a successful compromise of this
       service has occurred.
risk=HIGH
pmatch=!>GET
pregexi=cd
match=!>550
match=^The system cannot find the
match=specified.
```

This plugin uses the "**include**" keyword that identifies a file that lists several dozen PVS IDs, which identify well known services such as HTTP, DNS, and NTP. The plugin will not even get evaluated unless the target host is running one of those services.

The keyword "**trigger-dependency**" is needed to ensure the plugin is evaluated even if there is only one match in the **services.inc** file. Otherwise, the PVS would only evaluate this plugin if the target host was running all PVS IDs present in the **services.inc** file. The "**trigger-dependency**" keyword basically says that at least one PVS ID specified by one or more dependency or include rules must be present.

And finally, the logic of plugin detection is looking for the following type of response on a Windows system:



In this case, a user has attempted to use the "**cd**" command to change directories within a file system and the attempt was not allowed. This is a very common event that occurs once a remote hacker has compromised a Windows 2000 or Windows 2003 server with a buffer overflow. What the PVS plugin is looking for in this specific event is a network session that should not be there.

Looking at the plugin logic, there are "**pmatch**" and "**pregexi**" statements that attempt to ensure that the session is not an HTTP session, and that the previous side of the session contains the string "cd".

Copyright © 2002-2012 Tenable Network Security, Inc.



One could argue that the "**pregexi**" statement could be expanded to include the trailing space after the "d" character and also the first character.

The plugin then looks for the expected results of the failed "cd" command. The first match statement makes sure this pattern is not part of the FTP protocol. It turns out that looking for "cd" in one side of a session and the error of attempting to change to a directory in an FTP session would cause false positives for this plugin. Adding a rule to ignore if a line starts with "550" avoids this. While writing and testing this plugin, Tenable considered having a different set of plugins just for FTP, but the additional filter statement took care of any false positives we had been seeing. And finally, the last two match statements look for the results of the failed change directory attempt. They are spread across two match statements and could have been combined into one regular expression statement, but there was enough content in the basic message to have them split into higher-speed matching.

Passive Vulnerability Scanner Corporate Policy Plugins

Most companies have an "Acceptable Use Policy" that defines appropriate use of the company's IT facilities. Often, this policy is abused to some extent since detecting abuse can be difficult.

The PVS can help in this regard through use of PVS Corporate Policy plugins. These plugins can be used to look for policy violations and items such as credit card numbers, Social Security numbers, and other sensitive content in motion.

Tenable ships PVS with a large number of plugins that are frequently updated. The primary focus of these plugins is to discover hosts, applications and their related client/server vulnerabilities. The list of built-in PVS checks is available at the following location:

http://static.tenable.com/dev/tenable_plugins.pdf

Many of the available plugins already detect activities that would fall into the "Inappropriate Use" category in most companies. Some of the activities that are detected through these plugins include (but are not limited to):

- game server detection
- botnet client and server detection
- peer to peer file serving
- IRC server/client
- chat clients
- tunneling software or applications like Tor, GoToMyPC and LogMeIn

Detecting Custom Activity Prohibited by Policy

The plugins provided with PVS are useful for detecting generally inappropriate activities, but there may be times when more specific activities need to be detected. For example, a company may want to have an alert generated when email is sent to a competitor's mail service or if users are managing their myspace.com web page from the corporate network.

Tenable provides the ability for users to write their own custom plugins, as documented in the section "Writing Passive Vulnerability Scanner Plugins". These plugins are saved as "**prm**" files.

Copyright © 2002-2012 Tenable Network Security, Inc.

The following example shows how to create a custom plugin to detect users logging into myspace.com accounts. First, a unique plugin ID is assigned, in this case "9000". So, the first line of our plugin will be:

```
id=9000
```

Next, we will want to have a description of what the vulnerability detects:

```
description=The remote client was observed logging into a myspace.com
account.  You should ensure that such behavior is in alignment with corporate
policies and guidelines.  For your information, the user account was logged
as:<br> %L
```

The "%L" will be the results of our regular expression statement that will be created later. Basically, we want to log the source address of the offending computer as well as the user ID that was used to log in. Next, we create a distinct name for our plugin.

