Web Application Scanning with Nessus

Detecting "Low Hanging Fruit" and Environmental Weaknesses

January 15, 2010
(Revision 2)

Brian Martin
Nessus SME

Carole Fennelly
Director, Content & Documentation

Table of Contents

Introduction

Why is it that so many web applications are certified to be compliant with a particular standard, such as PCI DSS and yet are still compromised? According to data compiled by the DatalossDB project, breaches caused by web application flaws comprise 14% of all breaches while another 16% fall into the "hack" category (some of which may be web application related).

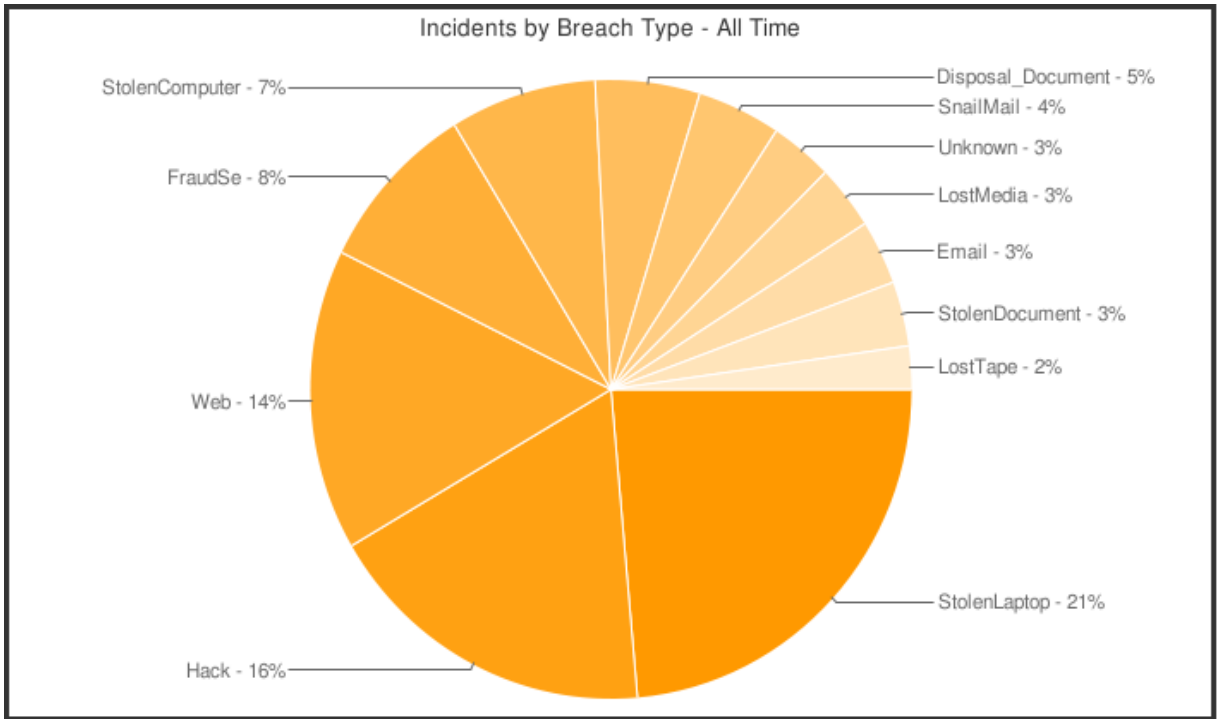


Image courtesy of DatalossDB.org and the Open Security Foundation

Is the scanner the problem? Is it the auditor? On the other hand, is it that the scope of the analysis was too narrow to account for all the other factors that secure the application? The simple answer is that the complexity in the application, network, supporting environment and the audit process makes it necessary to develop a comprehensive approach that includes people, process and technology for web application security assessments.

For the last decade, considerable resources have been directed at developing web-based applications. These range from simple applications that replace paper-based tasks to home banking applications for customer convenience to complex applications that attempt to automate lengthy or difficult tasks. As web servers increasingly host more diverse applications, would-be attackers are focusing on them in attempts to gain access to information or resources. With the prevalence of many web application vulnerability classes, these attacks range from nuisance to full compromises of your organization. Since many of these applications are developed in-house, administrators typically cannot rely on public vulnerability databases to determine if their applications are vulnerable. It is imperative that organizations analyze all the elements that support web applications.

