



Technical Overview

The Pansec Vulnerability Scanner (PVS) is a network vulnerability scanning system available either as a managed service or as a licensed software package.

Features

- Low-impact scanning – tests use limited bandwidth and are designed to be non-aggressive.
- Tailored test profiles – tests can be modified to suit user requirements.
- Scalable architecture – install on a single machine or on an array of servers.
- “Set and forget” scheduling – test scheduling can be fixed or randomised.
- Comprehensive vulnerability database – updated at least every two weeks.
- Offline Exposure and Change Analysis – results are analysed after scanning has finished.
- Single-click baselining – baseline a system so you can monitor changes.
- Range of report formats – technical, change, management, and summary reports.
- Email notification – a single daily email tells you the status of all your servers.
- Minimal user intervention – only required to add new tests and delete old ones.

Vulnerability assessment, as performed by PVS, is significantly less invasive than Penetration Testing. PVS identifies vulnerabilities without compromising or adversely affecting the operation of the target system in any way. You can then maximise the use of your resources by directing your security professionals to the systems and services most in need of remediation.

Architecture

PVS is a network vulnerability scanning system capable of performing vulnerability assessments on up to 120,000 IP addresses every day. Critical design characteristics were low resource usage, minimal system impact, no operational intervention, and clear, concise reports. The system is implemented as a set of software services allowing an installation to scale from a single server to a global infrastructure of many geographically distributed servers.

Because PVS is heavily modularised, different parts of the system can be updated without having to perform a complete reinstallation. In addition, new test components can be deployed without disturbing the rest of the system. Each major module is implemented as a system service allowing it to integrate closely with the operating system and facilitating hands-off management. Most of the communication between the different services takes place via a powerful database as this provides a reliable method of inter-process communication that can be monitored in real-time.

Starting with no knowledge of a system other than its IP address, PVS attempts to find out what a hacker could discover about the system and to determine ways in which the system could be open to attack. PVS has been carefully designed to use limited bandwidth during this process so as not to affect the target system’s connectivity. It has also been designed to examine the system without adversely affecting the system i.e. it does not attempt to break into the system, to write files to the system, or to exploit any of the vulnerabilities that may be found.

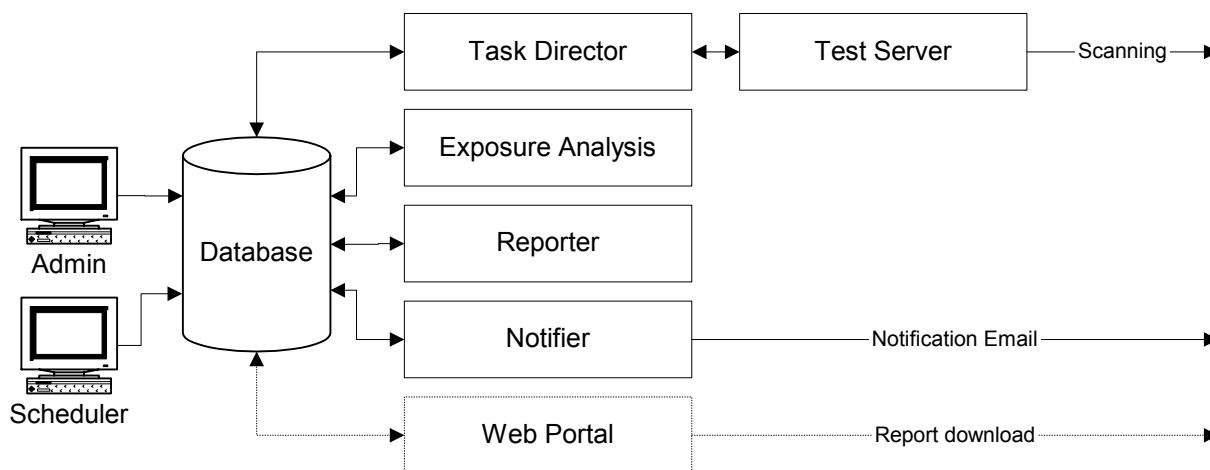


PVS scans each IP address it has been given and attempts to find out what ports are open, what operating system is running, and what network services are available. It then examines the results for possible vulnerabilities and generates reports describing the state of the system. Because PVS uses a database to store its results, it can compare new results against previous results or against a baseline. The generated reports can list changes to the target system giving the user a powerful tool to check the state of the system against an “ideal”, baseline state.

