

Passive Vulnerability Scanner 4.0 User Guide

September 18, 2014

(Revision 12)

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.....	6
Download or Obtain the Software	6
PVS Subscription.....	6
SecurityCenter Continuous View.....	6
Obtain a License Key for SecurityCenter	7
Evaluation License Key or Activation Code.....	7
Upgrading	7
Upgrading PVS on Linux.....	7
Upgrading PVS on Windows.....	8
Upgrading PVS on Mac OS X.....	11
Initial Installation.....	17
Linux Installation	17
Windows Installation	17
Mac OS X Installation	26
Starting and Stopping PVS for Mac OS X	32
Removing PVS	32
Removing PVS for Linux	32
Removing PVS for Windows	32
Removing PVS for Mac OS X	33
Initial Configuration of the PVS Server	33
Using the PVS Interface.....	35
Navigation	35
Monitoring.....	37
Results	41
Users	42
Configuration	42
Command Line Operation	47
Command Line Operations for Linux	47
Starting the Passive Vulnerability Scanner for Linux	47
Stopping the Passive Vulnerability Scanner for Linux	47
File Locations.....	47
Command Line Operations for Windows	49
File Locations.....	49
Starting and Stopping PVS	50
Command Line Operations for Mac OS X	50
Stopping the Passive Vulnerability Scanner for Mac OS X.....	50
File Locations.....	50
Common Command Line Options	51

Define Unknown or Customized Ports	53
PVS Real-Time Traffic Analysis Configuration Theory.....	53
Focus Network.....	53
Detecting Server and Client Ports	54
Detecting Specific Server and Client Port Usage	54
What this Means for Firewall Rules	55
Working with the SecurityCenter.....	55
Selecting Rule Libraries and Filtering Rules	55
Detecting Encrypted and Interactive Sessions.....	56
Routes and Hop Distance	56
Alerting.....	56
New Host Alerting.....	56
Internal Passive Vulnerability Scanner IDs.....	57
What is a Passive Vulnerability Scanner ID?.....	57
Internal Passive Vulnerability Scanner IDs.....	57
Working with Passive Vulnerability Scanner Plugins.....	58
Vulnerability and Passive Fingerprint Overview	58
Downloading New Vulnerability Plugins	58
Writing Custom Plugin Libraries	58
Restarting the Passive Vulnerability Scanner.....	58
Writing Passive Vulnerability Scanner Plugins	58
Plugin Keywords	58
Plugin Libraries	61
Basic Passive Vulnerability Scanner Example	61
More Complex Passive Vulnerability Scanner Example	61
Case Insensitive Example.....	62
Passive Vulnerability Scanner Network Client Detection.....	63
The Passive Vulnerability Scanner can Match “Previous” Packets.....	63
The Passive Vulnerability Scanner can Match Binary Data	64
Negative Matches	64
Time Dependent Plugins.....	65
Writing Passive Vulnerability Scanner Real-Time Plugins	66
Real-Time Plugin Model.....	66
New Keywords.....	66
Example Failed Telnet Login Plugin	66
Example Finger User List Enumeration Plugin	67
Example Unix Password File Download Web Server Plugin	67
Example Generic Buffer Overflow Detection on Windows Plugin	68
Passive Vulnerability Scanner Corporate Policy Plugins	69
Detecting Custom Activity Prohibited by Policy	70
Detecting Confidential Data in Motion	71
Passive Vulnerability Scanner Operating System Fingerprints.....	72
Passive Operating System Fingerprinting.....	72
For Further Information	73
Appendix 1: Working with SecurityCenter.....	74
Architecture	74
Managing Vulnerabilities.....	74
Updating the PVS Management Interface	74

The Passive Vulnerability Scanner is Real-Time 74

Appendix 2: Syslog Message Formats 75

Appendix 3: PVS Activation without Internet Access..... 77

About Tenable Network Security..... 79

Introduction

This document describes the **Passive Vulnerability Scanner 4.0** (Patent 7,761,918 B2) architecture, installation, operation, integration with SecurityCenter, and export of data to third parties. Please email any comments and suggestions to support@tenable.com.

The Passive Vulnerability Scanner 4.0 is available for the following platforms:

- Red Hat Linux ES 5 / CentOS 5 32-bit and 64-bit
- Red Hat Linux ES 6 / CentOS 6 32-bit and 64-bit
- Mac OS X 10.8 and 10.9 64-bit
- Microsoft Windows Vista, 7, 8, Server 2008, and Server 2012

This document describes the Passive Vulnerability Scanner installation and operation on the Red Hat Linux, Mac OS X, and Microsoft Windows platforms 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:

- detect when systems are compromised based on application intrusion detection
- 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.

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. Resource requirements to consider for PVS deployments include raw network speed, the size of the network being monitored, and the configuration of the PVS application.

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

Scenario	Minimum Recommended Hardware
Passive Vulnerability Scanner managing up to 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: 4 GB RAM (8 GB RAM 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 comparative recommendations as VMs typically see up to a 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.

Disk space requirements for PVS will vary depending on usage based on the amount and length of time data is stored on the system.

Software and Licensing

Download or Obtain the Software

To install the PVS, obtain the correct version for your desired operating system from the “Downloads” section of 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 to ensure binary compatibility.

PVS Subscription

A PVS subscription Activation Code is available to enable PVS to operate in a stand-alone mode. This mode enables the PVS results to be viewed from a HTML interface enabled on the PVS server.

SecurityCenter Continuous View

Continuous View includes PVS as part of a bundled license package with SecurityCenter. This license allows an unlimited number of PVS deployments to monitor an unlimited number of networks. SecurityCenter's IP view will be constrained by the license purchased with it.

Obtain a License Key for SecurityCenter

When using a PVS with SecurityCenter, a license key may be purchased as an upgrade to an existing SecurityCenter installation. A license key is needed for each PVS installation attached to a SecurityCenter.

- **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.

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 licensed range. If a narrower range than the licensed range is specified, the narrower range will be used. For example, an organization 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.

Evaluation License Key or Activation Code

To obtain an evaluation Activation Code for PVS, contact sales@tenable.com. Evaluation Activation Codes are handled the same way by the PVS as a full Activation Code, except that an evaluation Activation Code will only allow monitoring for 30 days. During an evaluation of the PVS, all of the features are available.

Upgrading

This section describes the steps to upgrade an existing PVS installation on Linux and Windows platforms. During the upgrade process on both platforms, items that are commented out in the configuration file are not migrated. Review the configuration file for any information that may need to be preserved.

Upgrading PVS on Linux



Before upgrading, the PVS services must be stopped. Failure to do so may result in errors. Custom SSL certificates must be backed up before an upgrade. It is assumed that all commands are run with root privileges.

If you have used a PVS RPM to install PVS previously, an upgrade 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
```

Copy the file `/opt/pvs/var/pvs/tenable.key` to the machine that will be used to configure the PVS after the upgrade is complete.

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

```
# rpm -Uvh pvs-4.x.x-es6.x86_64.rpm
Preparing... ##### [100%]
   1:pvs      ##### [100%]

[*] PVS installation completed.
#
```

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