Tenable's dedicated research group is constantly analyzing new threats and developing plugins to detect these threats. The Tenable product suite applies this research on an

enterprise level to correlate information from a variety of sources to help analysts get a complete picture of the supporting environment, in order to better audit and secure web applications.

Overview of Web Application Scanning

The rapid evolution of web applications has forced testing techniques to evolve more quickly in an attempt to not only find known vulnerabilities, but also to find the next threat. Early vulnerabilities in web applications such as cross-site scripting (XSS) were considered a novelty by many, not even rated as a serious risk. History has taught us that XSS attacks can cause serious widespread damage, can be trivial to carry out and can be the difference between achieving PCI compliance or not.

Developing and implementing a proper web application assessment methodology can be an extremely laborious and expensive undertaking. No two web applications are the same, so every test must be performed thoroughly as a single vulnerability could lead to a system, network or organization compromise. To compound the problem, the web application resides on a server that must be examined just as thoroughly; a properly secured web application can be compromised just as quickly through an insecure service in the underlying operating system.

Thoroughly testing a web application requires following a detailed methodology based on years of experience. Examining the operating system platform and web server typically falls in the scope of a network assessment, but since they are crucial to the security of the applications they support, it is just as important to examine them. The web server itself may contain numerous application related vulnerabilities such as header information leaks, dangerous HTTP methods, directories that can be indexed, improper use of SSL certificates, weak ciphers and protocols for secure connections. Moving past the basics, an application must be reviewed from several different perspectives that correspond to types of users: unauthenticated, guest, regular user, administrative user and more.

The way an application performs authentication can be a very complex process; so complex in fact, that many web application scanners have elaborate systems that try to record the transactions that make up the authentication in order to effectively repeat the process to perform authenticated testing. Even then, these scanners fail to properly maintain an authenticated state with the application causing hours of scanning results to be unreliable. The authentication sequence must be tested for a variety of issues such as username complexity and predictability, security of credentials, authentication method, password complexity, password reset security, account enumeration, self-provisioned account creation, brute force attacks, one-time passwords, multi-factor authentication, account lockout issues, challenge/response question security and much more.

Once authenticated, users often have access to an immense footprint of custom written application code that is designed to interact with backend systems, databases and users. Issues such as XSS and SQL injection typically become a bigger threat because the application has established a level of trust with the user. The assignment of privileges to the user must be fully tested to ensure the user cannot access portions of the application that are restricted such as administrative functions. The application must be tested for issues such as horizontal privilege escalation, vertical privilege escalation, split responsibility bypass, session termination, session concurrency, session fixation, cookie handling, URL re-writing, `Referer` header use, data caching, information disclosure, file upload, URL

redirection and a number of input validation issues that must be tested for every part of the application for each defined user role.

Throughout this testing, it is important to consider the application's use of technology such as Java applets, Active-X, Flash, streaming media, videoconferencing, instant messaging, e-mail functionality and published document metadata. Each of these technologies has its own set of tests, concerns and potential vulnerabilities that vary greatly depending on the use and implementation.

The complexity and uniqueness of each web application makes it imprudent to rely solely on an automated vulnerability scanner. A skilled application tester typically finds dozens of vulnerabilities that scanners missed, many of them critical. Automated scanners cannot tell what a page or a variable controls, understand differences in account roles or intuitively guess mistakes a developer may have made. Automated scanners are a useful tool, but only in the hands of a skilled auditor with the experience to validate scanner findings and intuition to locate additional problems. Regardless of how accurate or thorough an application scanner is, it cannot perform all the testing required to perform a comprehensive audit of a web application.

How Tenable Can Help

At Tenable, we believe it is important to audit the settings of the underlying operating system, applications and SQL database before performing an actual web application audit. Tenable's Unified Security Monitoring (USM) approach provides a unification of real-time vulnerability monitoring (24x7 discovery through remediation), critical log/event monitoring and web application scanning capabilities in a single, role-based interface for IT and security users to evaluate, communicate and report the results for effective decision making.