The PVS notifier service can be configured to generate a single notification email when all the reports for a given customer (or group of IP addresses) are ready. This email can be configured to alert the customer (or user) when there are significant changes or significant new exposures that require urgent attention. This email can reduce the overhead of monitoring an entire network to a few minutes each day.

Because PVS makes no assumptions about the targets it is scanning, it can be deployed to test either internet facing servers or an intranet with equal ease. More complicated deployments to scan both internal and external networks are also possible.

Workflow



The daily cycle of PVS activity is as follows:

1. The Scheduler runs daily to schedule the next day's tests in the database.
2. The Task Director picks up scheduled tests when they are ready to run, constructs test profiles which determine which individual tests are to be run and what actions to take on the results of those tests, and dispatches the profiles to a Test Server.
3. Each Test Server receives its allocated test profiles from the Task Director and runs them against a target IP address. Individual steps in the profiles tell the Test Server which Test Component to run, what parameters to send to the test component, and what results to expect back from each test component. When the test profile has finished executing, the results are sent back to the Task Director.
4. The Task Director receives the results from the Test Server and writes them to the database.
5. The Exposure Analysis Service reads the results of each test and examines them for exposures by looking for specific exposure signatures in the data. It also looks for changes



in the data from the last time the test was run or against a baseline. The changes and exposure details are then written back to the database.

6. The Reporter Service combines the results of the tests, the changes found, and the exposures found into reports ready for viewing directly or for download via the Web Portal.

In addition, it is possible to run ad-hoc or immediate tests directly from the administration console. These tests are stored in the database and are executed, analysed, and reported on as soon as possible.

System Components

Database

The core of PVS is the database which stores configuration data, target data, and results.

Configuration data consists of information about test components, test profiles, and products. Test profiles define the scans that are run against target systems in terms of the individual test components that are executed, the parameters passed to the components, and the actions to take based on the responses received. Test profiles are packaged into products which specify which profiles to run and how often.

A typical test profile checks a target IP address for visibility using a number of techniques – ICMP, SYN, ACK, reverse DNS, etc. It then performs a TCP scan and a UDP scan looking for open ports. Any ports found to be open are examined to determine what type of service and associated protocol might be using it. Each service is then probed for protocol specific data. The operating system is also fingerprinted using a range of techniques.

Another crucial part of the configuration data is the exposure signature table that is reloaded at regular intervals from the main Pansec Knowledge Base which is constantly updated by our security researchers. Users of the managed service will automatically receive the latest exposure information in their reports. Users of the licensed software will receive database updates every two weeks or more often if critical vulnerabilities are discovered.

Target data consists of information about nodes (individual IP addresses to be scanned) and networks, notification information for groups of nodes and networks, blackout information (times when nodes or networks should not be scanned), baseline information, and report configurations.

Result data consists of the data returned from the individual test components and the results of the exposure and change analysis.

Scheduler

The Scheduler generates a list of tests to be run on a specific day, divides the day into time slots according to the approximate running time of each test profile to be run, and schedules tests into these slots so that the tests for a day are spread throughout the day. The Scheduler also takes into account blackout times during which tests should not be executed.

Task Director



The Task Director reads scheduled tests from the database and constructs an XML document describing the components to be executed. This is then dispatched to a Test Server via a named pipe. When the Test Server has finished executing the test profile, it formats the results as an XML document and sends them back to the Task Director which parses the XML and writes the results to the database.

Each Task Director serves a given location. A Task Director will only access tests assigned to that location and will only send tests to the Test Servers assigned to the same location. PVS can run multiple Task Directors each serving a different location thus allowing it to assign tests to the Test Server(s) best positioned to execute the test. The Task Director is also responsible for ensuring that tests are evenly distributed between Test Servers assigned to the same location.

Multiple Task Directors can be run simultaneously for the same location. At any time, one Task Director will be active while the others monitor the active or primary Task Director and become active only if the current primary Task Director shuts down. This is the basis for the fault-tolerant, scalable architecture of PVS.

Test Server

Each Test Server receives test profiles from the Task Director, breaks them down into a list of Test Components to execute, and runs the individual Test Components. Depending on the results returned by each Test Component, a Test Server can also run further Test Components according to instructions contained in the test profile. For instance, if the TCPScan component indicates that TCP port 80 (HTTP) is responsive, the Test Server can then run the HTTPScan component to try to determine which web server is running on the target system. Once all the Test Components have run, the Test Server formats the results as an XML document and sends them back to the Task Director.