```
name=POLICY - myspace usage detection
```

Note that the name begins with the string "POLICY". This will make all POLICY violations easily searchable from the SecurityCenter interface.

You could also define a SecurityCenter dynamic asset list that contains only POLICY violators.

The next field defines a "family". For this example, the application is a web browser, so the family ID is defined as follows:

```
family=Web Clients
```

Since this is a web browser, a dependency can be assigned that will tell PVS to only look at clients that have been observed surfing the web:

```
dependency=1735
```

Further, since we are looking at client traffic, we will define:

```
clientissue
```

Next, we assign a risk rating for the observed behavior:

```
risk=MEDIUM
```

In the final section we create "**match**" and "**regex**" statements that PVS will look for passively. We want all of these statements to be true before the client is flagged for inappropriate usage:

```
match=>POST /
```

The web request must begin with a POST verb. This will weed out all "GET" requests.

```
match=^Host: login.myspace.com
```

Copyright © 2002-2012 Tenable Network Security, Inc.

The statement above ensures that they are posting to the host "login.myspace.com".

Finally, we have a **match** and **regex** statement that detects the user's login credentials:

```
match=email=
regex=email=.*%40[^&]+
```

Putting it all together, we have a single plugin as follows:

```
id=9000

family=Web Clients
clientissue
dependency=1735

name=MySpace_Usage
description=The remote client was observed logging into a myspace.com
account.  You should ensure that such behavior is in alignment with
Corporate Policies and guidelines.  For your information, the user account
was logged as:  %L e
risk=MEDIUM
solution=Stay off MySpace.

match=>POST /
match=^Host: login.myspace.com
match=email=
regex=email=.*%40[^&]+
```

This plugin could be named **myspace.prm** and added into the **/opt/pvs/var/pvs/plugins/** directory. If the SecurityCenter is being used to manage one or more PVS systems, use the plugin upload dialog to add the new .prm file.

If you wish to create a policy file that includes multiple checks, use the reserved word "NEXT" within the policy file. For example:

```
id=9000
…
rest of plugin
…

NEXT
id=9001
…
etc.
```

Detecting Confidential Data in Motion

Many organizations want to ensure that confidential data does not leave the network. PVS can aid in this by looking at binary patterns within observed network traffic. If critical documents or data can be tagged with a binary string, such as an MD5 checksum, the PVS will have the ability to detect these files being passed outside the network. For example:

Create a document that has a binary string of:

Copyright © 2002-2012 Tenable Network Security, Inc.

0xde1d7f362734c4d71ecc93a23bb5dd4c and
0x747f029fbf8f7e0ade2a6198560c3278

A PVS plugin could then be created to look for this pattern as follows:

```
id=9005
trigger-dependency
dependency=2004
dependency=2005
hs_dport=25
description=POLICY - Confidential data passed outside the
corporate network.  The Confidential file don'tshare.doc was
just observed leaving the network via email.
name=Confidential file misuse
family=Generic
clientissue
risk=HIGH
bmatch=de1d7f362734c4d71ecc93a23bb5dd4c
bmatch=747f029fbf8f7e0ade2a6198560c3278
```

These binary codes were created by simply generating md5 hashes of the following strings:

```
"Copyright 2006 BigCorp, file: don'tshare.doc"
"file: don'tshare.doc"
```

The security compliance group maintains the list of mappings (confidential file to md5 hash). The md5 hash can be embedded within the binary file and could then be tracked as it traversed the network.

Similar checks can be performed against ASCII strings to detect, for example, if confidential data was cut-and-pasted into an email. Simply create text watermarks that appear benign to the casual observer and map to a specific file name. For example:

```
"Reference data at \\192.168.0.2\c$\shares\employment files for HR data
regarding Jane Mcintyre" could be a string which maps to a file named
Finances.xls.
```

A PVS plugin could look for the string as follows:

```
id=9006
trigger-dependency
dependency=2004
dependency=2005
hs_dport=25
description=POLICY - Confidential data passed outside the
corporate network.  Data from the confidential file Finances.xls was just
observed leaving the network via email.
name=Confidential file misuse
family=Generic
clientissue
risk=HIGH
match=Reference data at
match=192.168.0.2\c$\shares\employmentfiles
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
match=for HR data regarding Jane Mcintyre
```

The two example plugins above (IDs 9005 and 9006) would detect files leaving the network via email. Most corporations have a list of ports that are allowed outbound access. SMTP is typically one of these ports. Other ports may include FTP, Messenger client ports (e.g., AIM, Yahoo and ICQ), or Peer2Peer (e.g., GNUTELLA and bittorrent). Depending on your specific network policy, you may wish to clone plugins 9005 and 9006 to detect these strings on other outbound protocols.

PASSIVE VULNERABILITY SCANNER OPERATING SYSTEM FINGERPRINTS

PASSIVE OPERATING SYSTEM FINGERPRINTING

The PVS has the ability to identify the likely operating system of a host by looking at the packets it generates. Specific combinations of TCP packet entries, such as the window size and initial time-to-live (TTL) values, allow the PVS to predict the operating system generating the traffic.

These unique TCP values are present when a server makes or responds to a TCP request. All TCP traffic is initiated with a "SYN" packet. If the server accepts the connection, it will send a response that is known as a "SYN-ACK" packet. If the server cannot or will not communicate, it will send a reset (RST) packet. When a server sends a "SYN" packet, the PVS will apply the list of operating system fingerprints and attempt to determine the type of the operating system.

Tenable Network Security has received permission to re-distribute the passive operating fingerprints from the author of SinFP open source project, which is available at:

http://www.gomor.org/sinfp

UNDERSTANDING THE FINGERPRINT LANGUAGE

The fingerprint language specifies a variety of settings within the TCP and IP header of a SYN-ACK packet. Below is a plugin for a Linux 2.2.20 server:

```
S20:64:1:60:M*,S,T,N,W0:.:Linux:2.2.20 and newer
```

The format of an operating system fingerprint is as follows:

```
wwww:ttt:D:ss:OOO...:QQ:OS:Details
```

From left to right, the plugin specifies the initial window size of the TCP communication, the initial time-to-live value in the IP header, the value of the "don't fragment" bit, the size of the SYN packet and a variety of unique TCP options, "quirks" and an operating system description. The description of the operating system includes both genre and type. For example, you may have a system that identified a host as "Windows" and of the "95" genre.

When specifying the plugin, TCP options and "quirks" can be specified. These are uniquely present in certain operating systems and can be used to accurately identify the operating system passively.

Copyright © 2002-2012 Tenable Network Security, Inc.

P0f2 TCP Options

```
N         NOP option
Wnnn      window scaling option, value nnn (or * or %nnn)
Mnnn      maximum segment size option, same as above
S         selective ACK is permitable
T         timestamp options are present
T0        a timestamp option is present with a zero value
?n        unrecognized option number n
```

P0f2 "Quirks"

```
E         End-Of-Line terminated options
P         Options are present past the EOL entry
Z         The IP packet ID number is 'zero'
I         IP options are present
U         The TCP 'urgent' value is non-zero
X         The unused (x2) TCP header option is non-zero
A         The ACK number is non-zero
T         The second timestamp option is non-zero
F         Unusual TCP flags are set
D         There is a data payload
!         There are broken options
```

Keep in mind that these checks are applied to the "SYN" packets that are emitted by the servers we are monitoring with the PVS. Based on the fields of the SYN packet, the operating system can be determined.

ADDING NEW FINGERPRINTS

In operation though, it is common for a specific server to have no match at all. If a match does not occur, the variables in the fields of the SYN will be reported by the PVS. For example, within a test system, the PVS may report the following:

```
The remote host OS could not be recognized. Its fingerprint is:
16384:255:1460:1:0:1:1:48
```

If the operating system of this packet was known, it could be added to the end of the **osfingerprints.txt** file. The PVS would require a restart prior to using the new passive operating system plugin.

Tenable encourages PVS users to submit their passive plugins to either Tenable via email or directly to the P0f home page.

FOR FURTHER INFORMATION

Tenable regularly updates PVS's plugins and these can be viewed online at http://static.tenable.com/dev/tenable_plugins.pdf. An RSS feed of the latest plugins is available here: http://www.tenable.com/pvs.xml. Tenable Network Security, Inc. may be contacted via email for PVS support at sales@tenable.com or support@tenable.com.

Copyright © 2002-2012 Tenable Network Security, Inc.

ABOUT TENABLE NETWORK SECURITY

Tenable Network Security, the leader in Unified Security Monitoring, is the source of the Nessus vulnerability scanner and the creator of enterprise-class, agentless solutions for the continuous monitoring of vulnerabilities, configuration weaknesses, data leakage, log management, and compromise detection to help ensure network security and FDCC, FISMA, SANS CAG, and PCI compliance. Tenable's award-winning products are utilized by many Global 2000 organizations and Government agencies to proactively minimize network risk. For more information, please visit http://www.tenable.com/.

Tenable Network Security, Inc.
7063 Columbia Gateway Drive
Suite 100
Columbia, MD 21046
410.872.0555
www.tenable.com

Copyright © 2002-2012 Tenable Network Security, Inc.

APPENDIX 1: EXAMPLE PVS.CONF CONFIGURATION FILE

```
options {
    # When adding new port, application, or vulnerability information to
    # the PVS model of the observed network, the report-threshold is used
    # to limit false positives and reports on ephemeral ports. An item
    # will not be reported until it has occurred the specified number
    # of times.
    report-threshold 3;