```
# service pvs start
```

After starting PVS, navigate to `https://<ipaddress or hostname>:8835`, which will display the PVS web frontend to log in for the first time. Follow the directions described in the section [Initial Configuration of the PVS Server](#) to complete the initial login.



Ensure that organizational firewall rules permit access to port 8835 on the PVS server.

Upgrading PVS on Windows



Before upgrading, the PVS services must be stopped. Failure to do so may result in errors. Custom SSL certificates must be backed up before an upgrade. All programs are run as a local administrator user. When UAC is enabled, right click on the installer program and select “Run as Administrator”.



Ensure that the latest version of the Microsoft Visual C++ 2010 Redistributable Package for your platform is installed prior to PVS software installation.



Prior to upgrading PVS, ensure that any other programs on the system utilizing WinPcap are stopped.

Stop the Tenable PVS Proxy Service from the Windows Services control panel.

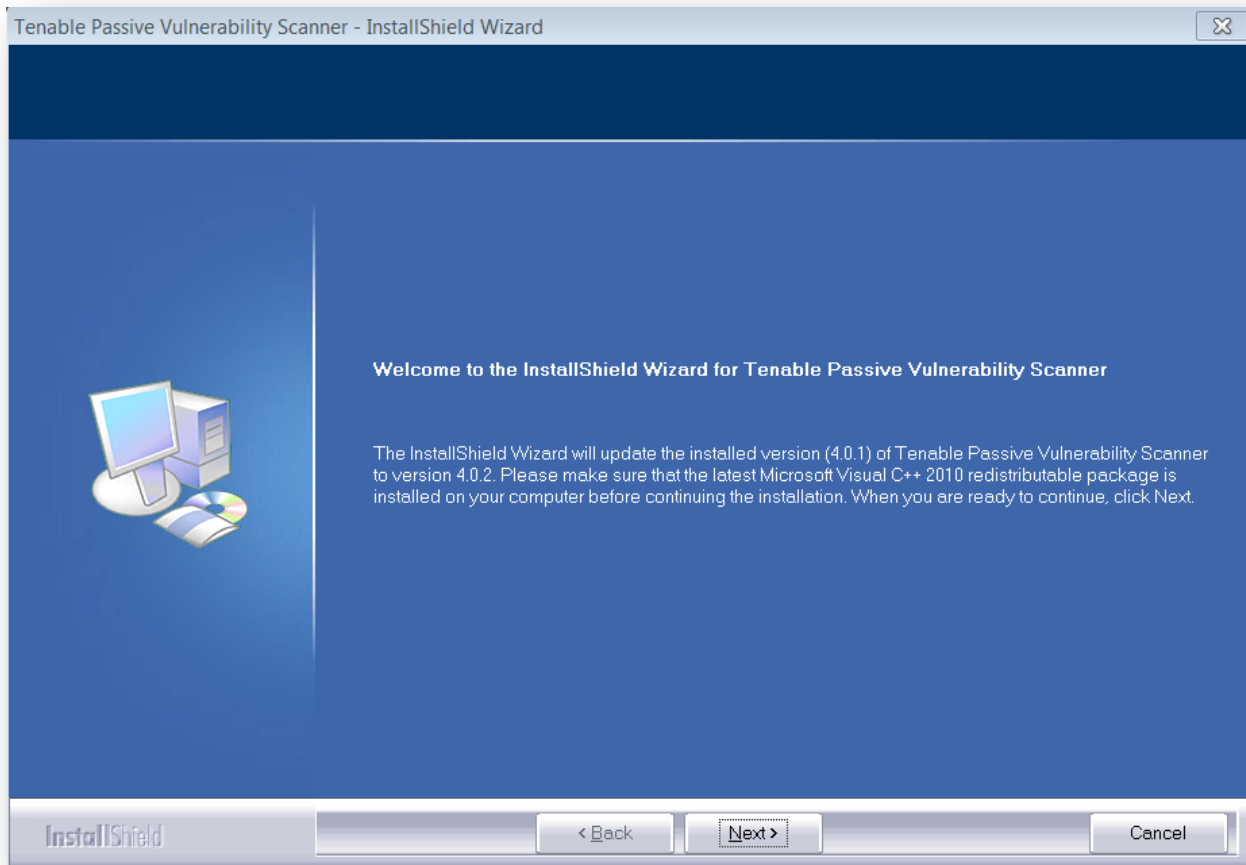
Copy the file `C:\ProgramData\Tenable\PVS\pvs\tenable.key` to the machine that will be used to configure the PVS after the upgrade is complete.

Start the PVS software for Windows upgrade by double-clicking on the `.exe` file downloaded from Tenable. Note that the specific filename will vary, depending on your platform and/or version:

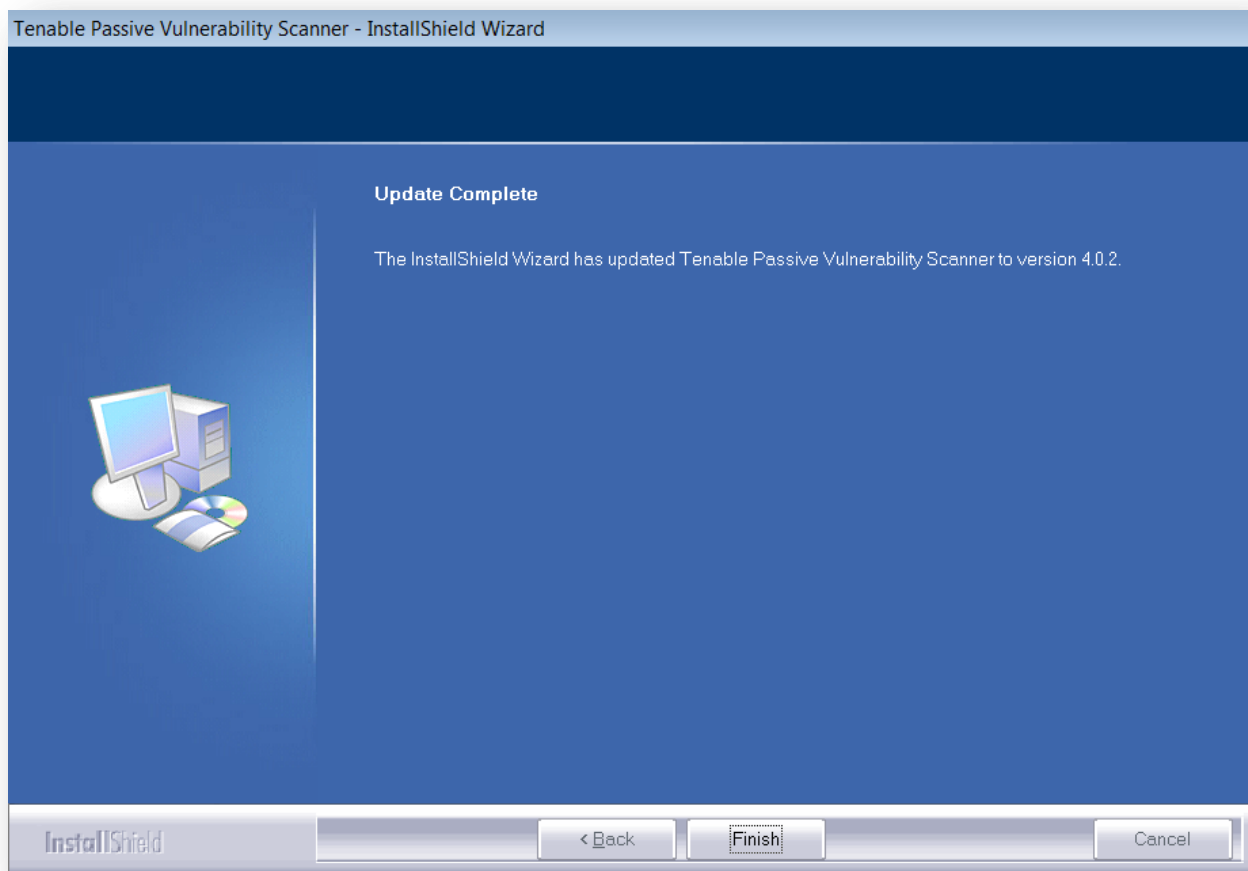


pvs-4.0.2-x64
InstallScript Setup Launcher
Tenable Network Security, Inc.

This will start the upgrade process by launching the InstallShield Wizard:



Clicking the “Next” button will begin the automated upgrade process. If the version of WinPcap is not at the appropriate level during the upgrade process, an upgrade window will be displayed to begin the process of upgrading WinPcap. Failure to install the recommended version of WinPcap may result in error with PVS monitoring.



Once completed, an “Update Complete” dialog will be displayed indicating that PVS has been updated to version 4.0. Select the “Finish” button to close the window.

After starting the PVS, navigate to `https://<ipaddress or hostname>:8835` to display the PVS web frontend to log in for the first time. Follow the directions described in the section [Initial Configuration of the PVS Server](#) to complete the initial login.



Ensure that organizational firewall rules permit access to port 8835 on the PVS server.

Upgrading PVS on Mac OS X

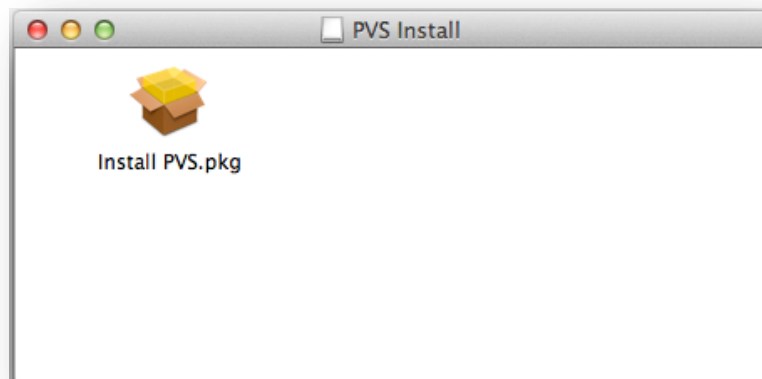


Before upgrading, the PVS services must be stopped. Failure to do so may result in errors. See the [“Starting and Stopping PVS for Mac OS X”](#) section for instructions. Custom SSL certificates must be backed up before an upgrade. All programs are run as a root user.

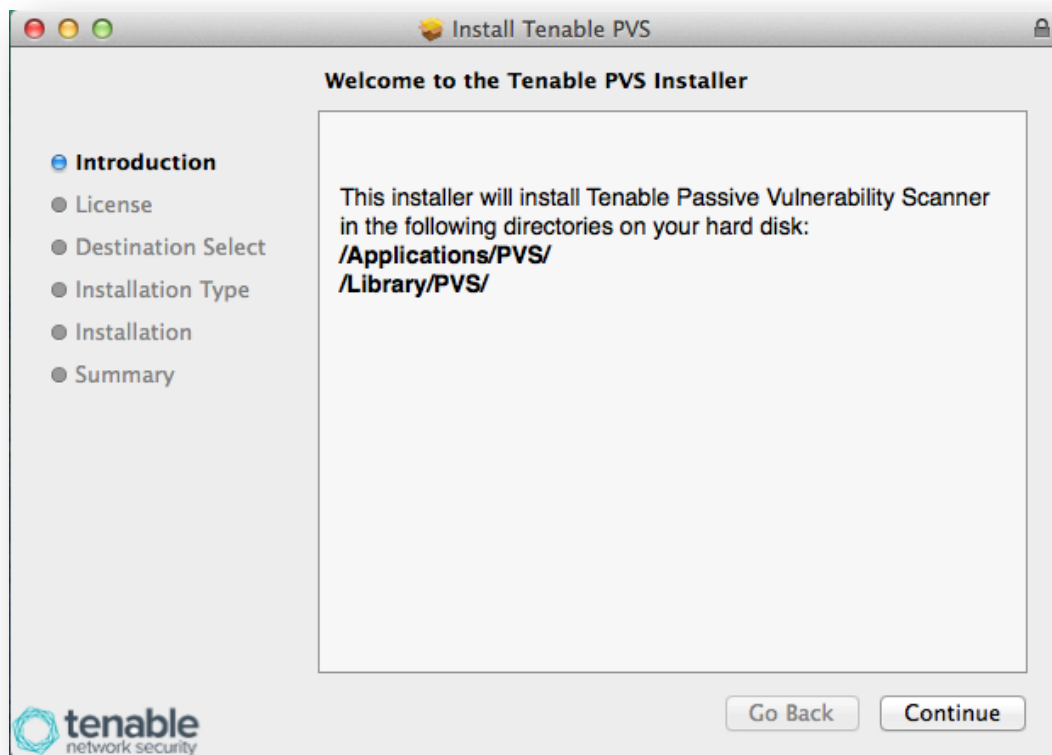
Begin upgrading the PVS software for Mac OS X by double clicking on the `.dmg` file downloaded from the [Tenable Support Portal](#) to mount the disk image “**PVS Install**”. Note that the specific filename will vary, depending on your version:



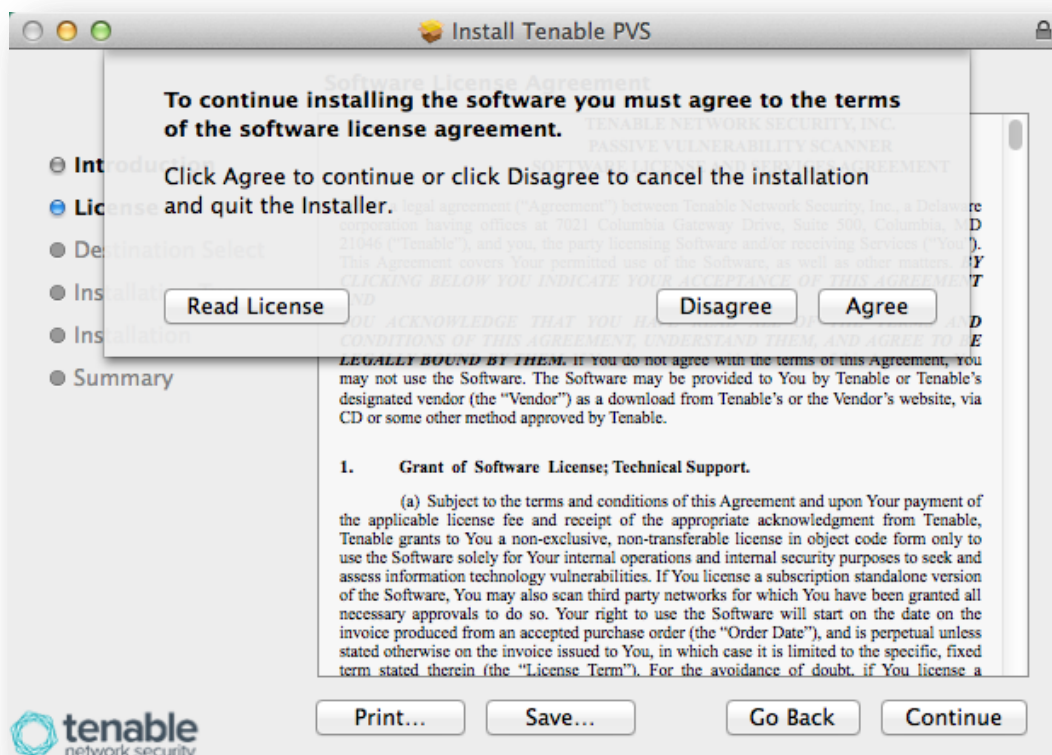
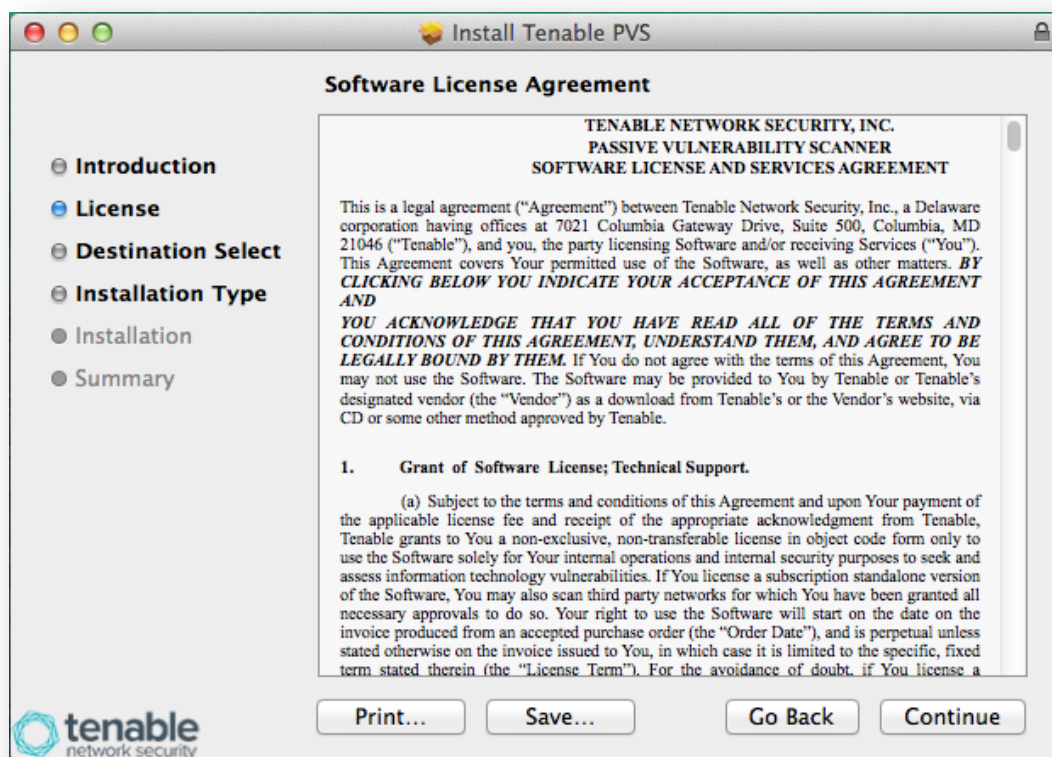
Double click on the Install PVS.pkg file to launch the Installer:



This will launch the Tenable PVS Installer, which will walk you through the upgrade process and any required configuration changes. At any point prior to completion, configuration options may be changed by clicking “Back” to go to the previous step. Clicking “Cancel” will abort the upgrade process completely.



The next screen displays the End User License Agreement (EULA). The text of the agreement can be copied and pasted into a separate document file for reference, saved using the “**Save...**” button, or it can be printed directly from this interface using the “**Print...**” button. You must agree to the license to continue the upgrade process and use PVS.



Click **Install** to begin the upgrade:



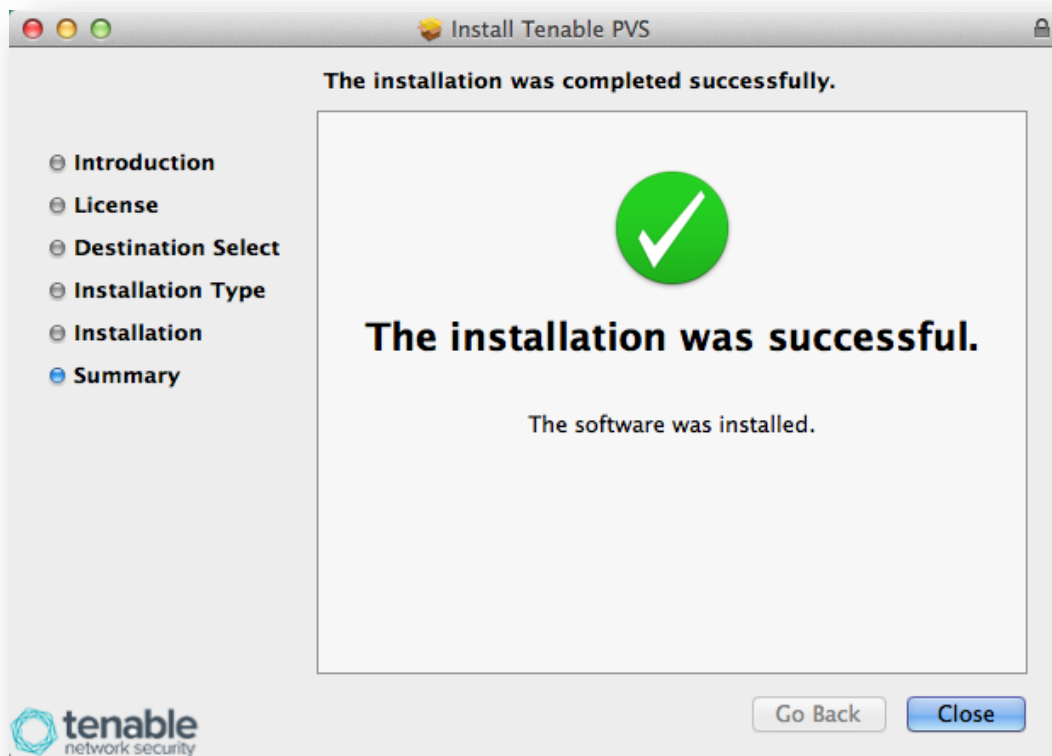
Next, the installation process will ask for authentication for permission to install the software.



The installer will request permission to allow PVS to accept incoming network connections. If this option is denied, PVS will be installed but will have severely reduced functionality.



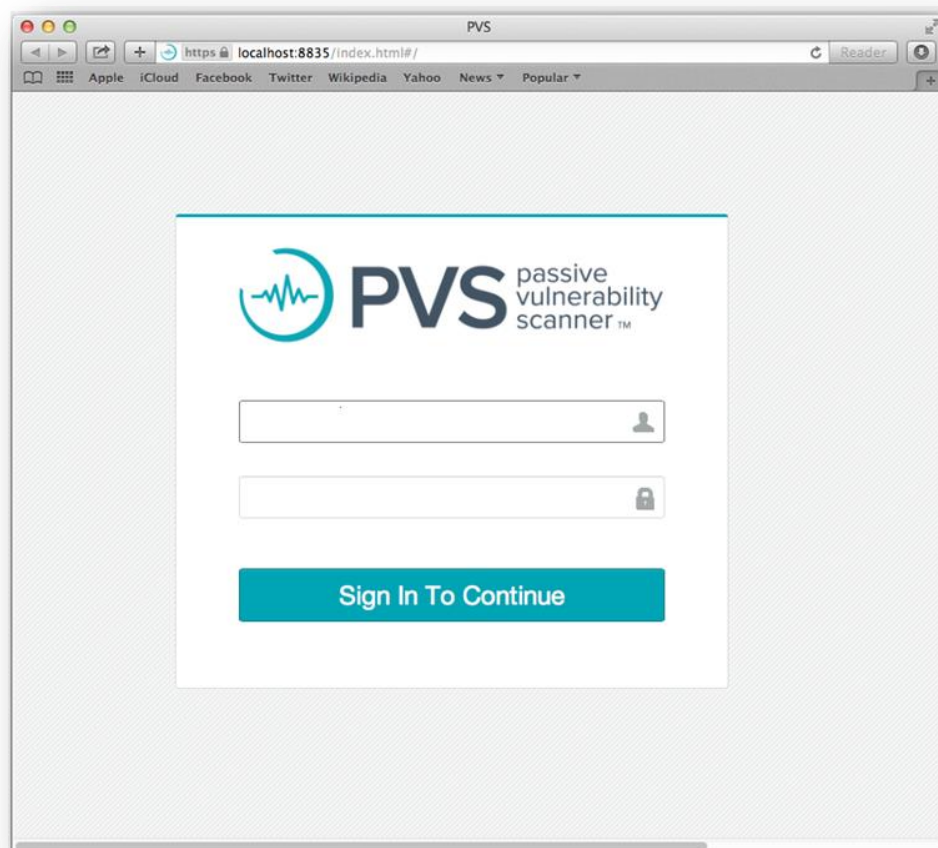
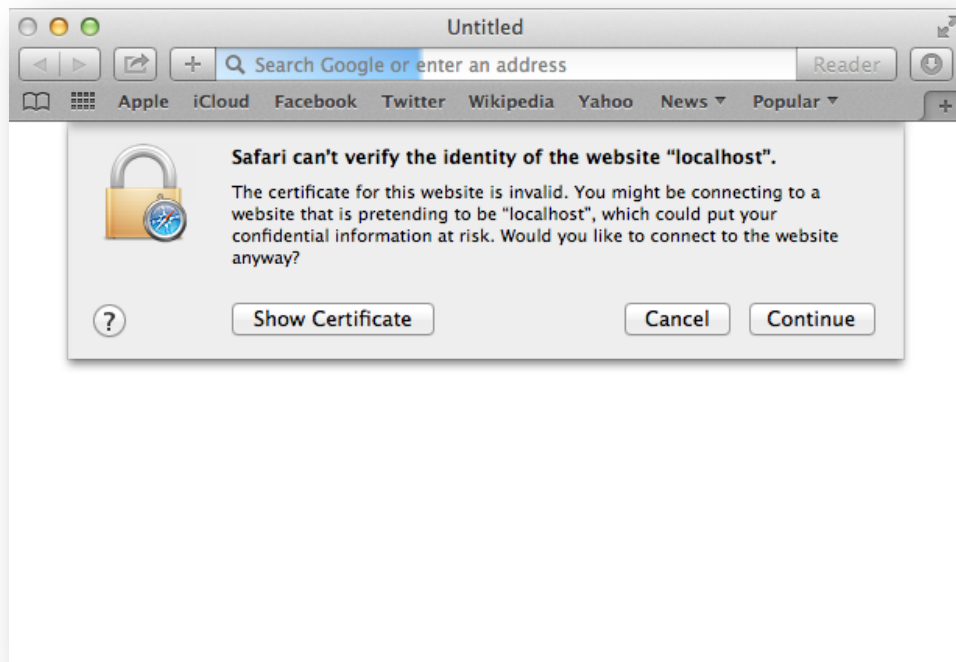
The installation will then be completed.



Immediately after the successful upgrade of PVS, the Installer will automatically launch the Safari browser to allow configuration of PVS for the environment. When presented with the identity dialog box, click "**Continue**".



Once the upgrade process is complete it is suggested to eject the PVS install volume.



Initial Installation

This section describes the steps required for an initial installation of PVS on Linux, Mac OS X, and Windows platforms.

Linux Installation



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



Unless otherwise noted, all commands are performed 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 platform and version:

```
# rpm -ivh pvs-4.0.2-es6.x86_64.rpm
Preparing...                               ##### [100%]
 1:pvs                                     ##### [100%]

[*] PVS installation completed.
#
```

The installation will create the directory `/opt/pvs`, which initially contains the PVS software, default plugins, and directory structure.

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