The key features of Tenable's products as they relate to web application scanning are as follows:

Asset Centric Analysis

The Security Center can organize network assets into categories through a combination of network scanning, passive network monitoring and integration with existing asset and network management data tools. This enables an auditor to review all components of a particular web application.

For example, consider a PHP based web application running via Apache on a Red Hat Enterprise Linux (RHEL) system. The application may communicate with middle-ware technology and a backend MySQL database. The entire group of servers comprises the "Store Front" asset. A critical security problem in the RHEL system, Apache modules, PHP, MySQL or a number of other components may equally put the asset at risk.

Data Loss Prevention

Both Nessus and the Passive Vulnerability Scanner (PVS) can identify sensitive data in web applications that may be subject to compliance requirements.

The Nessus scanner can be easily configured to look for common data formats such as credit card numbers and Social Security numbers. It can also be configured to search for

documents with unique corporate identifiers such as employee names, project topics and sensitive keywords. Nessus can perform these searches without an agent and only requires credentials to scan a remote computer.

The PVS can monitor network traffic to identify sensitive traffic in motion over email, web and "chat" activity. It can also identify servers that host office documents on web servers.

The Log Correlation Engine (LCE) can centrally correlate logs from web servers and applications to better understand the types and frequency of attacks.

The Security Center correlates the information about sensitive data gained from Nessus and the PVS that can be useful in several ways:

- Identifying which assets have sensitive data on them can help determine if data is being hosted on unauthorized systems.
- Classifying assets based on the sensitivity of the data they are hosting can simplify configuration and vulnerability auditing by focusing on web application hosts and not the entire network.
- Responding to security incidents or access control violations can be facilitated by knowing the type of information on the target system that helps identify if a system compromise also involves potential theft or modification of data.

Web Application Scanning

Tenable's Nessus scanner has a number of plugins that can aid in web application scanning. This functionality is useful to get an overall picture of the organization's posture before engaging in an exhaustive (and expensive) analysis of the web applications in the environment. Nessus plugins test for common web application vulnerabilities such as SQL injection, cross-site scripting (XSS), HTTP header injection, directory traversal, remote file inclusion and command execution.

Another useful Nessus option is the ability to enable or disable testing of embedded web servers that may be adversely affected when scanned. Many embedded web servers are static and cannot be configured with custom CGI applications. Nessus provides the ability to test these separately to save time and avoid loss of availability of embedded servers.

Nessus provides the ability for the user to adjust how Nessus tests each CGI script and determine the duration of the tests. For example, tests can be configured to stop as soon as a flaw is found or to look for all flaws. This helps to quickly determine if the site will fail compliance without performing the more exhaustive and time-consuming Nessus tests. This "low hanging fruit" approach helps organizations to quickly determine if they have issues that must be addressed before the more intensive tests are run.

Nessus also provides special features for web mirroring, allowing the user to specify which part of the web site will be crawled or excluded. The duration of the crawl process can be limited as well.

Network Vulnerability Scanning

The Nessus vulnerability scanner is a fast and diverse tool that helps any size organization audit their assets for security vulnerabilities. Featuring high-speed discovery, configuration auditing, asset profiling, sensitive data discovery and vulnerability analysis of your security

posture, Nessus scanners can be distributed throughout an entire enterprise, inside DMZs and across physically separate networks. Nessus stays current through automatic updates that pull the latest vulnerability checks directly from Tenable.

Server Patch Auditing

The underlying operating platform must be the first layer of a web application deployment to be inspected for vulnerabilities. If the system can be compromised due to vulnerabilities in core services, it does not matter how secure the application or web server is, it can still be compromised. Nessus can be used to look at the operating system and evaluate the presence of security patches. Using credentials to authenticate to the system, Nessus can enumerate the installed security patches in a matter of minutes and build a list of any that are missing.

Nessus operating system checks are constantly evolving and presently include checks for AIX, CentOS, Debian, Fedora, FreeBSD, Gentoo, HP-UX, Mac OS X, Mandriva, Red Hat, Slackware, Solaris, SuSE, Ubuntu, VMWare ESX and various Windows systems.