    # Specify the files into which PVS will write its report information.
    # These settings can be overridden at the command line using the -r
    # and -x options for the NSR and XML formats, respectively. For
    # example, 'pvs -r <report-file>' re-defines the name of the NSR-format
    # report file. In addition, HTML reports are generated for web browser
    # viewing in the /opt/pvs/var/pvs/reports/html/ directory.
    nsr-report-file    "/opt/pvs/var/pvs-proxy/scans/pvs-report.nsr";
    xml-report-file    "/opt/pvs/var/pvs/reports/report.xml";
    nessus-report-file "/opt/pvs/var/pvs/reports/report.nessus";

    # Specify the version of the nessus report to be written by PVS.
    nessus-report-version 2;

    # When the following option is enabled, PVS will generate HTML reports.
    # generate-html-reports;

    # PVS logging. The log-directory keyword specifies a location
    # for the pvs binary to write system logs. Log files of the
    # format YYYYMM.log (such as 200502.log) will be created. These
    # detail PVS operation and can be useful for debugging.
    log-directory      "/opt/pvs/var/pvs/logs";

    # which interface are we listening on ?
    interface "eth1";
    key-file "/opt/pvs/var/pvs/tenable.key";

    #disabled-plugins "/opt/pvs/var/pvs/disabled-plugins.txt";
    plugins-directory "/opt/pvs/var/pvs/plugins";
    scripts-directory "/opt/pvs/var/pvs/scripts";

    # The disabled-scripts keyword specifies a file containing names
    # of PASL scripts that should be disabled.
    disabled-scripts "/opt/pvs/var/pvs/disabled-scripts.txt";

    # The following file can be used to specify plugins that should be
    # run in real-time mode. Any plugins listed by ID in the file will
    # effectively have the realtime keyword added when they are loaded,
    # regardless of whether or not the keyword was originally present.
    # realtime-plugins  "/opt/pvs/var/pvs/realtime-plugins.txt";

    # The max-packet-cache-size keyword specifies the maximum size in
    # megabytes of the cache that will be used to store the contents of
    # the packets collected. By default it is set to 1024 MB. When the
    # cache is full, any subsequent packets captured will be dropped
    # until space in the cache becomes available. The size of the cache
    # will change dynamically and can be expected to grow to this maximum
    # number only when needed. The number configured should not be greater
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
# than 1024 MB. It can be set to a number less than 1024 MB.
max-packet-cache-size 1024;

# The memory keyword specifies the size in megabytes of the session
# table. By default it is set to 50. However, if you have a large
# network (such as a university network with 10,000 nodes or more)
# a setting of 500 should be used. PVS will use another 200 to 300
# MB to store vulnerability and host data in addition to the memory
# used in the session table. Please refer to the documentation for
# more information.
memory 50;

# The lifetime of a report (in days)
report-lifetime 30;

# When the following option is enabled, the PVS knowledge base will
# be saved on disk for recovery after the program is restarted.
save-knowledge-base;

# The maximum length of time in seconds that a knowledge-base entry
# remains valid after its addition.
kb-max-age 864000;

# Store the report to a file every X minutes where X is whatever
# integer value the report-frequency is set to. For example,
# 'report-frequency 60;' will result in the report being written
# to disk every 60 minutes (1 hour)
report-frequency 60;

# Detect encrypted sessions
# If PVS detects a server within the focus network which
# has one or more of these plugins active, it will attempt to
# analyze all traffic to/from the server for evidence that it
# is encrypted. This can be tailored further by specifying
# specific traffic to ignore based on port and network
detect-encryption {

####################
# Dependencies
####################
# web server
dependency 1442;
# dns server
dependency 1000;
# ssh server
dependency 1967;
# ftp server
dependency 1803;
# mail server
dependency 2004;
dependency 2005;
# imap server
dependency 1086;
# ntp server
dependency 1144;
dependency 1146;
dependency 1148;
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
dependency 1149;
dependency 1150;
dependency 1151;
# snmp
dependency 1120;
# vnc
dependency 1882;
# SSL server
dependency 1133;
dependency 1134;
dependency 1135;