```
# service pvs start
```

Once the service has started, a connection may be made using a web browser by navigating to `https://<hostname or IP address>:8835` to continue configuration of the PVS. For additional details, see the “[Initial Configuration of the PVS Server](#)” section of this document.



Ensure that organizational firewall rules permit access to port 8835 on the PVS server.

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

http://static.tenable.com/prod_docs/Tenable_Passive_Vulnerability_Scanner_4.x_Software_License_Agreement.pdf

Windows Installation



Ensure that the latest version of the Microsoft Visual C++ 2010 Redistributable Package is installed for your platform prior to PVS software installation.

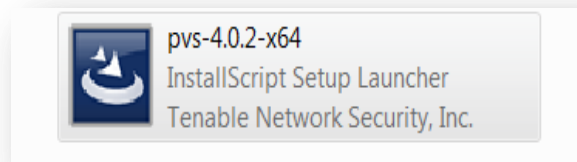


Unless otherwise noted, perform all commands as a local administrator user. When UAC is enabled, right click on the installer program and select “Run as Administrator”.

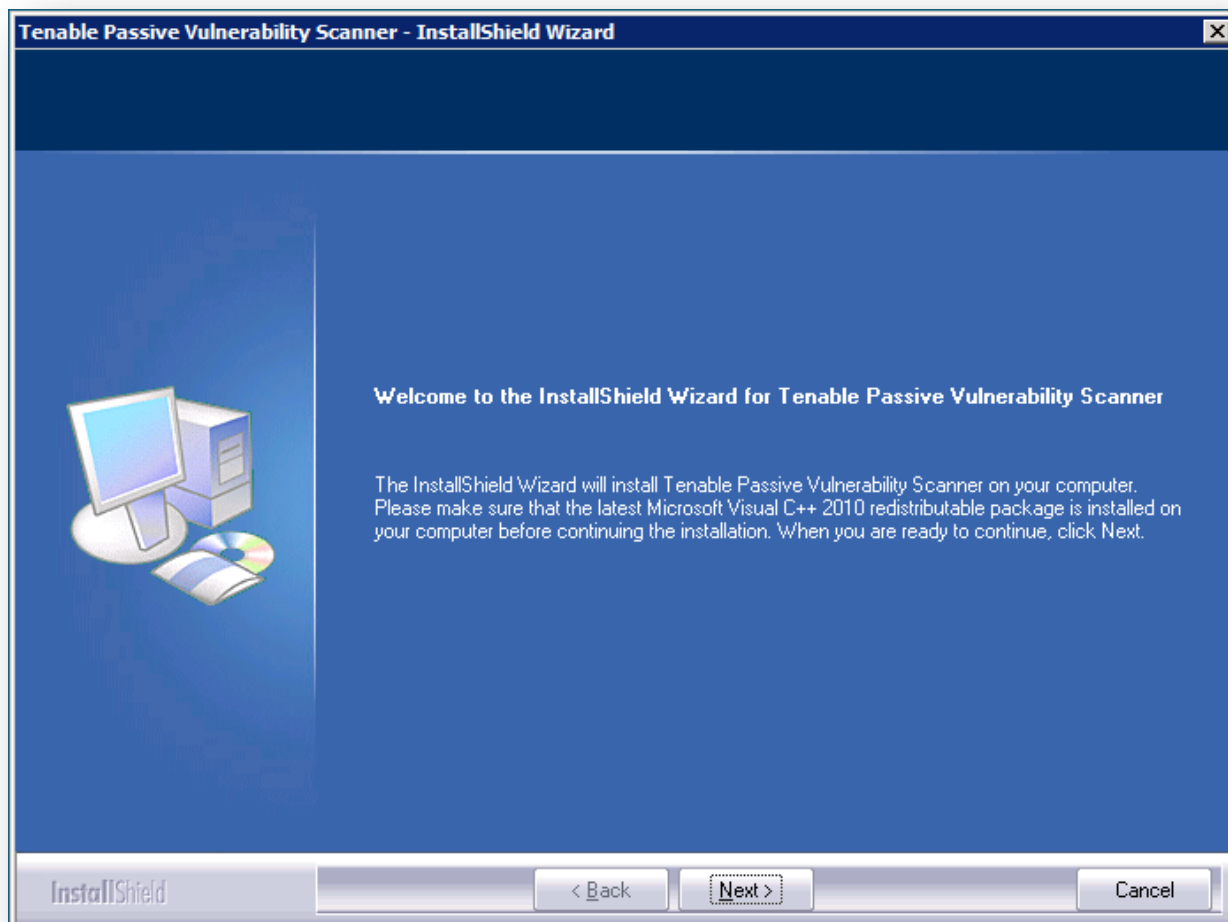


Prior to installing PVS, ensure that any other programs on the system utilizing WinPcap are stopped.

Install the PVS software for Windows by double-clicking on the **.exe** file downloaded from Tenable. Note that the specific filename will vary, depending on your platform and/or version:

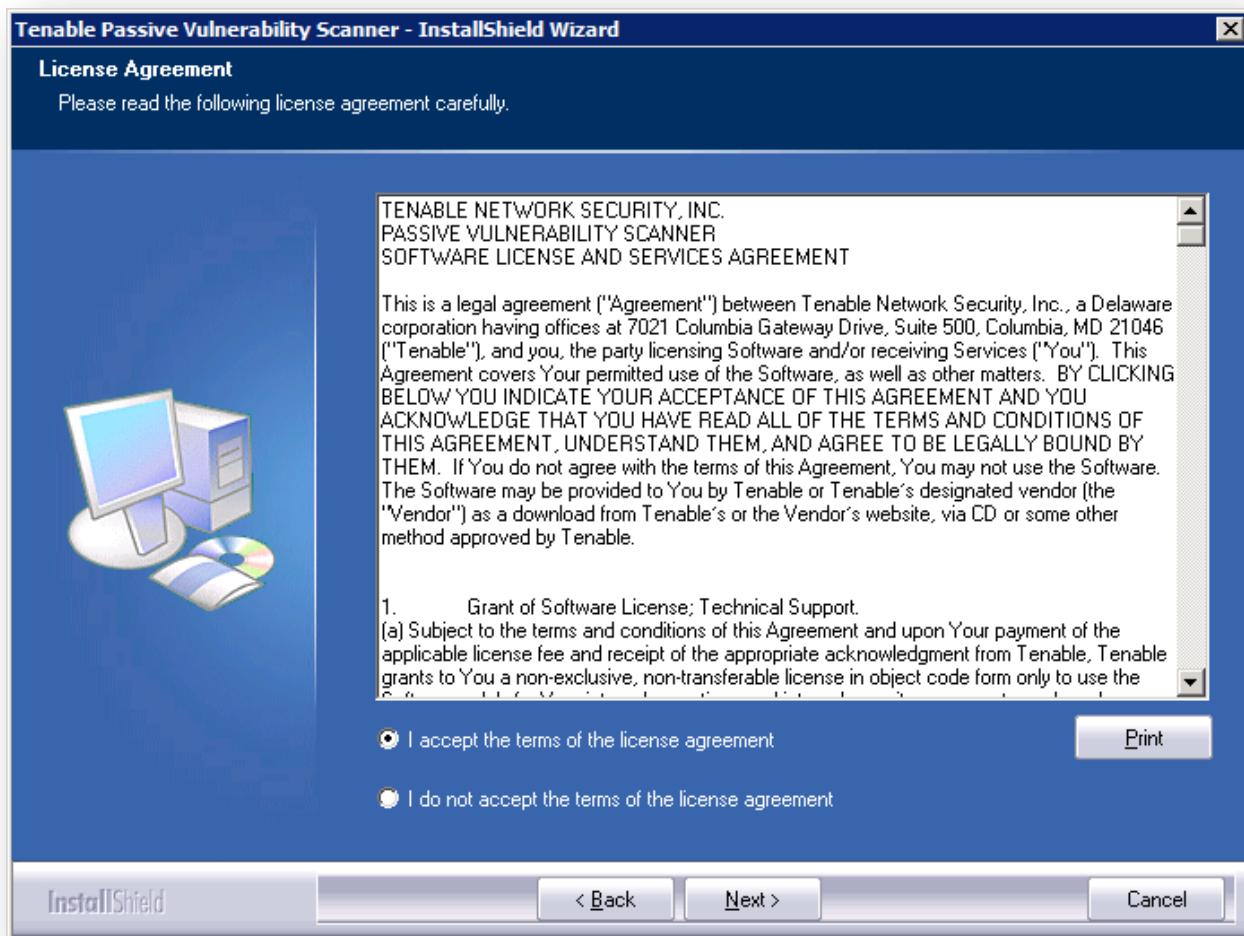


This will start the installation process by launching the InstallShield Wizard:



The InstallShield Wizard will walk you through the installation process and any required configuration. At any point prior to completion, configuration option can be changed by clicking “Back” to go to the previous step. Clicking “Cancel” will abort the installation process completely.

The next screen displays the End User License Agreement (EULA). The text of the agreement can be copied and pasted into a separate document file for reference or it can be printed directly from this interface using the “Print” button. You must agree to the license to continue the installation process and use the PVS.



Next, the installation process will ask for customer information. The user's name and company name are used to customize the installation, but are not related to any configuration options (e.g., for interfacing with SecurityCenter).


Tenable Passive Vulnerability Scanner - InstallShield Wizard [X]

Customer Information
Please enter your information.

Please enter your name and the name of the company for which you work.

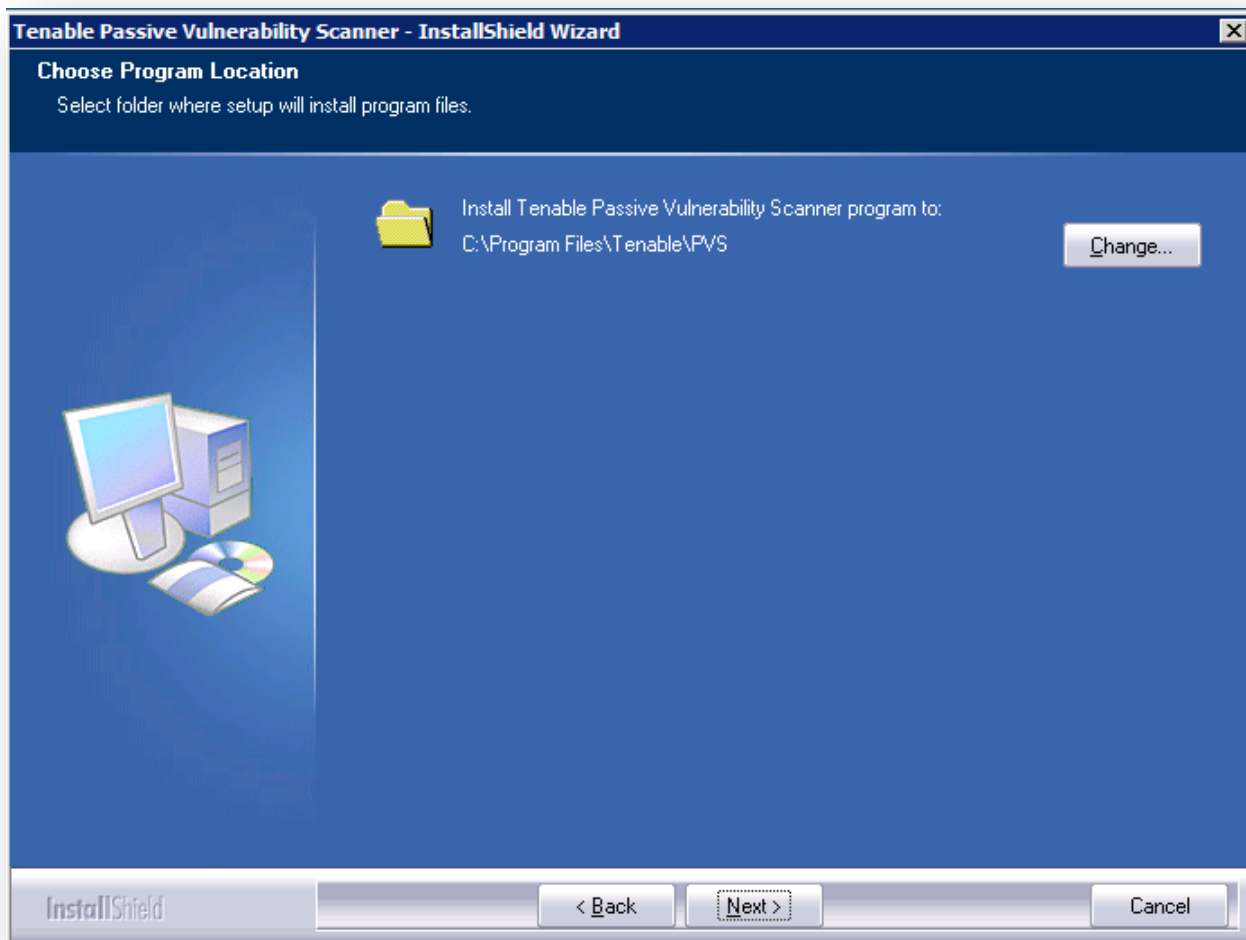
User Name:

Company Name:



InstallShield

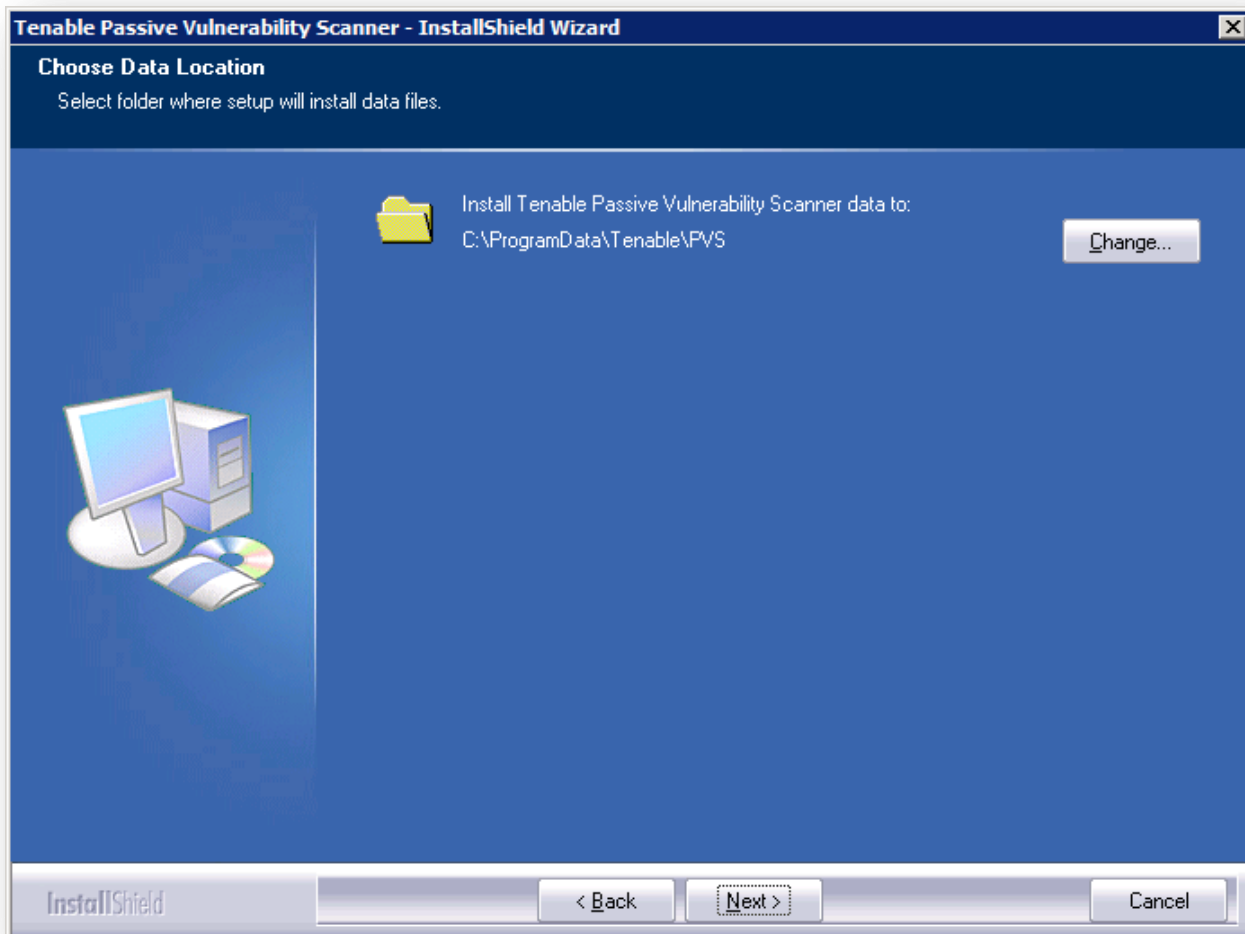
The installation process will then verify the path where the PVS binaries will be installed. Clicking on “Change...” will allow you to specify a custom path:



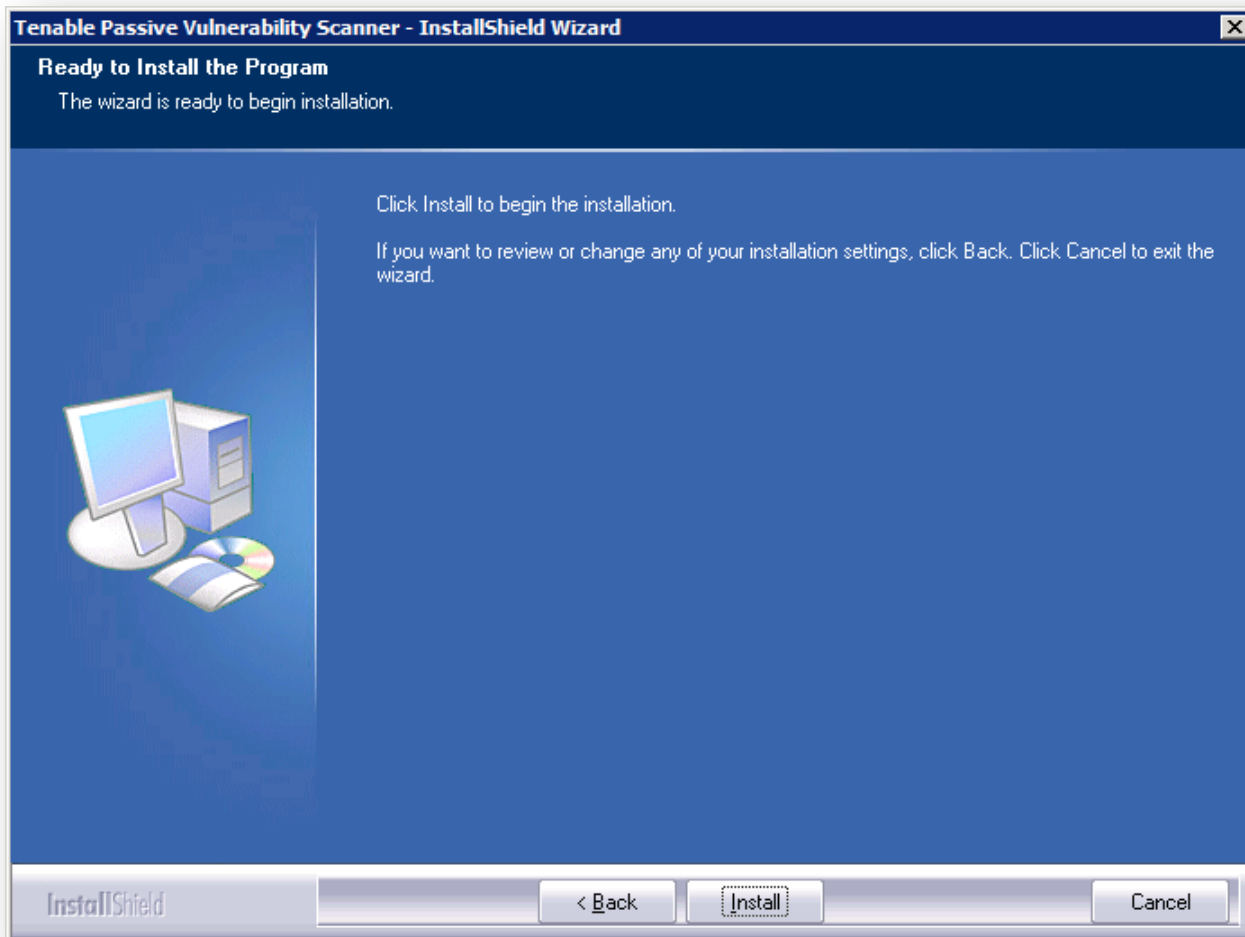
User data generated by PVS can be stored in a separate location, as specified in the next installation option. Clicking “Change...” will allow you to specify an alternate location for user data:



If connecting the PVS to a SecurityCenter, altering the data path will result in the SecurityCenter not being able to retrieve reports.



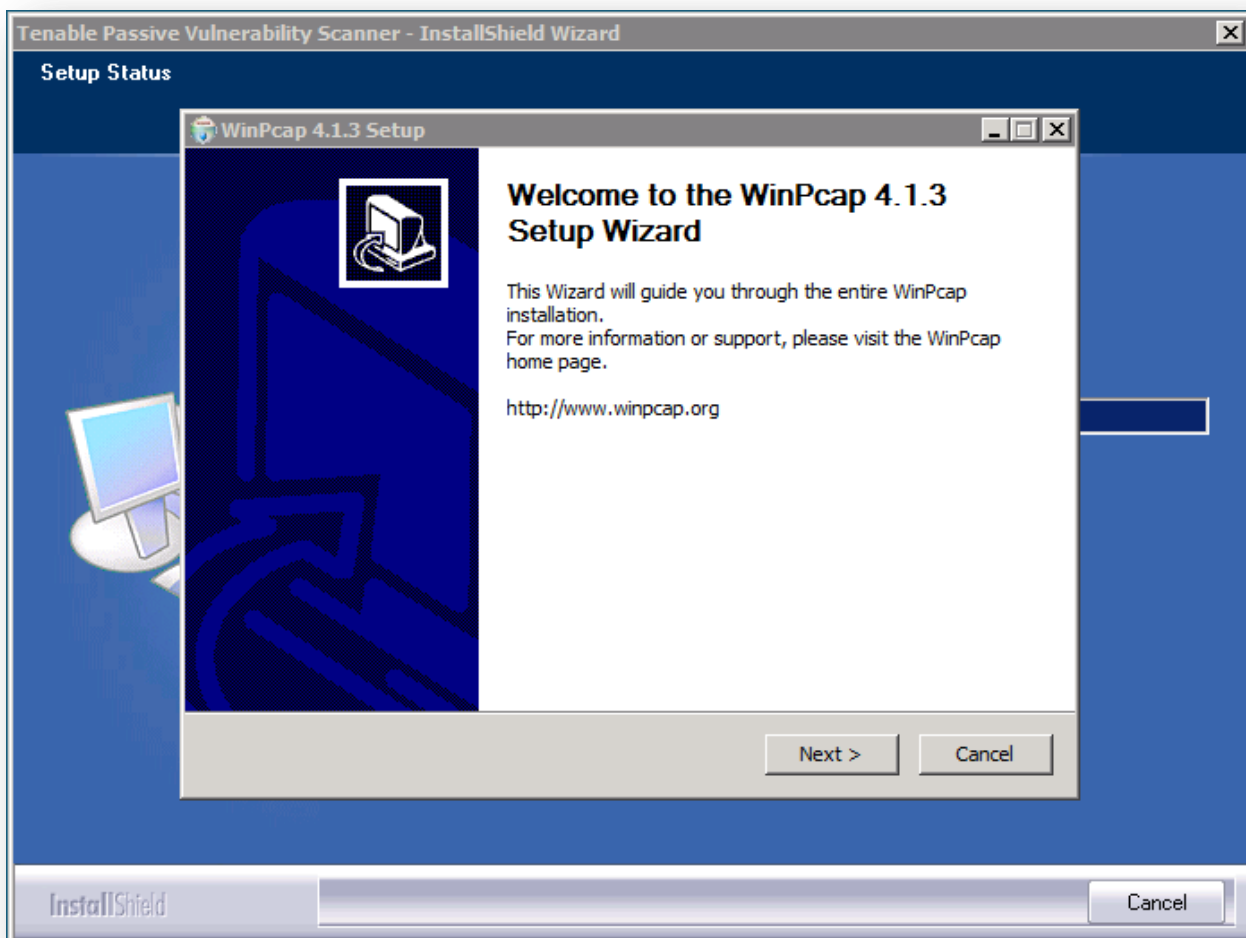
The final screen of the PVS installation configuration options provides the opportunity to go back to make any edits to information supplied on previous screens. If all of the configuration options specified are satisfactory, click “Install...” to complete the PVS installation process.



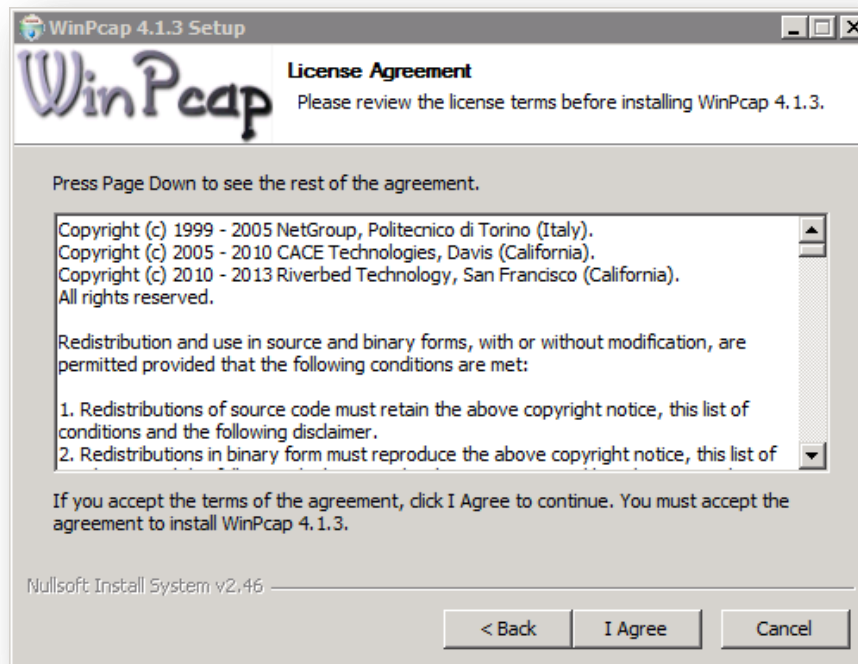
Once PVS has been installed, it will determine if WinPcap is already installed on the system. If the current version of WinPcap is installed and detected, the PVS installation process will ask if you wish to force installation or cancel installation of WinPcap. If it does not detect WinPcap or detects an older version, a second installer will be launched to install or upgrade that software:



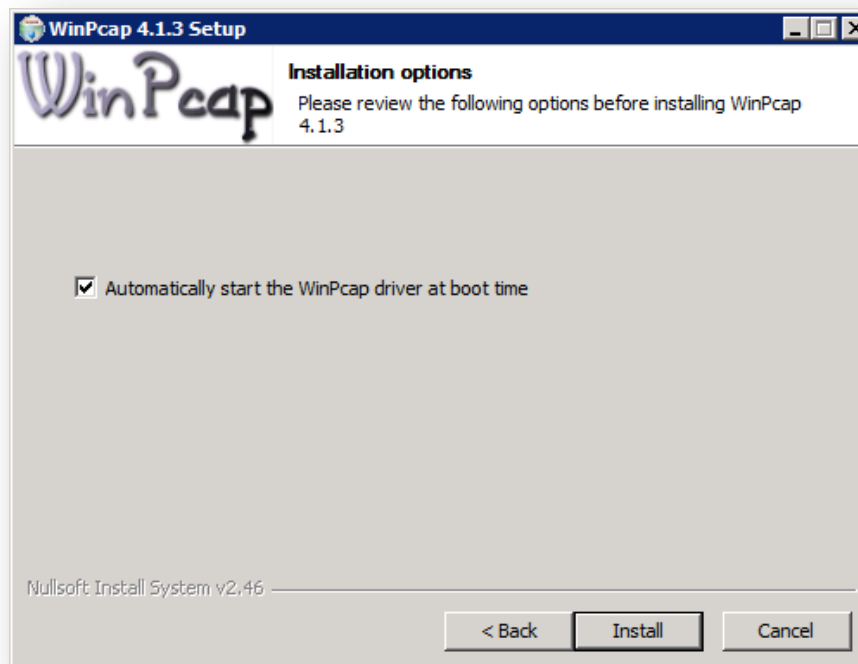
It is advised to use the provided version of WinPcap or newer. PVS has been designed and tested using the supplied version of WinPcap.



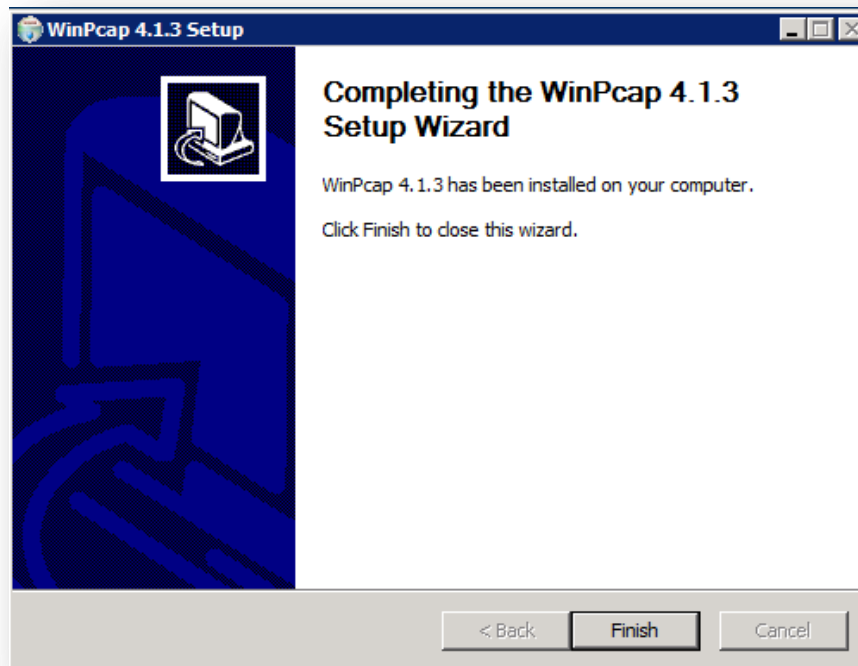
You must agree to the WinPcap end-user license agreement in order to complete the installation:



WinPcap can be configured to start during boot time. This is highly recommended as PVS cannot operate if this software is not running.



Once the license has been agreed to and the configuration option specified, click “Install” to complete the process.



After WinPcap is installed, the PVS installation process is complete. The user will be returned to the desktop.

As a part of the installation process, a new service is installed called “Tenable PVS Proxy Service”. The service is configured to start automatically when the server starts. Either navigate to Services and manually launch the service or restart the system to start the service.

Once the service has started, a connection may be made using a web browser by navigating to `https://<hostname or IP address>:8835` to continue configuration of the PVS. For additional details, see the “[Initial Configuration of the PVS Server](#)” section of this document.



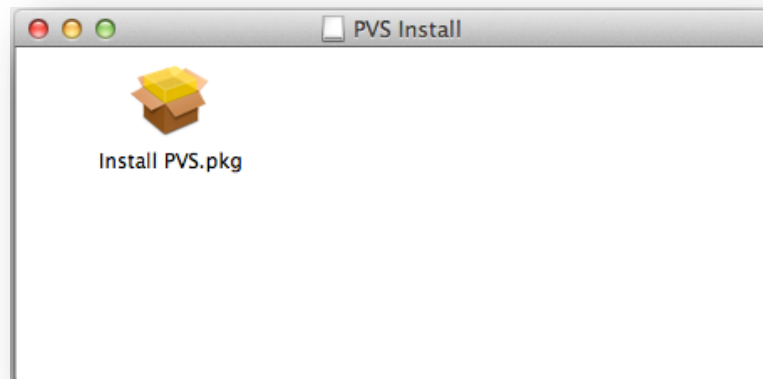
Ensure that organizational firewall rules permit access to port 8835 on the PVS server.

Mac OS X Installation

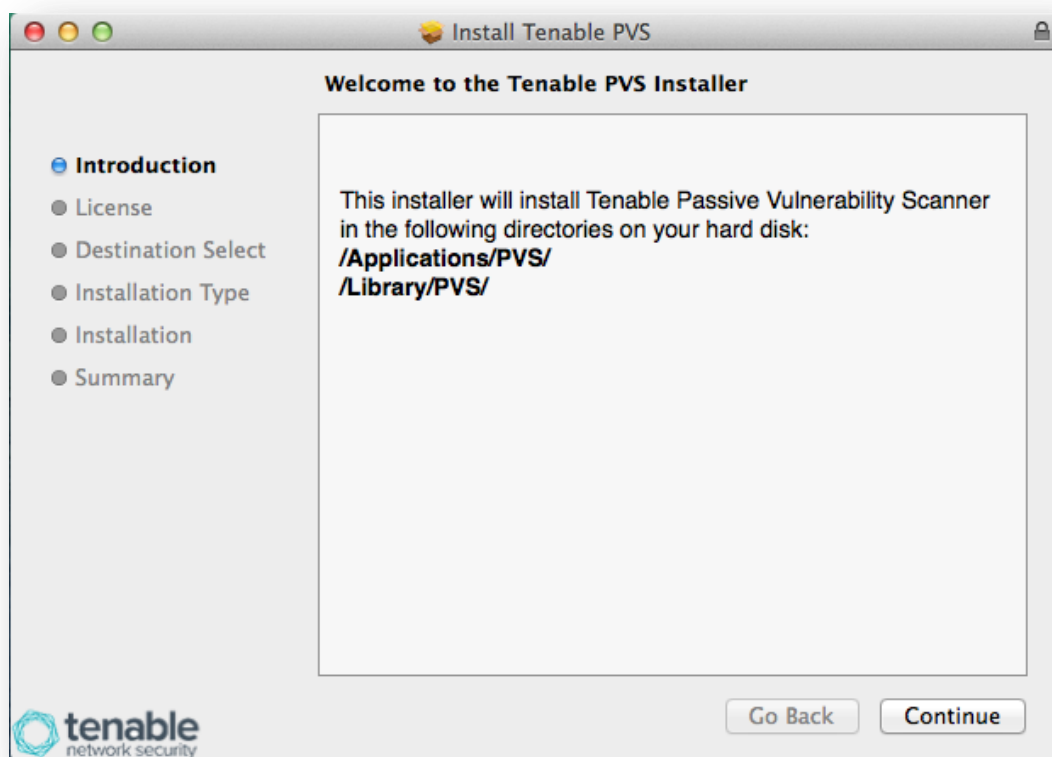
Install the PVS software for Mac OS X by double clicking on the `.dmg` file downloaded from the [Tenable Support Portal](#) to mount the disk image “**PVS Install**”. Note that the specific filename will vary, depending on your platform and/or version:



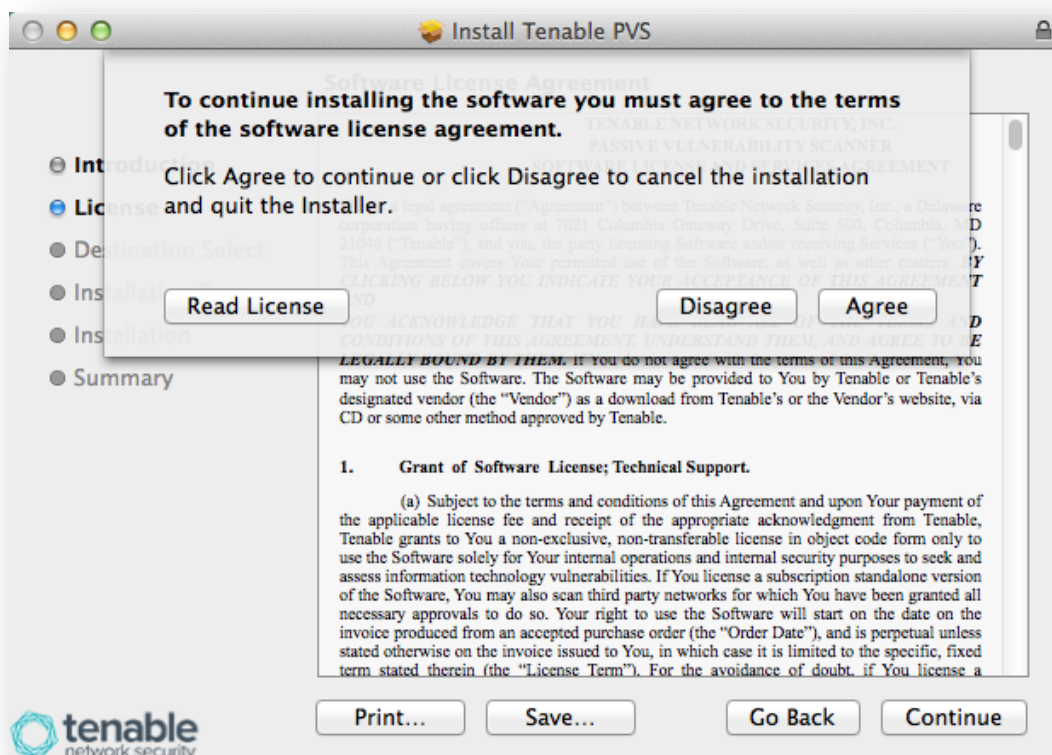
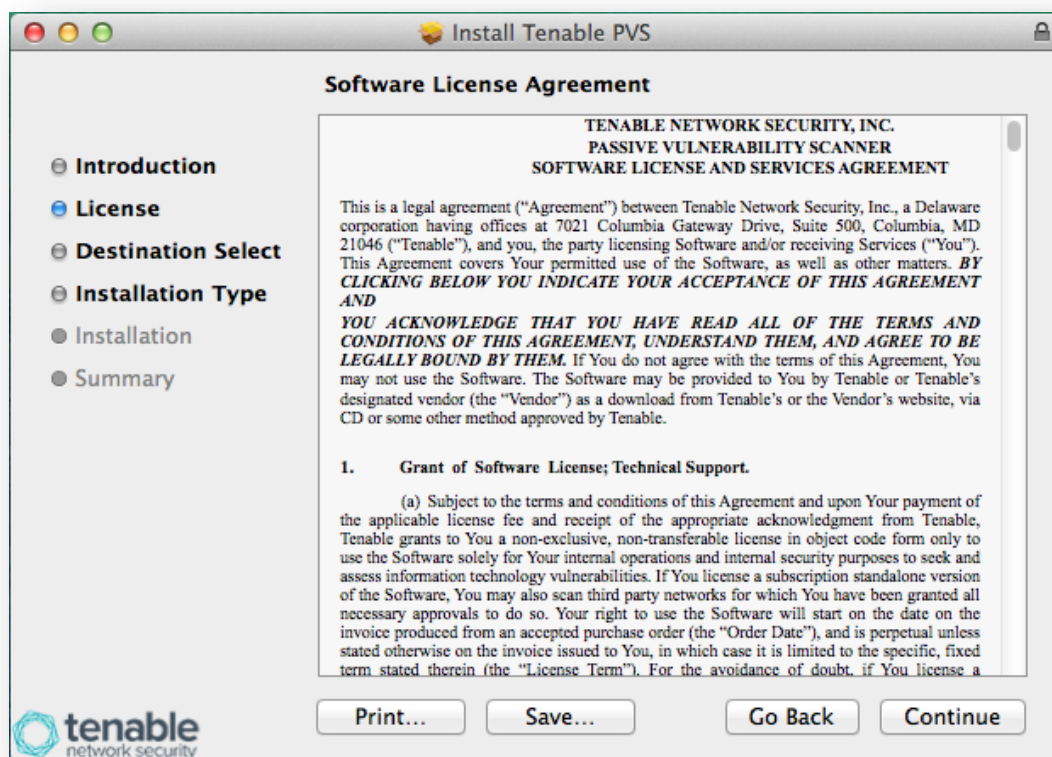
Double click on the Install PVS.pkg file to launch the Installer:



This will launch the Tenable PVS Installer, which will walk you through the installation process and any required configuration. At any point prior to completion, configuration options may be changed by clicking “Back” to go to the previous step. Clicking “Cancel” will abort the installation process completely.



The next screen displays the End User License Agreement (EULA). The text of the agreement can be copied and pasted into a separate document file for reference, saved using the “**Save...**” button, or it can be printed directly from this interface using the “**Print...**” button. You must agree to the license to continue the installation process and use PVS.



Click **Install** to begin the installation:



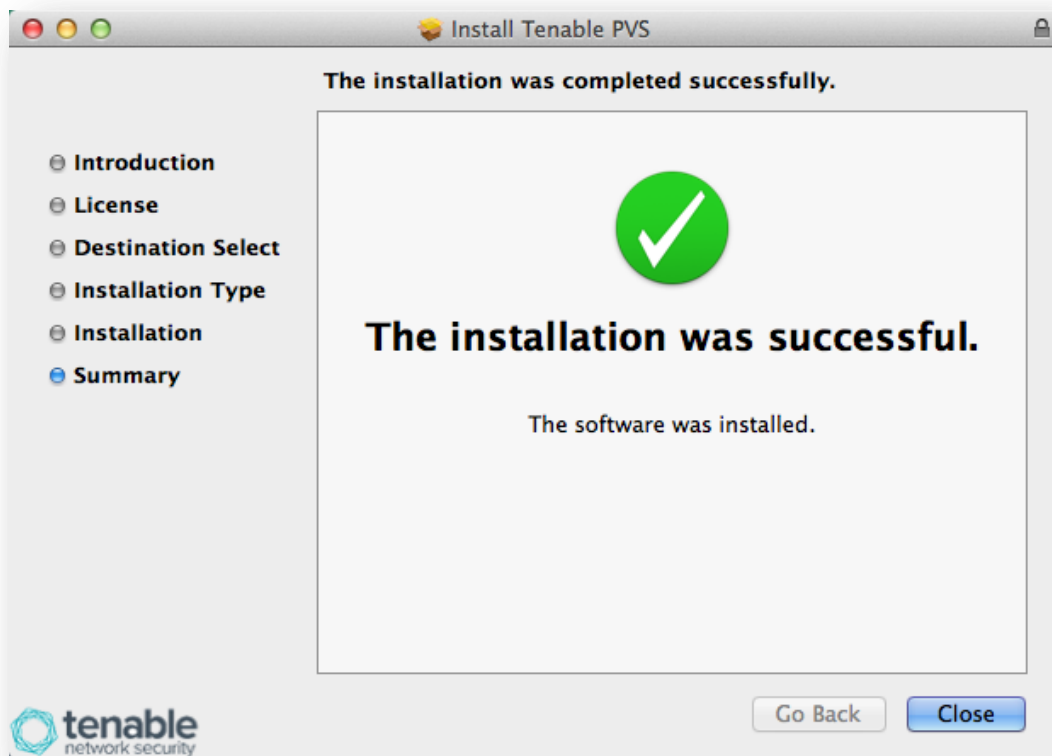
Next, the installation process will ask for authentication for permission to install the software.



The installer will request permission to allow PVS to accept incoming network connections. If this option is denied, PVS will be installed but will have severely reduced functionality.



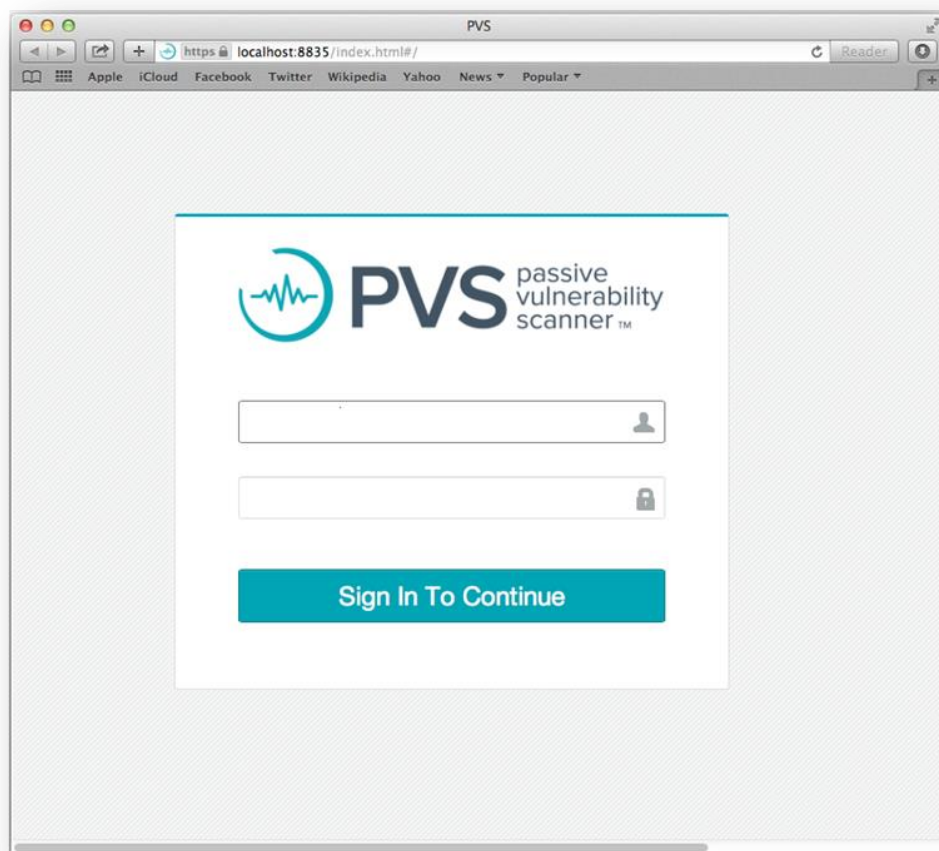
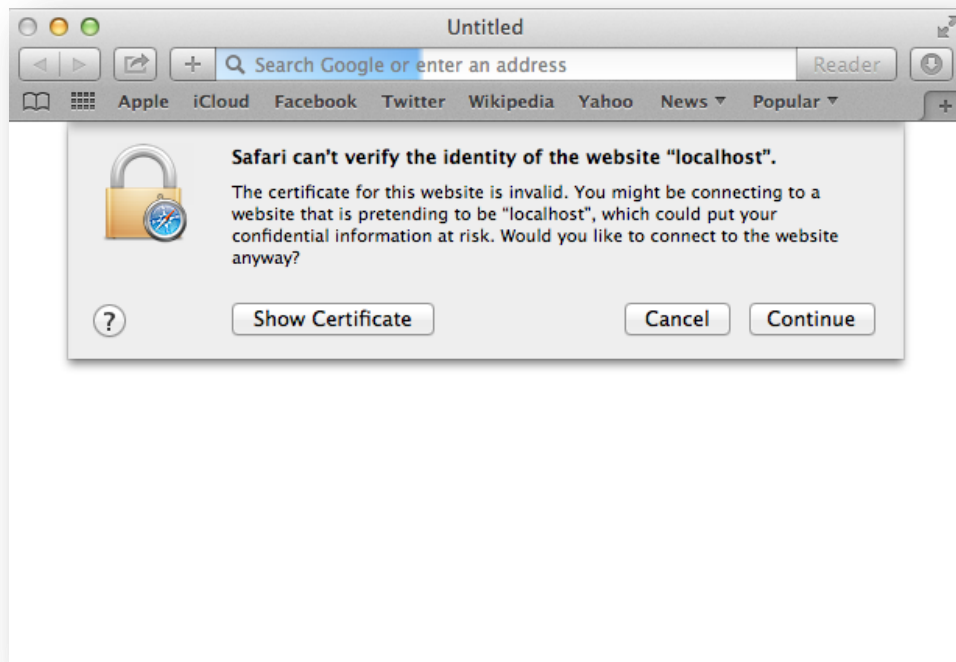
The installation will then be completed.



Immediately after the successful installation of PVS, the Installer will automatically launch the Safari browser to allow configuration of PVS for the environment. When presented with the identity dialog box, click "**Continue**".

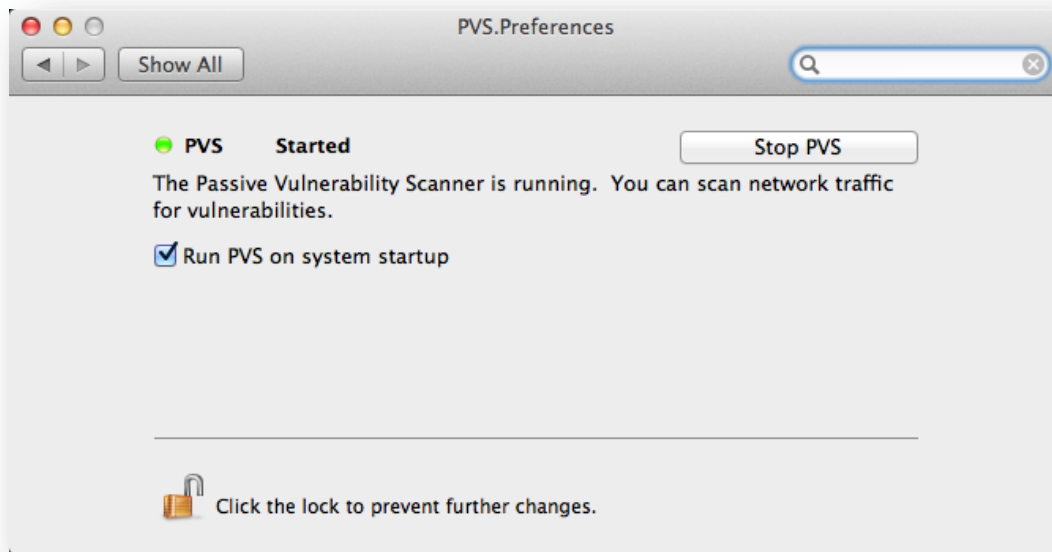


Once the installation process is complete it is suggested to eject the PVS install volume.



Starting and Stopping PVS for Mac OS X

The preferred method to start and stop the PVS service on Mac OS X is to use the “**PVS Preferences**” tab under “**System Preferences**”. Once launched, the following window will be displayed. To make changes to any of the states of PVS a root user or equivalent privileges must be used.



The window displays if the PVS is started or stopped and provides a button to start or stop the service. Additionally, a checkbox is available to enable or disable PVS from running on system startup.

Removing PVS

This section describes the steps required to remove PVS from Linux, Mac OS X, and Windows platforms.

Removing PVS for Linux

To remove versions of PVS for Linux, first stop the PVS services with the command “**service pvs stop**”. You will need to know what name the PVS 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 is registered as, run the following command:

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

This will produce output similar to the following:

```
pvs-4.0.x-es6.x86_64
```

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

```
# rpm -e pvs-4.0.x-es6.x86_64
```

Some files that are user created and/or modified are not removed with the previous command. To completely remove the remaining files, run the following command:

```
# rm -rf /opt/pvs
```

Removing PVS for Windows

To remove PVS, under the Control Panel open “**Programs and Features**” or “**Add or Remove Programs**”, depending on the Windows version. Select “**Tenable Passive Vulnerability Scanner**” and then click on the “**Change/Remove**” button.

This will open the InstallShield Wizard. Follow the directions in this wizard to completely remove PVS. If you select “Yes”, the PVS program and its features will be removed from the system.



In some cases, scan data and user modified files may be left in the `c:\program files\Tenable\PVS` and `c:\programdata\Tenable\PVS` directories. These files must be manually removed.

Additionally, the WinPcap program must be removed separately.

Removing PVS for Mac OS X

To remove PVS, first stop the PVS services. Then delete the following directories (including subdirectories) and files using the command line:

```
/Library/LaunchDaemons/com.tenablesecurity.pvs*  
/Library/PVS  
/Library/PreferencePanes/PVS*  
/Applications/PVS
```




If you are unfamiliar with Unix command line usage on a Mac OS X system, please contact Tenable Support for assistance.


There are freeware and shareware tools such as “DesInstaller.app” (<http://www.macupdate.com/info.php/id/7511>) and “CleanApp” (<http://www.macupdate.com/info.php/id/21453/cleanapp>) that can also be used to remove PVS. Tenable has no affiliation with these tools and they have not been specifically tested for removing PVS.


Initial Configuration of the PVS Server

After the installation and startup of PVS for Linux, Windows, or Mac OS X is complete, configuring the PVS server follows the same steps for all platforms.

Begin by connecting to the host using a web browser to the host URL at `https://<IP address or hostname>: 8835`. The default username is “**admin**” and the initial password is “**admin**”.

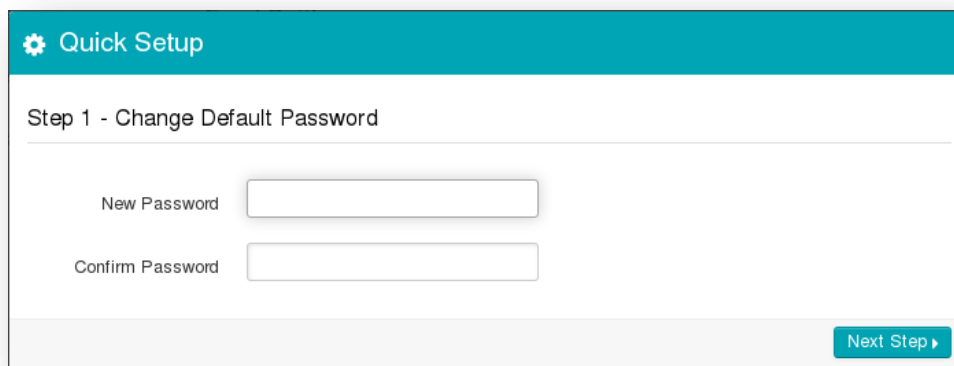
 **PVS** passive vulnerability scanner™

Username 

Password 

Sign In To Continue

After the initial login, a quick setup process begins. The first step is to change the default admin password. At a minimum, the new password must be at least 5 characters long, contain one capital letter, one lowercase letter, one digit, and one special character from the list of !@#\$%^&*().

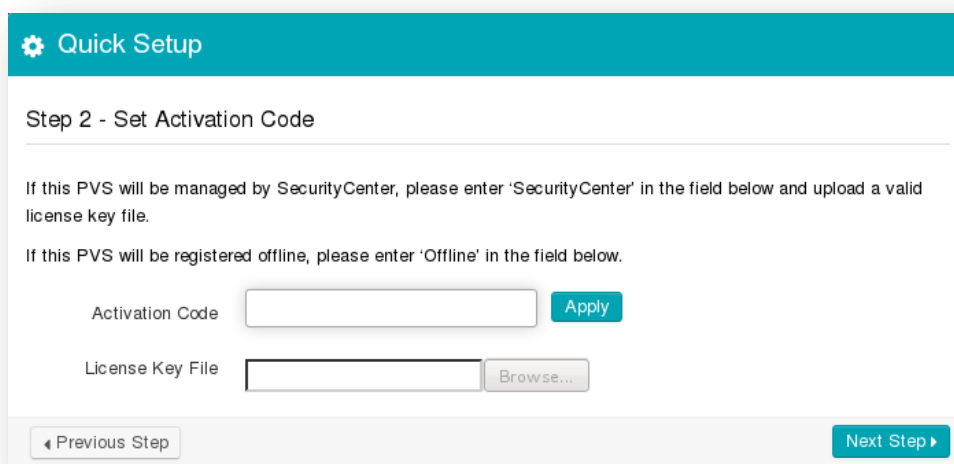


The screenshot shows the 'Quick Setup' window with a teal header. Below the header, the title 'Step 1 - Change Default Password' is displayed. There are two input fields: 'New Password' and 'Confirm Password'. A 'Next Step' button with a right-pointing arrow is located at the bottom right of the form.

The second step is to enter an Activation Code or license key. An Activation Code is required if the PVS will be acting as a standalone device. If it is to be managed by SecurityCenter, enter "SecurityCenter" without the quotes in the Activation Code box and upload the License Key File obtained from Tenable. If the PVS is to be registered in an Offline mode, enter "Offline" in the Activation Code box. Once the proper information is entered, proceed to the next step by clicking the "Next Step" button.



If this is part of an upgrade, the License Key File requested is the `tenable.key` file that was saved during the upgrade process.



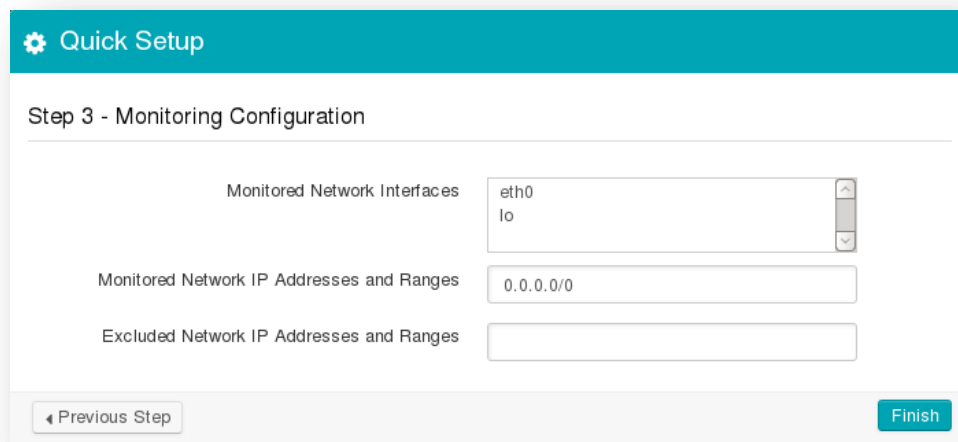
The screenshot shows the 'Quick Setup' window with a teal header. Below the header, the title 'Step 2 - Set Activation Code' is displayed. There is instructional text: 'If this PVS will be managed by SecurityCenter, please enter 'SecurityCenter' in the field below and upload a valid license key file.' and 'If this PVS will be registered offline, please enter 'Offline' in the field below.' There are two input fields: 'Activation Code' and 'License Key File'. The 'Activation Code' field has an 'Apply' button next to it. The 'License Key File' field has a 'Browse...' button next to it. At the bottom left is a 'Previous Step' button with a left-pointing arrow, and at the bottom right is a 'Next Step' button with a right-pointing arrow.



For more information about installing the Activation Code and performing offline plugin updates, please refer to the [PVS Activation Code Installation](#) document.

The third and final step of the Quick Setup is to define the **Monitoring Configuration**. On this screen, the **Monitored Network Interfaces** are selected from those that PVS has identified. One or more of the defined interfaces may be

selected. The “**Monitored Network IP Addresses and Ranges**” option determines the IP address ranges that PVS will monitor. The “**Excluded Network IP Addresses and Ranges**” option determines the IP address ranges that PVS will not monitor. Both fields accept IPv4 and/or IPv6 CIDR address definitions. When multiple addresses are used, separate the entries using commas.



Quick Setup

Step 3 - Monitoring Configuration

Monitored Network Interfaces: eth0, lo

Monitored Network IP Addresses and Ranges: 0.0.0.0/0

Excluded Network IP Addresses and Ranges:

Previous Step Finish

Once the Quick Setup steps are completed, log out of the web interface. The PVS service must be stopped and started for the settings to take effect. PVS is then ready to use. The user may log in and is presented with the default “Monitoring” page. Once PVS has started monitoring traffic, the page will display a list of hosts by using a bar chart that indicates the number of vulnerabilities each host has, color coded to the severity level.

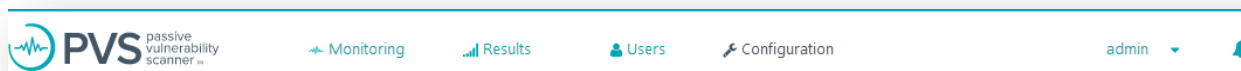
Using the PVS Interface

The primary feature change to PVS 4.0 is the ability to analyze data and configure PVS utilizing a HTML5 front end. The interface for PVS has been tested and works on browsers that support HTML5, including the latest available versions available at the time of this writing including Microsoft Internet Explorer 9-11 (version 9 being the minimum supported), Firefox 24 and later, Opera 16 and later, and Google Chrome 30 or later. Other browsers that support HTML5 have been reported to work, but have not been fully tested by Tenable. This allows PVS to be placed on networks, monitor traffic, and have the results provided to users without the need for SecurityCenter or a third party tool to analyze the data.

Once logged into PVS, the user is initially presented with the Monitoring page, which includes several navigation options across the top.

Navigation

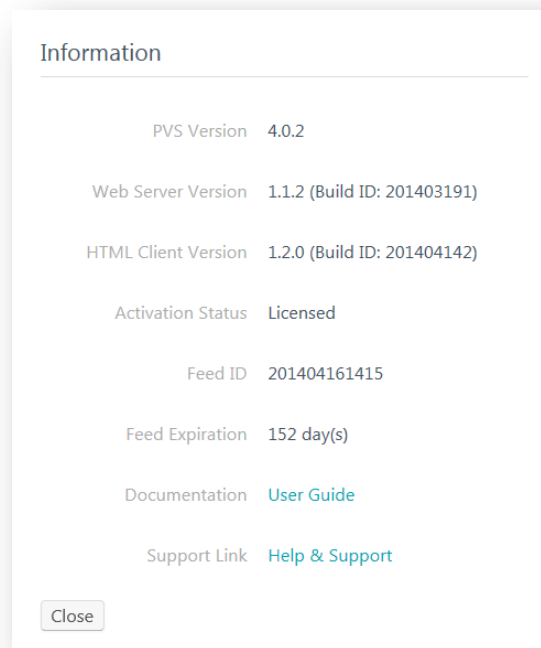
Navigation through the PVS interface starts with the options available across the top menu bar.



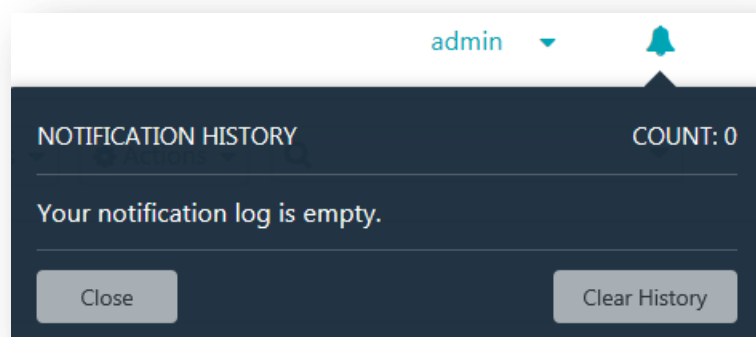
The four options are titled **Monitoring**, **Results**, **Users**, and **Configuration**. The Users and Configuration pages are only available to Administrator level users. From these pages, all of the primary analysis and maintenance tasks may be performed. Clicking any of the page names will present the user with that page.

The right side of the menu bar indicates the username of the currently logged in user. Clicking the name will present a drop-down menu with three selections. The first is to change the user’s password (which must contain 5 characters, including at least one each of a lowercase letter, uppercase letter, digit, and a special character). Information about the

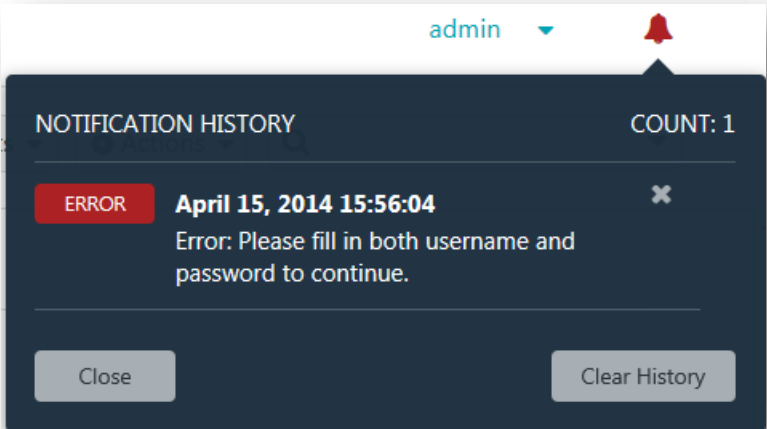
PVS version, Web Server Version, HTML client version, links to support and documentation, and license and feed status can be viewed by selecting the **Help & Support** link. Selecting the “**Sign Out**” link will end the current user’s session.



An additional feature introduced with the PVS HTML client 1.2 is the Notification Center, which is in the shape of a bell and is found to the far right on the menu bar. The bell icon is a color-coded graphical representation of **notifications**, **errors**, and **system information** generated by PVS. If no alerts are present, all messages are of an informational level, or if all alerts have been read, then the bell icon will be color-coded **blue**. The list will retain a maximum of 1,000 alerts and will stop adding more until they are cleared.



The notification icon will change from **blue** to **red** making the user aware that there are unread alerts in the notification area. Each individual notification can be removed by clicking the “X” to the right of the description of each event, or the entire history can be deleted by clicking the “Clear History” button in the lower right corner of the notification pane.



Notifications are not preserved between sessions. Unread notifications will be removed from the list when the user logs out.

Monitoring

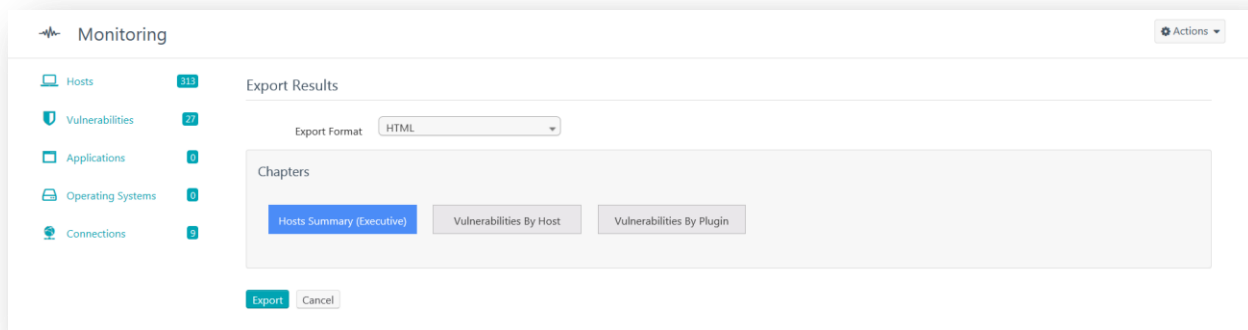
The monitoring page provides a centralized view of the vulnerabilities discovered by PVS. From this page, the vulnerabilities may be viewed in a variety of different ways including by host, vulnerability, application, operating system, and connections. The results may also be exported to different formats for use in other programs.



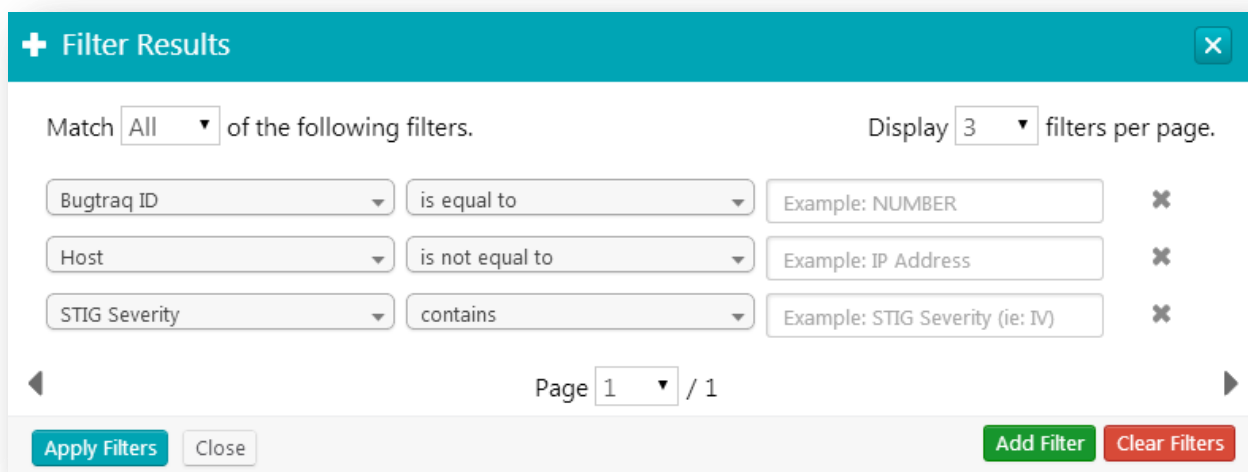
Across all of the viewable methods available on the Monitoring page, filter options are available to increase granularity when viewing results.

The “**Sort Hosts**” drop-down provides an option to sort the host either by hostname or by the count of severity items found on the hosts. These sorting options can be displayed in either ascending or descending order.

The **Actions** drop-down tab allows results to either be exported in Nessus V2 format, CSV format for use in other programs, HTML for viewing the report in a web browser, or for results to be deleted. For the HTML export option, the exported file may contain one or more chapters including Hosts Summary, Vulnerabilities by Host, and Vulnerabilities by Plugin. Once the file type and any applicable options are selected, click the “Export” button to generate and download the report. Large amounts of data will take longer to export.



The Filter option text field allows for quick filtering based on entered text for the current view of the Monitoring page. Select the down arrow on the right of the filter to view a rich selection of options based on discovered vulnerability information to filter the results based on entered values. Results are displayed based on a match of “Any” or “All” entered fields. The search field contains example hints when empty, but if an incorrect filter value is entered, the field will display a red border. If values are still incorrect when applying the filters, a notification will be displayed in the upper right corner of the screen to notify the user of errors that need to be corrected before filtering can be applied.



Name	Description
Bugtraq ID	Filter the results of discovered vulnerabilities based on their Bugtraq identification.
CPE	Filter the results of discovered vulnerabilities based on their CPE identifier.
CVE	Filter the results of discovered vulnerabilities based on their CVE identifier.
CVSS Base Score	Filter the results of discovered vulnerabilities based on the base CVSS score as reported by the vulnerability plugins.
CVSS Temporal Score	Filter the results of discovered vulnerabilities based on the temporal CVSS score as reported by the vulnerability plugins.
CVSS Temporal Vector	Filter the results of discovered vulnerabilities based on the CVSS temporal vector as reported by the vulnerability plugins.
CVSS Vector	Filter the results of the discovered vulnerabilities based on the CVSS vector.
Host	Filter the results of the discovered vulnerabilities based on the discovered IP address of the device.
IAVA ID	Filter the results of the discovered vulnerabilities based on the IAVA ID of the vulnerability.
IAVB ID	Filter the results of the discovered vulnerabilities based on the IAVB ID of the vulnerability.
IAVT ID	Filter the results of the discovered vulnerabilities based on the IAVT ID of the vulnerability.
OSVDB ID	Filter the results of the discovered vulnerabilities based on the discovered OSVDB identifier.
Plugin Description	Filter the results of the discovered vulnerabilities based on text available in the description of the vulnerabilities.
Plugin Family	Filter the results of the discovered vulnerabilities based on the family of vulnerabilities discovered.
Plugin ID	Filter the results of the discovered vulnerabilities based on the plugin ID that identified the vulnerability.
Plugin Name	Filter the results of the discovered vulnerabilities based on text contained in the name of the plugin that discovered the vulnerability.
Plugin Output	Filter the results of the discovered vulnerabilities based on text contained in the output of the plugin that discovered the vulnerability.
Port	Filter the results of the discovered vulnerabilities based on the port the vulnerability was discovered on.
Protocol	Filter the results of the discovered vulnerabilities based on the detected protocol: tcp, udp, or icmp.
Risk Factor	Filter the results of the discovered vulnerabilities based on the identified risk factor.

See Also	Filter the results of the discovered vulnerabilities based on the text available in the “See Also” field of the plugin.
Solution	Filter the results of the discovered vulnerabilities based on text available in the solution section of the plugin.
STIG Severity	Filter the results of the discovered vulnerabilities based on STIG severity level in the plugin.
Synopsis	Filter the results of the discovered vulnerabilities based on text available in the synopsis section of the plugin.

The **Hosts** tab on the **Monitoring** page displays a list of the discovered hosts along with a bar chart that is color coded to indicate the variety of severity levels detected on the host, as well as the total number of each level if there is room to display it.



Selecting a host from the list will display the vulnerabilities discovered by severity order by default from “Critical” down to “Informational”. The names of the vulnerability, vulnerability family, and the number discovered will be listed. Selecting any of these from the list will display vulnerability details including a synopsis, description, solution, plugin information, risk information, reference information, and affected ports and services for the host. Selecting the “**Remove**” button will remove the detected vulnerability from the results.

The **Vulnerabilities** tab provides a list of the vulnerabilities detected by PVS. The initial sort is by severity level from “Critical” to “Informational”. The vulnerability names are displayed along with the family and the number of detected vulnerabilities. Selecting any of these from the list will display vulnerability details including a synopsis, description, solution, plugin information, risk information, reference information, and affected hosts. Selecting an affected host will display the ports, plugin information, and affected service as available. Selecting the “**Remove**” button will remove the detected vulnerability from the results.

Vulnerabilities > Plugin 3875

Teredo IPv6 Client Detection
Remove

Synopsis

The remote host is running software that should be authorized with respect to corporate policy.

Description

The remote client is a Teredo client. Teredo allows clients to tunnel IPv6 traffic over IPv4. The protocol operates over UDP port 3544 and the RFC draft is sponsored by Microsoft. Teredo client puts the IPv6 data inside of an IPv4 packet and sends it to a gateway machine. The gateway machine then strips away the IPv4 header and delivers the IPv6 packet. Given this, Teredo can be used to circumvent firewall rules.

Solution

Ensure that this sort of functionality is authorized with respect to existing policies and guidelines.

Affected Host List (3)

[Redacted]	1
[Redacted]	1
[Redacted]	1

Plugin Details

Severity: Medium
ID: 3875
Family: Generic

Risk Information

CVSS Base Score: 5.0
CVSS Vector: CVSS2#AV:N/AC:L/Au:N/C:N/In:P