Web Server Configuration Auditing

After evaluating the operating system, the web server hosting the application must be examined for configuration options that could manifest into vulnerabilities. Providing the appropriate credentials to Nessus enables the scanner to authenticate to the system and audit the web server configuration file. If any settings are detected that may pose a risk to the server, Nessus will report them and suggest more secure settings. In addition, Nessus can be used to audit PHP settings and other technologies that support web servers. These audit abilities are available to ProfessionalFeed and Security Center customers for Microsoft IIS 6 and Apache.

Database Configuration Auditing

Most web applications interface with a database to manage large amounts of user data. Such databases are typically complex software that may add a significant amount of virtual surface available to attackers. This includes local utilities, remote services that provide access to data and remote management services. Improper configuration of the database may present serious risk to an organization as a remote attacker could leverage flaws in an application to run custom SQL queries against the database. With the proper credentials, Nessus can authenticate to the system and audit the database configuration file to look for common weaknesses and errors that lead to insecure deployments. In addition, Nessus can authenticate directly to the database to audit password settings, insecure stored procedures and more. These audit abilities are available to ProfessionalFeed and Security Center customers for SQL Server 2005, MySQL, Oracle 9 and Oracle 10.

Tenable Enterprise Product Features

Tenable's ability to audit custom web applications is built on several key functions:

- The Security Center can organize network assets into categories through a combination of network scanning, passive network monitoring and integration with existing asset and network management data tools.

- The Security Center can manage scans of software under development to detect vulnerabilities early in the development cycle.
- Nessus can be used to audit the underlying operating system for any vulnerability in the OS, application or database.
- Nessus can be used to audit the configuration of the operating system, application and database.
- Nessus can perform a variety of web application audits to test for common web application vulnerabilities such as SQL injection, XSS, HTTP header injection, directory traversal, remote file inclusion and command execution.
- Nessus has the ability to send POST requests in addition to GET requests, which enables testing of HTML forms for vulnerabilities.
- Nessus has the ability to enable or disable testing of embedded web servers that may be adversely affected when scanned.
- Nessus scans can be configured to stop as soon as a flaw is found or to look for all flaws. This helps to quickly determine if issues need to be addressed before running exhaustive scans.
- Nessus provides special features for web mirroring, allowing the user to specify which part of the web site will be crawled or not.
- The LCE can be used to monitor any logs generated by software under development to detect anomalies and errors. The Security Center can perform enterprise-wide log searches that could indicate an installation that is not in sync with the rest of the deployment.
- On a production system, the PVS can monitor network traffic to look for evidence of SQL injection issues and other types of web application errors.
- The PVS can monitor network traffic for particular data types such as encrypted or sensitive data to ensure that web application are not passing sensitive data in the clear.
- The LCE can make use of web and database logs to look for web application probes and testing.