# don't examine web traffic
exclude 0.0.0.0/0 80;
exclude 0.0.0.0/0 443;

# don't examine ssh traffic
exclude 0.0.0.0/0 22;
}

# Detect interactive sessions
# If PVS detects a server within the focus network which
# has one or more of these plugins active, it will attempt to
# analyze all traffic to/from the server for evidence that it
# is utilizing an interactive network session such as telnet,
# a command shell from a buffer overflow or chat. This can be
# tailored further by specifying specific traffic to ignore
# based on port and network address.
detect-interactive-sessions {
####################
# Dependencies
####################
# web server
dependency 1442;
# dns server
dependency 1000;
# ssh server
dependency 1967;
# ftp server
dependency 1803;
# mail server
dependency 2004;
dependency 2005;
# imap server
dependency 1086;
# ntp server
dependency 1144;
dependency 1146;
dependency 1148;
dependency 1149;
dependency 1150;
dependency 1151;
# snmp
dependency 1120;
# vnc
dependency 1882;
# SSL server
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
dependency 1133;
dependency 1134;
dependency 1135;

#################
# Exclusions
#################
# ftp control channel
exclude 0.0.0.0/0 21;
# ssh
exclude 0.0.0.0/0 22;
# telnet
exclude 0.0.0.0/0 23;
# smtp
exclude 0.0.0.0/0 25;
# pop3
exclude 0.0.0.0/0 110;
# imap
exclude 0.0.0.0/0 143;
}

# Are we running on a high-speed network ?
# high-speed;

# When the following option is enabled, PVS will
# strip VLAN tags when processing PVS packets.
# strip-vlan-tags;

# Uncomment this file to record PVS's realtime IDS events
# realtime-file "/opt/pvs/var/pvs/realtime-logs.txt";

# The max-realtime-file option specifies the maximum amount of
# data from realtime IDS events that will be stored in one
# text file. The option must be specified in kilobytes, megabytes,
# or gigabytes by appending a 'K', 'M', or 'G' to the value. After
# this value is reached, a numbered file with the same base name
# as above will be used to store additional events. When the
# option is disabled, there is no limit on realtime file size.
max-realtime-file "2G";

# Uncomment the following line and specify the IP address of a
# SYSLOG server to send events for PVS's realtime signatures.
# Mulitple realtime-syslog keywords can be used to send realtime
# alerts to mulitple syslog servers.
# realtime-syslog 192.168.20.10;
# realtime-syslog 192.168.20.11;

# Uncomment the following line and specify the IP address of a
# SYSLOG server to send events for PVS's vulnerability signatures.
# Mulitple vulndata-syslog keywords can be used to send vulnerability
# alerts to mulitple syslog servers.
# vulndata-syslog 192.168.20.20;
# vulndata-syslog 192.168.20.21;

# Uncomment this keyword to cause PVS to record which ports are
# browsed by each host. For example, if a particular host is discovered
# to check email, send email and browse the web, ports 110, 25 and 80
```

Copyright © 2002-2012 Tenable Network Security, Inc.

```
    # would be marked as 'Client Side Port Usage'.
    # connections-to-services;

    # Uncomment this keyword to cause PVS to record who each host
communicates
    # with. For example, if a particular host checked email, sent email and
    # browsed the web on port 80, a list of each host being used to send or
    # receive email, as well as each web server would be shown. PVS will not
    # record how many times a connction occurred, just that a particular
network
    # connection has occurred.
    # show-connections;

    # Alert on new hosts not seen after running for 'N' days.
    # Tenable recommends a setting of at least 1 or 2 days. After
    # that, any host which has not been discovered during this
    # time period, will be alerted on as new. Without this setting,
    # PVS would 'discover' all of your hosts which are currently
    # running and not really 'new'.
    #
    # In addition, two other keywords are used to specify where and
    # how often PVS can persist a list of IP addresses which are
    # alive. The backup-file keyword specifies where the file
    # should be created and the backup-interval keyword specifies
    # how often in minutes, the file should be updated.
    #
    # Wait at least two days before finding 'new' hosts:
    # new-host-alert 2;
    #
    # How often and where our backup file of current hosts should
    # be persisted:
    # backup-interval 60;
    # backup-file "/opt/pvs/var/pvs/pvs-hosts.bin";