Test Components

Each Test Component exposes a standard interface to the Test Server. This interface allows the Test Server to send parameters to the component, to start or stop the component, and to receive result, finish or error events from the component. This architecture allows PVS to be easily extended with new or modified components. The following components are currently deployed:

Component	Description
DNSScan	Determines whether a DNS service is running, what version of the DNS service is running, and whether it will resolve DNS queries.
FingerScan	Determines whether a Finger service is running and attempts to enumerate users.
FTPScan	Determines whether an FTP server is running, attempts to perform an anonymous login, and to execute the PORT command.
FW1Detect	Determines whether Checkpoint Firewall-1 SecureRemote is running on the target system.
HTTPScan	Determine whether a web server is running on the target system. Attempts to access a list of CGI/script files, returns a service



Component	Description
	banner if possible, attempts to execute the OPTION command.
MySQLDetect	Determines which version of MySQL is running.
NameScan	Attempts to enumerate NetBIOS names on the target system.
NetEnum	Enumerates the IP addresses in a given range.
NetScan	Checks whether a system is visible using ICMP, TCP SYN, ACK and FIN, and DNS lookup.
NNTPScan	Determines whether a News server is running on the target system. Probes for specific newsgroups and attempts to return a service banner.
OsDetect2	Examines a target's TCP/IP stack to determine what operating system the system is running.
PortmapScan	Attempts to enumerate RPC services running on the target machine.
POP3Scan	Determines whether a POP3 service is running.
RegistryScan	Attempts to connect to the target system's registry and enumerate the values of certain keys.
RexecScan	Determines whether a remote exec service is running.
RloginScan	Determines whether the target system is running a remote login (rlogin) service and attempts to login and return a service banner.
RshellScan	Determines whether the target system is running a remote shell (rsh) service, attempts to login and return a service banner.
ServiceScan	Attempts to enumerate NetBIOS services.
ShareScan	Attempts to enumerate NetBIOS shares.
SMTPScan	Determine whether the target system is running an SMTP service. Returns the service banner and attempts to execute certain basic SMTP commands – EXPN, VRFY, VERB.
SNMPScan	Determines whether the target system is exposing an SNMP service. Attempts to read the values of certain key Object IDs.
SQLScan	Determines whether the target system is running SQL Server. Attempts to login using default users and passwords.
SSHDetect	Checks whether SSH is being run on the target and attempts to determine the package and version.
TCPScan	Performs a SYN scan looking for TCP ports that are definitely open or definitely closed.
TelnetScan	Determines whether the target system is running telnet and attempts to login and return the service banner.
TFTPScan	Determines whether the target system is running TFTP.
UDPScan	Scans UDP ports looking for a positive or negative response. To improve on standard UDP scans, this component uses a database of preformatted packets to elicit responses from known UDP services such as DNS or SQL Server.
UserScan	Attempts to enumerate NetBIOS users on the target system.
XwinScan	Determines whether the target system is running an X Windows service.



Exposure Analysis

Once a test has finished, the Exposure Analysis Service examines the results for running services and for exposures by checking them against service and exposure signatures. It stores the signature matches in the database. These results together with the list of open ports are then compared with the results of either the previous test run against the same IP address or against a baseline set for that node. The results of these comparisons constitute the change analysis performed after every test.

Report Generator

Once exposure analysis is complete, the Report Generator combines the result data, the exposure and service data, and the change data into reports. The following reports are currently supported:

Name	Description
ETR	Engineer's Test Report – This report gives a detailed description of the current state of a single system. It consists of an executive summary, a summary of changes to exposures, services and ports, detailed exposure descriptions, and a section for each of the test components that returned positive results.
XTR	Exception Test Report – Similar to the ETR but the XTR places more emphasis on changes to the system.
MTR	Manager's Test Report – Similar to the ETR but with less detail.
ESR	Engineer's Summary Report – A summary of all the tests performed for a given customer during a specified time period. It consists of an executive summary, a summary of the visibility of each system together with a graph of the number of exposures per IP address, a summary of changes to ports, services, and exposures for each machine, and summaries of exposures, services, and ports over the range of machines.
WCR	Weekly Consolidated Report – Provides a summary of changes over a period of time.
NTR	Network Test Report – Shows the results of a network scan which tests a range of IP addresses for system visibility.

The Report Generator generates PDF reports that are stored on disk by customer or group and by date allowing easy access to a specific report.

Notifier

The Notifier sends a notification email to each customer or user when their reports are ready for examination. This service can be configured to send no message, to send a standard message, or to send a message indicating that urgent action should be taken. This is based on the maximum level of exposure found or change discovered during exposure analysis.