The **Applications** tab provides a list of discovered applications and their affected vulnerabilities. The summary page displays a list sorted by the highest reported severity and includes the name and the number of discoveries. Selecting a particular application will present a list of affected hosts. Clicking on a host will display the affected port and protocol, the software and version, and the service as available.

The **Operating Systems** tab provides a list of discovered operating systems. The summary page lists the severity, operating system name as detected, and the number of discoveries. Selecting an operating system name from the list will display the severity, the version of the operating system, and service as available.

The **Connections** tab displays the Client Connection Summary list. A list of hosts is displayed and clicking on any of the hosts provides information on connections from the host to other hosts, which port(s) were used, and the services if known. Selecting the “**Remove**” button will remove the detected vulnerability from the results.

Results

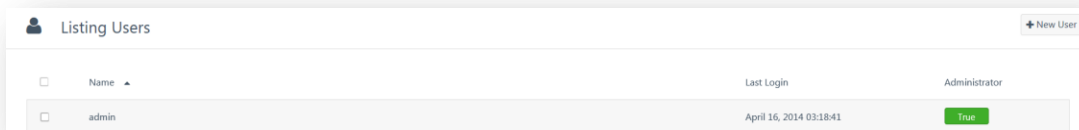
The **Results** page contains snapshots of the monitored data, results from reading in PCAP files, and uploaded PVS reports. The Monitored Data snapshots are generated regularly based on the Report Frequency setting. They are stored until deleted or the Report Lifetime setting is put into effect. When a result grouping is selected, it may be viewed using the same analysis tools described in the previous Monitoring section.

Listing Results
Upload
Filter Results
Clear Filter

<input type="checkbox"/>	Result Title	Last Updated	Result Type
<input type="checkbox"/>	Monitoring Snapshot - Apr 16 2014 03:43:40	April 16, 2014 03:43:40	Snapshot
<input type="checkbox"/>	Monitoring Snapshot - Apr 16 2014 03:28:40	April 16, 2014 03:28:40	Snapshot
<input type="checkbox"/>	Monitoring Snapshot - Apr 16 2014 03:13:40	April 16, 2014 03:13:40	Snapshot

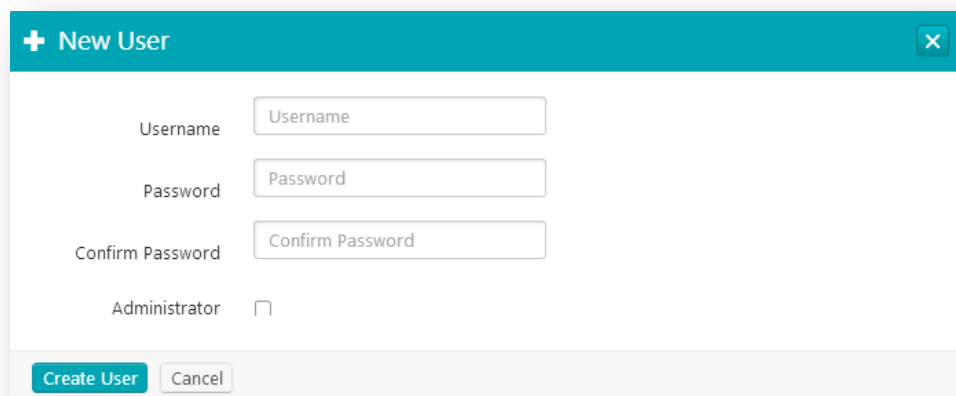
Users

The **Users** screen provides a list of the available users on the PVS server. This screen is only available to Administrator level users. User accounts may be managed from this screen. The list includes a user's login ID, date of last login, and a true/false message indicating if the account is an administrator or not. Hovering over a user account will display an "x" on the right hand side. When clicked, a dialog box opens asking to confirm the deletion of the user. Logged in users may not delete their own account. Multiple users may be selected using the checkboxes on the left side of the "Name" list. Checking any of these boxes will cause a new "**Actions**" menu to appear to the left of the "**New User**" menu allowing the user to select "**Delete User**" from the Actions drop-down menu.



<input type="checkbox"/>	Name	Last Login	Administrator
<input type="checkbox"/>	admin	April 16, 2014 03:18:41	True

A new user may be created by clicking the "New User" button. A dialog box will open and prompt for the new user's information. The username is case sensitive and the password must conform to the minimal PVS password policy. Selecting the Administrator checkbox will create the user with those rights.



+ New User

Username

Username

Password

Password

Confirm Password

Confirm Password

Administrator

☐

Create User

Cancel



When users are created which will authenticate with SSL Client Certificates, the user name must match the Common Name in the certificate.

Selecting a user from the list of existing users will open an Edit User dialog box. This allows for changing the user's password and the user's Administrator status.

Configuration

The Configuration page allows Administrator level users to configure PVS for the local environment. There are three tabs available: **Web Proxy Settings**, **Feed Settings**, and **PVS Settings**.

The **Web Proxy Settings** tab configures the settings for a web proxy if one is needed for plugin updates. These settings include the proxy host IP address, port, username, password, and a user-agent field if a custom agent string is needed.

The **Feed Settings** tab allows for updating the Activation Code, plugins, performing offline updates, and configuring a custom plugin feed host.

The Activation Code and manual plugin update buttons are only used when using PVS in a stand-alone mode (not attached to a SecurityCenter). The Activation Code will only need to be updated when it expires. When PVS is used with SecurityCenter, entering "SecurityCenter" in the Activation Code box will enable the ability to upload and apply the appropriate license key to work with SecurityCenter installations.

The **Offline Update** allows for manually updating the plugins when the PVS host is not able to connect to the Internet. After downloading the plugin update archive from Tenable, select the **"Choose File"** button and select the archive to upload. Click the **"Upload Archive"** button to send the file to the PVS host, which will then update the plugins. If a new client is part of the update, the browser needs to be refreshed to view the updated client.

The **Custom Plugin Feed** host is an alternate feed host. These are typically hosted on a local network to provide custom PVS plugins.

The **PVS Settings** tab provides options for configuring the network settings for PVS. This is the main configuration page that controls what network(s) are monitored or excluded, how to monitor the network, and what network interfaces PVS has identified for monitoring.




When options are changed on this page and submitted, the PVS service on the host must be stopped and started for them to take effect.

Name	Description
ACAS Classification	
ACAS	<p>Support for ACAS banners may be enabled from the command line of the PVS server service using the command <code>./pvs --config --add "ACAS Classification" "SECRET"</code> from the binary directory on the server and is case sensitive. "SECRET" may be replaced by "UNCLASSIFIED", "CONFIDENTIAL", "TOP SECRET", or "NOFORN". Once enabled, a drop-down selection for the ACAS option will appear in the GUI front end.</p> <p>Support for ACAS banners may be disabled from the command line of the PVS server using the command <code>./pvs --config --delete "ACAS Classification"</code> from the binary directory on the server and is case sensitive.</p>
Memory	
Sessions Cache Size	The Sessions Cache Size is the size in megabytes (MB) of the session table. Adjust the session size as needed for the local network.
Packet Cache Size	The 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 before processing. By default it is set to 512 MB with a maximum size of 512 MB. When the cache is full, any subsequent packets captured will be dropped until space in the cache becomes available.
Monitoring	
Monitored Network Interfaces	The Monitored Network Interfaces field specifies the network device(s) to use for sniffing packets. Devices may be selected individually or in multiples. At least one interface must be selected from the list of available devices.

Monitored Network IP Addresses and Ranges	Specifies the network(s) to be monitored. The default setting is to monitor all IPv4 addresses with the setting of 0.0.0.0/0. This should be changed to only monitor target networks; otherwise PVS may quickly become overwhelmed. It may contain both IPv4 and IPv6 addresses. Multiple addresses are separated by commas. When monitoring VLAN networks, a syntax of “vlan ipaddress/subnet” must be used. Example: 192.168.1.0/24,2001:DB8::/64,10.2.3.0/22,vlan 172.16.0.0/16,192.168.3.123/32
Excluded Network IP Addresses and Ranges	Specifies any network(s) to specifically exclude from PVS monitoring. Specify networks using CIDR notation. It may contain both IPv4 and IPv6 addresses. Multiple addresses are separated by commas. When excluding VLAN networks a syntax of “vlan ipaddress/subnet” must be used. If left blank, no addresses will be excluded. Example: 192.168.1.0/24,2001:DB8::/64,10.2.3.0/22,vlan 172.16.0.0/16,192.168.3.123/32
PVS Proxy	
PVS Proxy Username	This configures the username that SecurityCenter 4.7.x or earlier will use to connect to the PVS proxy.
PVS Proxy Password	The PVS proxy password is the password that SecurityCenter 4.7.x or earlier will use in combination with the username to connect to the PVS proxy.
PVS Proxy IP Address	By default, the PVS proxy listens on all IPv4 addresses with the setting of 0.0.0.0. This may be configured to listen on a specific IPv4 address. The PVS proxy only listens on the configured address if configured for SecurityCenter 4.7.x or earlier connectivity.
PVS Proxy Port	This setting configures the PVS proxy listening port. By default it is set to 1243 and may be altered as appropriate for the local environment. The PVS proxy only listens on the configured port if configured for SecurityCenter 4.7.x or earlier connectivity.
PVS Restart Attempts	The PVS proxy monitors the state of the PVS engine. If the engine stops running, the proxy will attempt to restart the PVS engine the specified number of times, with the default setting being 3 and may be set to a value of 1 to 9. Once the restart attempt limit is reached, it will stop trying for a period of 30 minutes.
PVS Restart Interval	This setting configures the amount of time, in minutes, between PVS restart attempts. This setting is set to 10 minutes by default and may be set to a value of 1 to 3600.
PVS Web Server	
Enable SSL for Web Server	PVS enables SSL protection for connections to the web server by default. Disabling this option will send traffic between a web browser and PVS unencrypted and is therefore not recommended. Custom SSL certificates may be installed in the <code>/opt/pvs/var/pvs-proxy/ssl</code> directory. Changes to this setting require that the PVS services be stopped and started.
Minimum Password Length	The Minimum Password Length option determines the lowest number of characters a password may contain. This may be between 5 and 32 characters, with a default setting of 5.
PVS Web Server Address	By default the PVS web server listens on all available IPv4 and IPv6 addresses utilizing the setting 0.0.0.0. This may be changed to listen on a specific address or multiple addresses separated by commas.
PVS Web Server Port	This is the port the PVS web server listens on, which by default is 8835. This may be altered as appropriate for the local environment.

PVS Web Server Idle Session Timeout	This setting is the number of minutes after which a web session becomes idle. The default setting for this timeout is 30 minutes. The valid settings are between 5 and 60 minutes.
Enable SSL Client Certificate Authentication	When selected, the web server will only accept SSL client certificates for user authentication.
Enable Debug Logging for PVS Web Server	When selected, debug information will be included in the web server logs for troubleshooting issues related to the web server. The logs will grow rather large if this is routinely enabled.

Plugins


Process High Speed Plugins Only	PVS is designed to expect to find various protocols on non-standard ports. For example, PVS can easily find an Apache server running on a port other than 80. However, on a high traffic network, PVS can be placed into a “high-speed” mode that allows it to focus certain plugins on specific ports. When the high speed option is enabled, any plugin that utilizes the keywords <code>hs_dport</code> or <code>hs_sport</code> will be executed only on traffic traversing the specified ports.
Enable Automatic Plugin Updates	<p>When enabled, PVS will automatically update its plugins from the Tenable site on a daily basis. Disabling this option if the PVS server is not connected to the Internet is recommended.</p> <div>  <p>When the HTML Client is updated the web browser needs to be refreshed to utilize the new client. In some cases the browser's cache must be deleted to view the new client.</p> </div>

Realtime Events

Realtime Events File Size	The Realtime Events File Size option specifies the maximum amount of data from real-time 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.
Log Realtime Events	This option records PVS detected real-time events to a local log file.

Reports

Report Threshold	This setting is used to determine the number of times the encryption detection algorithm is executed during a session. Once the threshold is reached, the algorithm is no longer executed during the session. This variable has a default of “3”.
Report Lifetime	Reports can be cached for a specified number of days. After the configured day count is met, PVS's entire model of a discovered network is completely removed. PVS starts over again learning about the hosts that are involved on the network. This value can be set to a maximum value of 90 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 7 days.
Report Frequency	This variable specifies in minutes (default 15) how often the PVS will write a report. SecurityCenter 4.6 and higher will retrieve the PVS report every 15 minutes.
Knowledgebase Lifetime	The maximum length of time in seconds that a knowledgebase entry remains valid after its addition.

New Asset Discovery Interval	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, PVS can be configured to run for several days to gain knowledge about which hosts are active. This prevents PVS from issuing an alert for hosts that already exist. The number of days PVS should monitor traffic to learn which hosts are active is specified by this setting. For large networks, Tenable recommends that PVS operate for at least two days (the default setting) before detecting new hosts.
Connections to Services	When enabled, this option enables 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 PVS of this type are logged as PVS 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.
Session Analysis	
Encrypted Sessions Dependency Plugins	This list of Plugin IDs, separated by commas, is used to detect encrypted traffic.
Encrypted Sessions Excluded Network Ranges	This setting defines the list of IPv4 and IPv6 addresses and ports in CIDR notation to be excluded from monitoring for encrypted traffic. Example: 192.168.1.0/24,2001:DB8::/64,10.2.3.0/22,vlan 172.16.0.0/16,192.168.3.123/32
Interactive Sessions Dependency Plugins	This list of Plugin IDs, separated by commas, is used to detect interactive sessions.
Interactive Sessions Excluded Network Ranges	This setting defines the list of IPv4 and IPv6 addresses and ports in CIDR notation to be excluded from monitoring for interactive sessions. Example: 192.168.1.0/24,2001:DB8::/64,10.2.3.0/22,vlan 172.16.0.0/16,192.168.3.123/32
Syslog	
Realtime Syslog Server List	Specifies the IPv4 or IPv6 address and UDP port of a Syslog server to receive real-time events from the PVS. The field accepts up to 255 characters for the Syslog IP addresses. A local Syslog daemon is not required. Multiple addresses are separated by commas. Example: 192.168.1.12:4567,10.10.10.10:514,[2001:DB8::23B4]:514
Vulnerability Syslog Server List	<p>Specifies the IPv4 or IPv6 address and UDP port of a Syslog server to receive vulnerability data from PVS. The field accepts up to 255 characters for the Syslog IP addresses. A local Syslog daemon is not required. Multiple addresses are separated by commas. Example: 192.168.1.12:4567,10.10.10.10:514,[2001:DB8::23B4]:514</p> <div>  <p>While PVS may display multiple log events related to one connection, it will only send a single event to the remote Syslog server(s).</p> </div>

Command Line Operation

The PVS engine provides many options to update and configure PVS from the command line in both Windows and Linux versions. The HTML5 interface is considered the primary method to make changes. When using the command line interface in Linux, it is assumed the commands are being performed by a root user or equivalent. When the command line is used in Microsoft Windows, it is assumed that the shell has been launched using the “Run as Administrator” or an equivalent option.

This section is separated into three sections: a section each for Windows, Linux, and Mac OS X specific command line options and a section for common options.

Command Line Operations for Linux

Starting the Passive Vulnerability Scanner for Linux

The **service** command is the preferred method to launch the **pvs** and **pvs-proxy** binaries as follows:

```
# service pvs start
Starting PVS                                [ OK ]
Starting PVS Proxy                          [ OK ]
# ps aux|grep pvs
root      18626  9.5  70.5 1492840 1357176 pts/3  Sl   15:38   0:03 /opt/pvs//bin/pvs
root      18629  0.2   0.2   21160   4892 pts/3   S    15:38   0:00 /opt/pvs//bin/pvs-
        proxy
```

Once running, the **pvs** binary will be monitored by the **pvs-proxy** daemon utilizing its watchdog options.

In this mode, the PVS will store its vulnerability reports in a directory monitored by the PVS Proxy. When the SecurityCenter connects to the PVS Proxy, it will copy the appropriate report found in this directory to the SecurityCenter and import it into the relevant databases.

Once a day, as scheduled, if the SecurityCenter has received new passive vulnerability plugins from Tenable, it will install them in the PVS plugin directory. PVS will detect the change and automatically reload and begin using the new plugins.

Real-time PVS data will be communicated to the configured Log Correlation Engine server or syslog server(s) in real-time.

Stopping the Passive Vulnerability Scanner for Linux

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 to restart the service, use the command:

```
# service pvs restart
Stopping PVS Proxy                          [ OK ]
Stopping PVS                               [ OK ]
Starting PVS                               [ OK ]
Starting PVS Proxy                          [ OK ]
```

File Locations

PVS installs its files into the following locations:

Path	Purpose
/opt/pvs	Base directory

/opt/pvs/etc (deprecated)		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		Contains the folders for PVS and the PVS-Proxy
/opt/pvs/var/pvs		Plugins, discovered vulnerabilities, log files, keys, software license agreement, and other miscellaneous items among its directories and sub-directories
	db	This directory contains the database files relating to the configuration, reports, and users for PVS.
	kb	This directory stores the PVS knowledgebase, if used.
	logs	Contains PVS logs
	plugins	<p>Contains the <code>tenable_plugins.prmx</code> pushed down by SecurityCenter. May also contain custom plugins.</p> <div>  <p>Do not change from the default of <code>C:\ProgramData\Tenable\PVS\pvs</code> if SecurityCenter is being used to manage the plugins.</p> </div>
	pvs-services	A file that PVS uses to map service names to ports. This file may be edited by the user. Plugin updates will not overwrite modifications to the file.
	reports	Contains reports generated by PVS with the exception of <code>.nsr</code> . This folder contains the <code>.nessus</code> file generated by default.
	scripts	Folder for custom scripts, if any
	users	Contains folders for user files and reports
	www	Contains the files for the PVS web front-end
/opt/pvs/var/pvs-proxy		Parent folder for files used/created by the PVS proxy
	logs	Contains PVS proxy and PVS proxy service logs
	scans	By default, PVS creates the <code>.nsr</code> file in the scans folder. The proxy is then responsible for handing the report to SecurityCenter when SecurityCenter attempts to pull it.
	ssl	Contains SSL certificates used by the proxy and web server for the SSL connection between itself and SecurityCenter or the web browser

Command Line Operations for Windows


This section describes some operations that are performed on the PVS server from a command line in Windows. Command line operations need to be executed from a Windows shell that has been launched using the “Run as Administrator” command or similar, depending on the Windows version.

File Locations

PVS installs its files into the following locations:

Path	Purpose
C:\Program Files\Tenable\PVS	Contains PVS binaries and dependent libraries
C:\ProgramData\Tenable\PVS	Contains all data files consumed and output by PVS and PVS Proxy (i.e., configuration, plugins, logs, reports)

The following is the folder layout under C:\ProgramData\Tenable\PVS\:

Folder		Purpose
Conf (deprecated)		Contains PVS and PVS Proxy configuration files
pvs		Parent folder for PVS logs, reports, plugins, and scripts directories. Also contains the pvs-services file.
	db	This directory contains the database files relating to the configuration, reports, and users for PVS.
	kb	This directory stores the PVS knowledgebase, if used.
	logs	Contains PVS logs
	plugins	Contains the tenable_plugins.prmx pushed down by SecurityCenter. May also contain custom plugins. <div> Do not change from the default of C:\ProgramData\Tenable\PVS\pvs if SecurityCenter is being used to manage the plugins.</div>
	pvs-services	A file that PVS uses to map service names to ports. This file may be edited by the user. Plugin updates will not overwrite modifications to the file.
	reports	Contains reports generated by PVS with the exception of .nsr . This folder contains the .nessus file generated by default.
	scripts	Folder for custom scripts, if any
	users	Contains folders for user files and reports
	www	Contains the files for the PVS web front-end

pvs-proxy		Parent folder for files used/created by the PVS proxy
	logs	Contains PVS proxy and PVS proxy service logs
	scans	By default, PVS creates the <code>.nsr</code> file in the scans folder. The proxy is then responsible for handing the report to SecurityCenter when SecurityCenter attempts to pull it.
	ssl	Contains SSL certificates used by the proxy and web server for the SSL connection between itself and SecurityCenter or the web browser
run		Contains process ID temporary files

Starting and Stopping PVS

PVS is controlled as a service under Windows. To start or stop PVS, launch the Services control panel utility. Under the list of services find “Tenable PVS Proxy Service”. Right clicking on the service will provide a list of options for the service, including the ability to start or stop the Tenable PVS or Tenable PVS Proxy service.

Command Line Operations for Mac OS X

Stopping the Passive Vulnerability Scanner for Mac OS X

After the installation, the `PVS` service will start. During each reboot, the service will automatically start. If there is a reason to start or stop the service, it can be done via a Terminal window (command line). If performed via the command line, it must be run as “root”, or via `sudo`:


Action	Command to Manage PVS
Start	<code># launchctl load -w /Library/LaunchDaemons/com.tenablesecurity.pvs-proxy.plist</code>
Stop	<code># launchctl unload -w /Library/LaunchDaemons/com.tenablesecurity.pvs-proxy.plist</code>

Alternately, the PVS service can be managed via System Preferences.

File Locations

PVS installs its files into the following locations:

Path	Purpose
<code>/Library/PVS</code>	Base directory
<code>/Library/PVS/etc (deprecated)</code>	Configuration files for PVS and the PVS Proxy
<code>/Library/PVS/docs</code>	This directory contains the PVS license agreement in various file formats.
<code>/Library/PVS/bin</code>	Location of the PVS and PVS Proxy executables, plus several helper tools for the PVS Proxy daemon
<code>/Library/PVS/var/pvs</code>	Plugins, discovered vulnerabilities, log files, keys, software license agreement, and other miscellaneous items among its directories and sub-

		directories
	db	This directory contains the database files relating to the configuration, reports, and users for PVS.
	kb	This directory stores the PVS knowledgebase, if used.
	logs	Contains PVS logs
	plugins	Contains the <code>tenable_plugins.prmx</code> pushed down by SecurityCenter. May also contain custom plugins. <div>  <p>Do not change from the default of <code>C:\ProgramData\Tenable\PVS\pvs</code> if SecurityCenter is being used to manage the plugins.</p> </div>
	pvs-services	A file that PVS uses to map service names to ports. This file may be edited by the user. Plugin updates will not overwrite modifications to the file.
	reports	Contains reports generated by PVS with the exception of <code>.nsr</code> . This folder contains the <code>.nessus</code> file generated by default.
	scripts	Folder for custom scripts, if any
	users	Contains files and reports for the PVS users.
	www	Contains the files for the PVS web front-end
/Library/PVS/var/pvs-proxy		Parent folder for files used/created by the PVS proxy
	logs	Contains PVS proxy and PVS proxy service logs
	scans	By default, PVS creates the <code>.nsr</code> file in the scans folder. The proxy is then responsible for handing the report to SecurityCenter when SecurityCenter attempts to pull it.
	ssl	Contains SSL certificates used by the proxy and web server for the SSL connection between itself and SecurityCenter or the web browser

Common Command Line Options

PVS may be run from the command line to analyze `pcap` files and generate a report file for use with SecurityCenter or other programs, update plugins, and perform configuration tasks. The following is a list of options available and their purpose. Running the PVS binary with the `-h` option will display a list of available options.



When using the PVS binary at the command line to perform tasks or change options, the PVS services need to be stopped. Failure to do so may result in undesired results in the performance of the command line task or the PVS monitoring.

The PVS binary for Windows is located at:

C:\Program Files\Tenable\PVS>**pvs.exe**

The PVS binary for Mac OS X is located at:


/Library/PVS/bin

The PVS binary for Linux is located at:

/opt/pvs/bin/pvs



Running the **pvs** command on Linux without specifying the full path will result in launching the Linux **pvs** (physical disk volume) tool rather than the Tenable Passive Vulnerability Scanner.

Option	Purpose
-m	Shows various aspects of memory usage during the processing of the pvs command.
-p packet_dump_file	Replace packet_dump_file with the local file name or path to file name to write out the captured packets to a file.
-r report	Replace the word report with the local file or path to the file that you wish to write the report to if it is other than the default report location.
-f packet_dump_file	Replace packet_dump_file with the path to the pcap file you wish to have pvs process.
-h	Displays the command line options help file.
-a <activation code>	Enter the Activation Code to activate the PVS to enable plugin updates and monitoring functions.
-k key_file	Replace key_file with the local file or path to the file that contains the license key to display pertinent information about the designated license key. When used without an option will display the PVS activation status.
-l	Displays a list of the plugin IDs that are loaded by the PVS.
-v	Shows the version information about the installed PVS.
-d debug mode	<div>This command allows PVS to run in debug mode for troubleshooting purposes</div> <div> It is important to note that with PVS operating with this option enabled will result in more resources used by the system and should only be used when directed by a Tenable Support Technician</div>
--config --list	Lists the current PVS and PVS Proxy configuration parameters. Parameter values are listed to the left of the colon character and are case sensitive. The value of the parameter is displayed to the right of the colon character.

<code>--config --add "custom_paramater name" "parameter value"</code>	Add a custom configuration parameter for PVS or PVS Proxy. The double quote characters are required, and single quotes may be used when special characters are required.
<code>--config "parameter name" ["parameter value"]</code>	Displays the defined parameter value. If a value is added at the end of the command, the parameter is updated with the new setting. The double quote characters are required, and single quotes may be used when special characters are required.
<code>--config --delete "custom_parameter name"</code>	The delete command may be used to remove custom configuration parameters.
<code>--register-offline <.rc file></code>	When using PVS in an offline mode, the <code>.rc</code> file obtained from Tenable is used in this command.
<code>--challenge</code>	When using PVS in an offline mode, a challenge code is required by Tenable to generate the proper <code>.rc</code> file. This command provides the required challenge code.
<code>--update-plugins <plugins tarball></code>	When using the PVS in an offline mode, updating the plugins requires downloading a tarball from Tenable. When updating the plugins from the command line, this command is used to identify the file to use for updating the plugins.

Define Unknown or Customized Ports

Many networks will contain traffic on ports PVS has defined as a different traffic type or on alternate ports. If the port is not defined at all it will be displayed as “Unknown”. The `pvs-services` file may be edited to customize or add the port information to provide accurate reporting for the ports on the network.

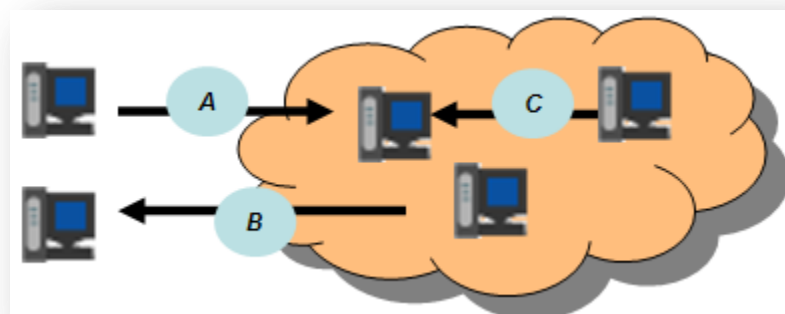
For example, there are two lines in the `pvs-services` file by default that define SMTP traffic. They read “smtp 25/tcp” and “smtp 25/udp”. If the organization routinely sends SMTP data over port 2525 those lines can be changed to or have lines added to the file that reads “smtp 2525/tcp” and “smtp 2525/udp”.

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.



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 option “connections to services” enables PVS to log network client to server activity.

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 “**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.

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. This is the “**show-connections**” keyword on the configuration page. 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 IPv4 address is 1.1.1.1 or your IPv6 address is 2001:DB8::AE59:3FC2 and you use the SSH service to connect to “some_company.com”, use of these options would record the following:

```
show-connections:  
→ some_company.com:SSH  
2001:DB8::AE59:3FC2 -> some_company.com
```

```
connections-to-services
```

→ SSH
2001:DB8::AE59:3FC2 -> SSH

Using the “**connections-to-services**” option lets you know that the system at 1.1.1.1 and 2001:DB8::AE59:3FC2 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 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/log-correlation-engine>.

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 SecurityCenter's Vulnerability Analysis 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 an encrypted library of passive vulnerability detection scripts. This file cannot be modified by the end users of the PVS. However, if certain scripts need to be disabled, they can be specified by the PASL ID and '.pasl' appended, such as "1234.pasl", to disable the PASL with the ID of 1234 on a single line in the disabled-scripts.txt file.

If a plugin needs to be disabled, enter its ID on a single line in the disabled-plugins.txt file. If a plugin needs to be made "realtime", enter its ID on a single line in the realtime-plugins.txt file.

If any of the referenced files do not exist, simply create them using the appropriate method for the operating system. The file locations are in the following table for each operating system.

Operating System	File Path
Linux	/opt/pvs/var/pvs

Windows	C:\ProgramData\Tenable\PVS\pvs\
Mac OS X	/Library/PVS/var/pvs



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 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 balancers, 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 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.

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 some of the more commonly encountered 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	Internal Client Trusted 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 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.
00015	Internal Server Trusted Connections	The PVS has logged a unique network session of source IP, destination IP, and destination port.

Working with Passive Vulnerability Scanner Plugins

Vulnerability and Passive Fingerprint Overview

The PVS has two sources of “plugin” information: the `.prm` and `.prp` plugin libraries in the `plugins` directory.

Tenable distributes its passive vulnerability plugin database in an encrypted format. This file is known as `tenable_plugins.prm` 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 `.prp`.

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.

Downloading New Vulnerability Plugins

When PVS is registered as a stand-alone scanner using an Activation Code, plugins are updated automatically every 24 hours after the service is started. To manually update the PVS plugins from the web interface, navigate to the Feed Settings tab on the Configuration page. Next, click the “**Update Plugins**” button next to the Activation Code box. The plugins may be updated from the command line using the command “`pvs --update-plugins`”. If SecurityCenter is being used to manage a PVS, new plugins for the PVS will automatically be sent at scheduled intervals and the PVS Proxy will restart the PVS as needed.

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 `.prp` extension for the PVS to see it. The next section details how to write PVS plugins.

Restarting the Passive Vulnerability Scanner


Once new passive plugins are available to the PVS, it must be stopped and started to recognize the newly available plugins.

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
<code>bid</code>	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.
<code>bmatch</code>	This is the same as “ <code>match</code> ” but can look for any type of data. A <code>bmatch</code> must always

	have an even number of alphanumeric characters.
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 “ \n ” can be used to invoke a new line.
Exploitability: canvas core cvsstemporal metasploit	<p>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:</p> <pre>CANVAS : D2ExploitPack CORE : true CVSSTEMPORAL : CVSS2#E:F/RL:OF/RC:C</pre> <div>  <p>These keywords are displayed only in vulnerabilities detected by PVS 3.4 and greater.</p> </div>
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.
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 ”.
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.
plugin_output	This keyword displays dynamic data for a given vulnerability or event. The dynamic data is usually represented using %L or %P, and its value is obtained from the regular expressions defined using regex, regexi, pregex, or pregexi.
pmatch	This keyword is the same as “ match ” but is applied against the previous packet on the other side of the reconstructed network session.
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.
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.
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.

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.

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
hs_sport=143
name=IMAP Banner
description=An IMAP server is running on this port. Its banner is :\n %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.

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:
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 : \n%P\nWe saw : \n%L
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.

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=123456789ABCDEF5
```

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

```


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.

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 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
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

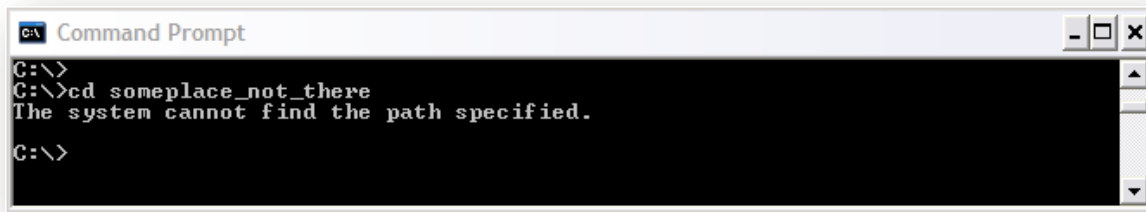
One of the PVS's strongest intrusion detection features is its ability to recognize specific services, and then to look for traffic occurring on those services that should never occur unless they have been compromised. Since the PVS can keep track of both sides of a conversation and make decisions based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

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



```
C:\>
C:\>cd someplace_not_there
The system cannot find the path specified.
C:\>
```

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”.



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. 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 Facebook accounts 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.

The following example shows how to create a custom plugin to detect users logging into their Facebook 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 Facebook account. You should ensure that such behavior is in alignment with corporate policies and guidelines. For your information, the user account was logged as:\n %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 - Facebook 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: *.facebook.com
```

The statement above ensures that they are posting a host with a domain of "*.facebook.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=Facebook_Usage
description=The remote client was observed logging into a Facebook account.
  You should ensure that such behavior is in alignment with
Corporate Policies and guidelines.  For your information, the user account
was logged as:
risk=MEDIUM
solution=Stay off of Facebook.

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

This plugin could be named **Facebook.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:

```
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\employmentfiles 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
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

Tenable uses a hybrid approach to operating system fingerprinting. Primarily, plugins are used to detect and identify the OS of a host. If this is not possible, PVS will use detected packets to identify the OS.

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>

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>

A document describing Tenable Product Plugin Families is available on the Tenable website:

http://static.tenable.com/documentation/Tenable_Products_Plugin_Families.pdf

Tenable Network Security, Inc. may be contacted via email for PVS support at sales@tenable.com or support@tenable.com.

Appendix 1: Working with SecurityCenter

Architecture

One mode PVS operates under is under the control of a SecurityCenter that provides it with passive vulnerability data and retrieves scanned 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 active vulnerability scanners. By combining active and passive vulnerability scanning, SecurityCenter can be used to efficiently and accurately manage security across large networks.

Managing Vulnerabilities

A screen capture of SecurityCenter 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.



The screenshot shows the 'Vulnerability Summary' page in SecurityCenter. It features a sidebar with navigation options like 'Cumulative', 'Mitigated', 'History', and 'Vulnerability Summary'. The main area displays a table of vulnerabilities with columns for Plugin ID, Name, Family, Severity, and Total. The table lists 13 vulnerabilities, including Firefox Version Detection, PHP 5.3 < 5.3.4 Multiple Vulnerabilities, Apache < 2.2.15 Multiple Vulnerabilities, BIND 9 DNSSEC Bogus NXDOMAIN Response Remote Cache Po..., BIND 9 DNSSEC Query Response Remote Cache Poisoning, MySQL Database Server Detection, '.dll' File Detection, HTTP Server Basic Authentication Detection, .pdf Document File Detection, PHP 5.3 < 5.3.6 String To Double Conversion DoS, and Telnet Server Detection (High Port).

Plugin ID	Name	Family	Severity	Total
3706	Firefox Version Detection	Web Clients [PVS]	Info	4
5732	PHP 5.3 < 5.3.4 Multiple Vulnerabilities	Web Servers [PVS]	High	4
5356	Apache < 2.2.15 Multiple Vulnerabilities	Web Servers [PVS]	High	4
5323	BIND 9 DNSSEC Bogus NXDOMAIN Response Remote Cache Po...	DNS Servers [PVS]	Medium	4
5243	BIND 9 DNSSEC Query Response Remote Cache Poisoning	DNS Servers [PVS]	Medium	4
5135	MySQL Database Server Detection	Database [PVS]	Info	4
4711	'.dll' File Detection	Data Leakage [PVS]	Info	4
4225	HTTP Server Basic Authentication Detection	Web Servers [PVS]	Medium	4
3963	.pdf Document File Detection	Data Leakage [PVS]	Info	4
5824	PHP 5.3 < 5.3.6 String To Double Conversion DoS	Web Servers [PVS]	High	4
3189	Telnet Server Detection (High Port)	Generic [PVS]	Info	4

Updating the PVS Management Interface

On occasion, the PVS management interface needs to be updated to provide new or updated features. When managed by SecurityCenter 4.7.1 or earlier, the PVS web server and interface are not updated automatically by the plugins provided through SecurityCenter. Therefore, when the web components are to be updated, it must be performed manually on each PVS.

To manually update the plugins, first download the latest plugins from <https://downloads.nessus.org/sc-passive.tar.gz>. Next, log in to the PVS interface as an admin user and navigate to the **Configuration** page, and then the **Feed Settings** tab. The **Offline Update** section contains the “**Browse**” button, which opens a dialog box to allow you to select the archive file to upload. Click the “**Upload Archive**” button to send the file to the PVS host, which will then update the plugins. After stopping and starting the PVS service on the host, the new interface will be available for use.

The Passive Vulnerability Scanner is Real-Time

Since the PVS's vulnerability data is constantly being fed into SecurityCenter 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.

Appendix 2: Syslog Message Formats

PVS provides options to send real-time and vulnerability data as syslog messages. There are four formats of syslog files sent from PVS as described here.

1. Syslog message format for syslog generated by real-time PRMs:

```
<priority>timestamp pvs:
  src_ip:src_port|dst_ip:dst_port|protocol|plugin_id|plugin_name|matched_text_current_packet|matched_text_previous_packet|risk
```

2. Syslog message format for syslog generated by real-time PASL or vuln PRM or PASL:

```
<priority>timestamp pvs:
  src_ip:src_port|dst_ip:dst_port|protocol|plugin_id|plugin_name|plugin_description|risk
```

3. Syslog message format for Open Port alert, Service Connection alert, Client and Server Connection alerts, Tracked Sessions alert, New Host alert, and Accepts External Connection alert:

```
<priority>timestamp pvs:
  src_ip:src_port|dst_ip:dst_port|protocol|plugin_id|plugin_name|plugin_specific_data|risk
```

4. Encrypted/Interactive session alert:

```
<priority>timestamp pvs:
  src_ip:src_port|dst_ip:dst_port|protocol|plugin_id|plugin_name|risk
```

The following table describes each field.

Name	Description
priority	The syslog facility level of the message.
timestamp	This field provides the date and time of the syslog message.
src_ip	This field is the source IP address reported for the traffic.
src_port	This field is the source port for the reported traffic.
dst_ip	This field is the destination IP address for the reported traffic.
dst_port	This field is the destination port for the reported traffic.
protocol	This reports the protocol used for the reported traffic.

plugin_id	<p>The reported PVS plugin id triggered by the reported traffic.</p> <p>Some examples:</p> <p>0 for open port alert 2 for service connection alert 3 for client connection alert 4 for internal interactive session 5 for outbound interactive session 6 for inbound interactive session 7 for internal encrypted session 8 for outbound encrypted session 9 for inbound encrypted session 10 for tracked sessions 13 for new host alert 14 for accepts external connection alert 15 for server connection alert</p>
plugin_name	<p>The name of the PVS plugin triggered by the reported traffic</p> <p>Some examples:</p> <p>'new-open-port' for open port alert 'connection-to-service' for service connection alert 'connection' for client connection alert 'tracked-session' for tracked session alert 'new-host-alert' for new host alert 'accepts-external-connections' for accepts external connection alert 'server-connection' for server connection alert</p>
risk	<p>The associated risk level of the reported vulnerability. This can be "NONE", "LOW", "MEDIUM", "HIGH", or "INFO".</p>
matched_text_current_packet	<p>Reports the payload which causes a match in the packet to trigger the PVS event.</p>
matched_text_current_packet	<p>Reports the payload from the other side of the session, which causes a match in the packet to trigger the PVS event.</p>
plugin_specific_data	<p>The data provided is determined by the type of data reported.</p> <p>Some examples:</p> <p>'new host alert' is the value is the MAC address of the host 'tracked session alert' is the value of the payload of packet</p> <p>This field is not applicable for 'service connection' alert, 'client connection' alerts, 'server connection' alerts, 'open port alert', and 'accepts external connection' alert.</p>

Appendix 3: PVS Activation without Internet Access

If your PVS installation cannot reach the Internet directly, use the following procedure to register and update plugins:

On the system running PVS, type the following command:

Platform	Command to Run
Red Hat Linux / CentOS	<code># /opt/pvs/bin/pvs --challenge</code>
Mac OS X	<code># /Library/PVS/bin/pvs --challenge</code>
Windows	<code>C:\Program Files\Tenable\PVS>pvs --challenge</code>

This will produce a string called “challenge” that appears similar to following:

```
569ccd9ac72ab3a62a3115a945ef8e710c0d73b8
```

Next, go to <https://plugins.nessus.org/offline-pvs.php> and paste the “challenge” string as well as the Activation Code that you received previously into the appropriate text boxes. This will produce a URL that will give you direct access to the PVS plugins. **Save this URL, which will be used every time you update your plugins.** In addition, a file called `nessus-fetch.rc` will be produced. Copy this file to the host running PVS in the appropriate directory:

Platform	Directory
Red Hat Linux / CentOS	<code># /opt/pvs/etc/pvs/</code>
Mac OS X	<code># /Library/PVS/etc/</code>
Windows	<code>C:\Program Files\Tenable\PVS\</code>

Once the `nessus-fetch.rc` file has been copied, run the `pvs --register-offline` command to install the file:

Platform	Directory
Red Hat Linux / CentOS	<code># /opt/pvs/bin/pvs --register-offline /opt/pvs/etc/pvs/nessus-fetch.rc</code>
Mac OS X	<code># /Library/PVS/bin/pvs --register-offline /Library/PVS/etc/nessus-fetch.rc</code>
Windows	<code>C:\Program Files\Tenable\PVS>pvs --register-offline "C:\Program Files\Tenable\PVS\nessus-fetch.rc"</code>

The newest plugins can be obtained by going to the URL that was provided in the previous step. Here, you will receive a TAR file (e.g., `sc-passive.tar.gz`). Copy the file to the PVS system and then type the appropriate command for your platform:

Platform	Command
Red Hat Linux / CentOS	# /opt/pvs/sbin/pvs --update-plugins /path/to/sc-passive.tar.gz
Mac OS X	# /Library/PVS/bin/pvs --update-plugins /path/to/sc-passive.tar.gz
Windows	C:\Program Files\Tenable\PVS>pvs --update-plugins C:\path\to\sc-passive.tar.gz

About Tenable Network Security

Tenable Network Security provides continuous network monitoring to identify vulnerabilities, reduce risk, and ensure compliance. Our family of products includes SecurityCenter Continuous View™, which provides the most comprehensive and integrated view of network health, and Nessus®, the global standard in detecting and assessing network data. Tenable is relied upon by more than 24,000 organizations, including the entire U.S. Department of Defense and many of the world's largest companies and governments. For more information, please visit www.tenable.com.

GLOBAL HEADQUARTERS

Tenable Network Security
7021 Columbia Gateway Drive
Suite 500
Columbia, MD 21046
410.872.0555
www.tenable.com