}

networks {
    0.0.0.0/0;
}

excluded-networks {

}
```

Copyright © 2002-2012 Tenable Network Security, Inc.

APPENDIX 2: WORKING WITH SECURITYCENTER

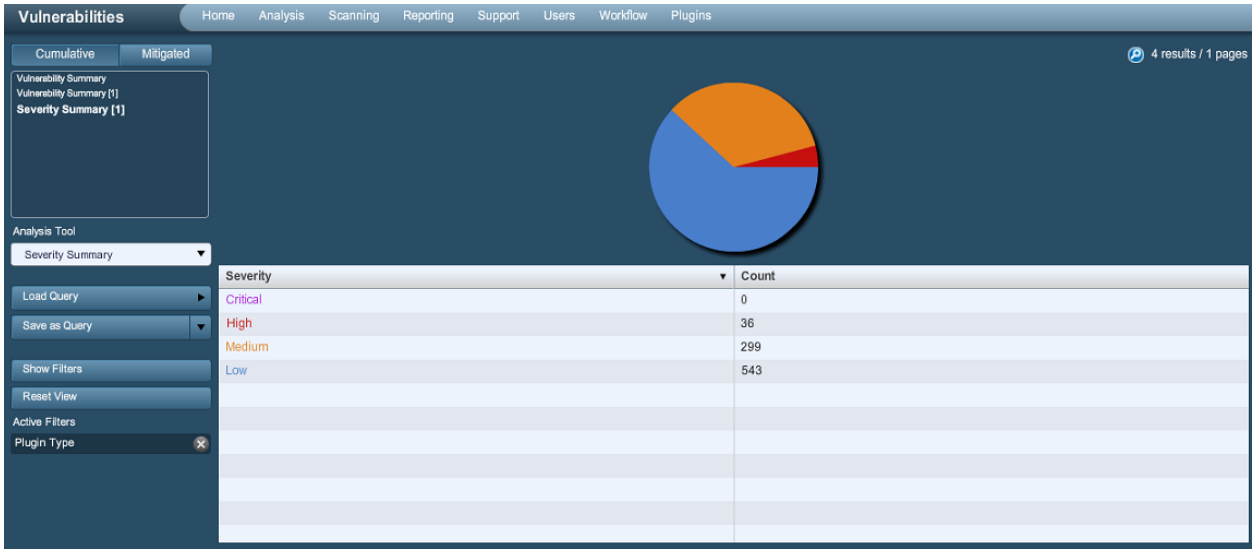
ARCHITECTURE

The PVS operates under the control of a SecurityCenter that provides it with passive vulnerability data. SecurityCenter has a variety of reporting, remediation and notification mechanisms to efficiently distribute vulnerability information across large enterprises. In addition, it can also control a distributed set of Nessus or other active vulnerability scanners. By combining active and passive vulnerability scanning, SecurityCenter can be used to efficiently and accurately manage security across large networks.

SecurityCenter can also push out new passive vulnerability plugins to multiple scanners and distribute the results of each PVS sensor to specific organizations or system administrators.

MANAGING VULNERABILITIES

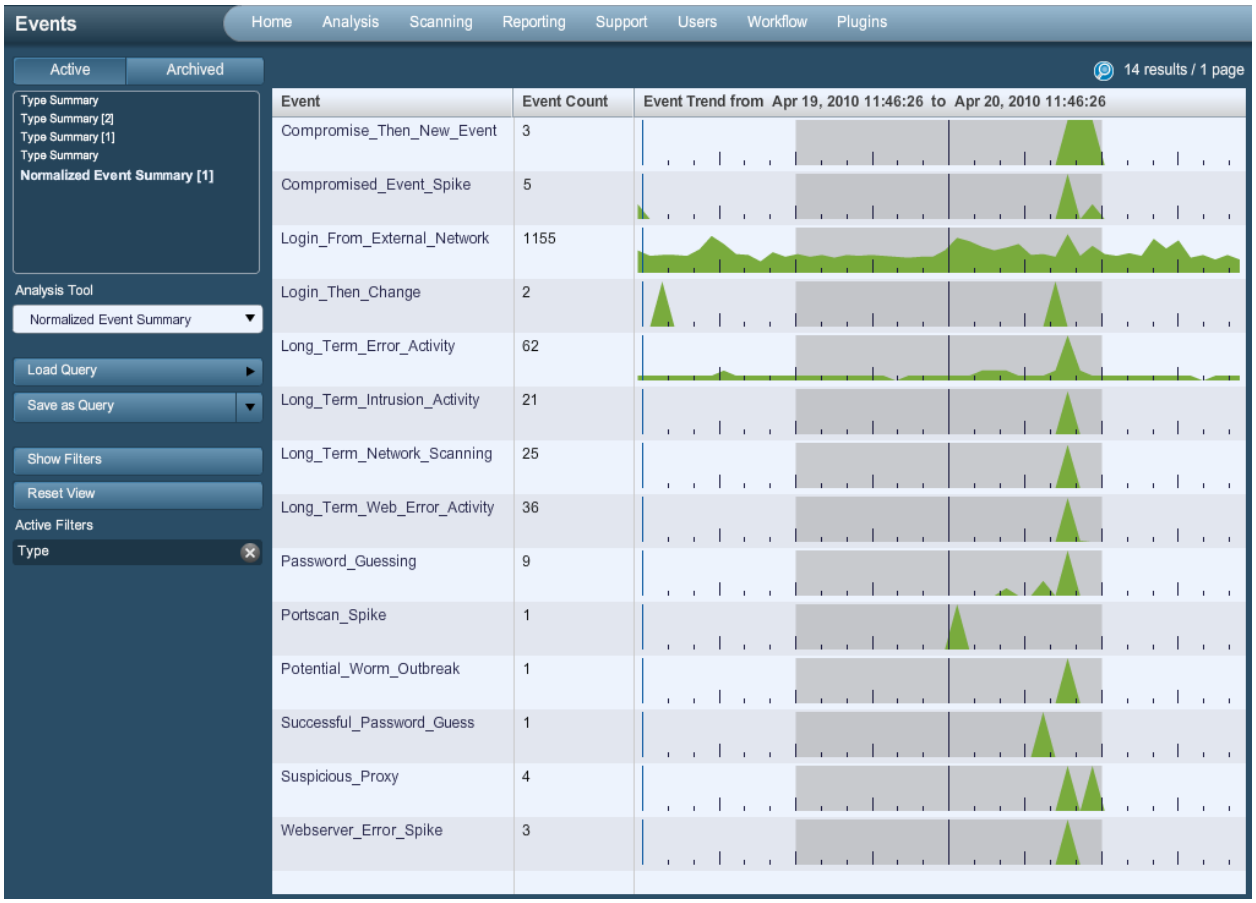
A screen capture of SecurityCenter 4 displaying a summary of vulnerabilities detected by the PVS is shown below. These vulnerabilities can be independently viewed by many different users with different access control. SecurityCenter also enables security managers to issue recommendations that help guide network administrators as to which vulnerabilities should be mitigated.



MANAGING IDS EVENTS

Starting with SecurityCenter 4, the LCE performs IDS event aggregation and automatically makes use of any vulnerability detected by the PVS when assessing when an IDS event may be impacting a vulnerable server.

Copyright © 2002-2012 Tenable Network Security, Inc.



In the above example, a filter is applied to only display events that have been correlated.

The Passive Vulnerability Scanner is Real-Time

Since the PVS's vulnerability data is being fed into SecurityCenter 24x7 and PVS's plugins are updated by Tenable, the accuracy of the passive vulnerability data in SecurityCenter greatly enhances the quality of the security information available to the SecurityCenter's users.

Copyright © 2002-2012 Tenable Network Security, Inc.

APPENDIX 3: WORKING WITH NESSUS

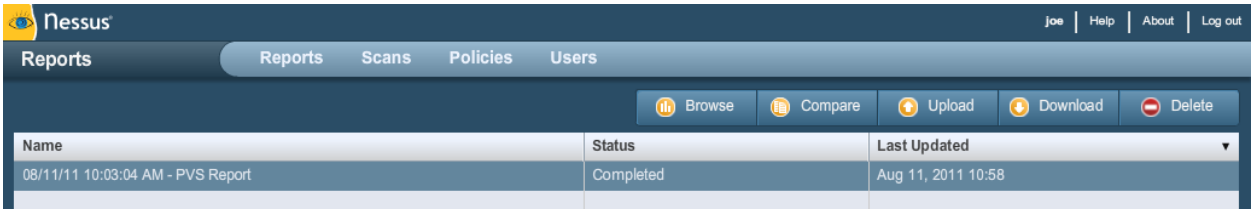
It is possible to view a PVS report from within the Nessus interface by enabling the **nessus-report-file** option in **/opt/pvs/etc/pvs.conf** and setting the path and filename for the report file output. To ensure that the greatest amount of PVS data is able to be imported into Nessus, it is necessary that the **nessus-report-version** be set to "2" as shown below.

```
# Specify the files into which PVS will write its report information.
# These settings can be overridden at the command line using the -r
# and -x options for the NSR and XML formats, respectively. For
# example, 'pvs -r <report-file>' re-defines the name of the NSR-format
# report file. In addition, HTML reports are generated for web browser
# viewing in the /opt/pvs/var/pvs/reports/html/ directory.
nsr-report-file    "/opt/pvs/var/pvs-proxy/scans/report.nsr";
xml-report-file    "/opt/pvs/var/pvs/reports/report.xml";
nessus-report-file "/opt/pvs/var/pvs/reports/report.nessus";

# Specify the version of the nessus report to be written by PVS.
nessus-report-version 2;

# When the following option is enabled, PVS will generate HTML reports.
# generate-html-reports;
```

1. Once the report is generated you may import the resulting **.nessus** file directly into your Nessus server. Confirm that the file is in a location that you can access from your web browser.
2. To import a scan, click on the "Upload" button from the "Reports" screen in the Nessus management interface. Use the "Browse…" button to select the **.nessus** scan file you want to import and click "Submit". Nessus will parse the information and make it available in the "Reports" interface where it may be viewed as any other report.



TUNING FOR NESSUS ONLY REPORTING

If SecurityCenter is not to be used to view PVS reports, there are opportunities to optimize settings in the PVS configuration to enhance its performance by disabling features that are unneeded when it is to only be viewed in the Nessus interface.

Review and tune monitored and excluded networks. Open the **/opt/pvs/etc/pvs.conf** file with a text editor. Networks are specified using CIDR notation and placed between the brackets after the directive. The **networks** option specifies the networks to be monitored. This is initially set by the PVS installation script and specifies your monitored network address space. The **excluded-networks** option specifies any networks that are to be excluded from PVS monitoring. If left blank, no addresses will be excluded.

When using PVS with Nessus only reporting, disable the following PVS reporting options in **pvs.conf** unless specifically needed for other uses:

Copyright © 2002-2012 Tenable Network Security, Inc.

- **generate-html-reports**
- **nsr-report-file**
- **xml-report-file**
- **realtime-plugins**
- **realtime-file**
- **realtime-syslog**
- **vulndata-syslog**

If connection data is not needed, disabling **connections-to-services** and **show-connections** in **pvs.conf** will allow PVS to only report vulnerability data.

Further tuning can be performed by disabling unwanted plugins through use of a disabled-plugins.txt file via the **disabled-plugins** option in **pvs.conf**. The file contains a list of IDs, one per line, followed by a new-line. If the file exists, PVS will disable the specified plugins when it starts.

When using the PVS without SecurityCenter, PVS plugin updates must be performed either manually or scheduled to run via the cron utility using the **/opt/pvs/bin/pvs-update-plugins.sh** script. The pvs service must be restarted following the plugin update.

Copyright © 2002-2012 Tenable Network Security, Inc.

APPENDIX 4: NON-TENABLE LICENSE DECLARATIONS AND PATENT

Below you will find 3rd party software packages that Tenable provides for use with the Passive Vulnerability Scanner (Patent 7,761,918 B2).

Section 1 (b) (ii) of the Passive Vulnerability Scanner License Agreement reads:

(ii) The Software may include code or other intellectual property provided to Tenable by third parties, including Plug-Ins that are not owned by Tenable, (collectively, "Third Party Components"). Any Third Party Component that is not marked as copyrighted by Tenable is subject to other license terms that are specified in the Documentation. By using the Software, you hereby agree to be bound by such other license terms as specified in the Documentation.

RELATED 3RD PARTY AND OPEN-SOURCE LICENSES

blowfish.h
This product includes cryptographic software written by Eric Young (eay@mincom.oz.au).
This product includes software written by Tim Hudson (tjh@mincom.oz.au).

```
crypto/bf/blowfish.h
Copyright (C) 1995-1997 Eric Young (eay@mincom.oz.au)
All rights reserved.

This package is an SSL implementation written by Eric Young
(eay@mincom.oz.au).
The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the
following conditions are aheared to.  The following conditions apply to all
code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code;
not just the SSL code.  The SSL documentation included with this distribution
is covered by the same copyright terms except that the holder is Tim Hudson
(tjh@mincom.oz.au).

Copyright remains Eric Young's, and as such any Copyright notices in the code
are not to be removed. If this package is used in a product, Eric Young
should be given attribution as the author of the parts of the library used.
This can be in the form of a textual message at program startup or in
documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the copyright notice, this list
of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must
display the following acknowledgement:
   "This product includes cryptographic software written by Eric Young
(eay@mincom.oz.au)"
```

Copyright © 2002-2012 Tenable Network Security, Inc.

The word 'cryptographic' can be left out if the rouines from the library being used are not cryptographic related :-).
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement:
"This product includes software written by Tim Hudson (tjh@mincom.oz.au)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

WinPcap

Copyright (c) 1999 - 2005 NetGroup, Politecnico di Torino (Italy).
Copyright (c) 2005 - 2007 CACE Technologies, Davis (California).
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the Politecnico di Torino, CACE Technologies nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the University of California, Lawrence Berkeley Laboratory and its contributors.
This product includes software developed by the Kungliga Tekniska Högskolan and its contributors.
This product includes software developed by Yen Yen Lim and North Dakota State University.

Copyright © 2002-2012 Tenable Network Security, Inc.