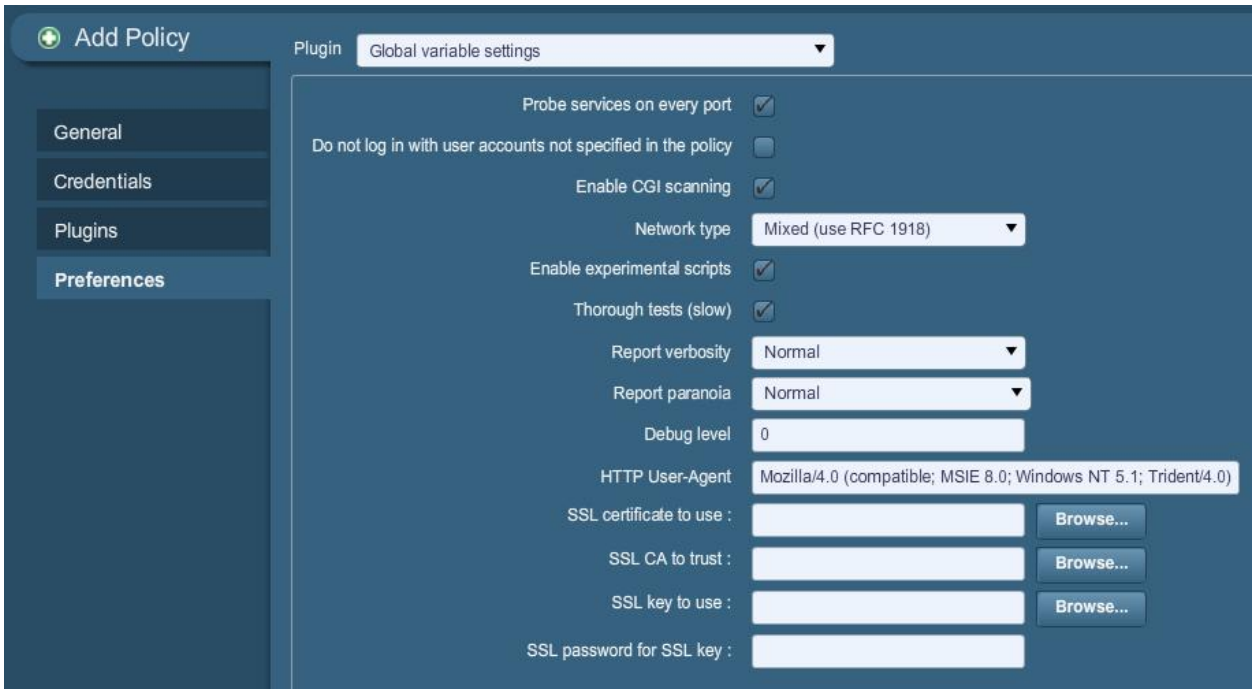
Nessus also has the ability to audit the content of specific files. If an issue with a customer web application system is discovered, it can easily be scanned without the need to program a new check in Nessus.

For non-web based applications, Nessus performs a wide variety of third party library audits for vulnerabilities. Libraries tested include .Net, Java, PHP and Adobe AIR.

Nessus Web Application Auditing Features

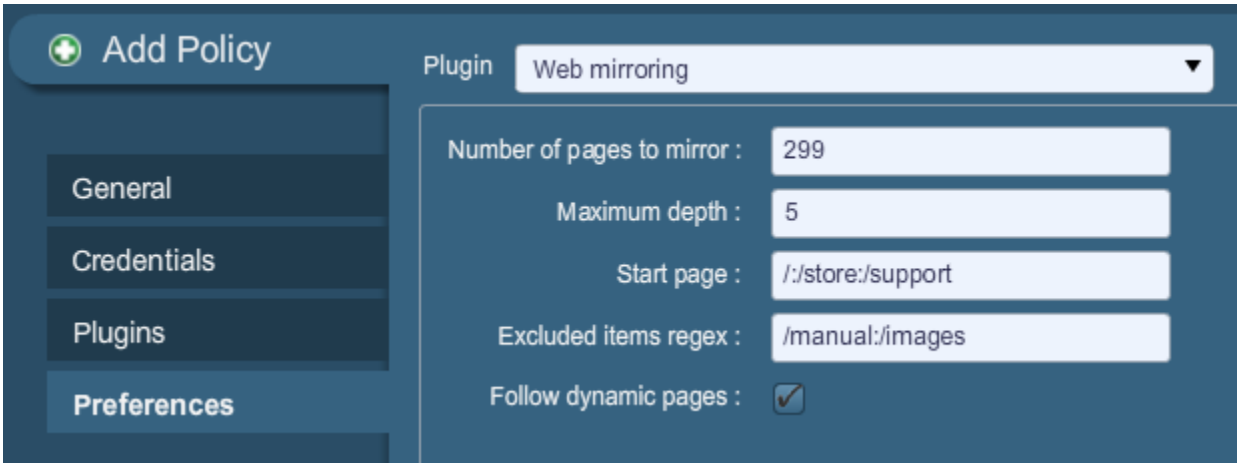
As of June 2009, Nessus received significant enhancements in its ability to assess web applications. Users have greater flexibility in configuring the web mirroring process and web application test settings that control the granularity of testing. In addition, several new types of tests are performed to provide more robust assessments.

With the release of Nessus 4, CGI scanning is not enabled by default. The "Global variable settings" under the "Preferences" tab allow for one-click enabling of CGI scanning, which is separate from custom application testing. Enabling CGI scanning will direct Nessus to look for known problems in public and commercial software, independent of the web mirroring process. These settings further allow a user to enable experimental scripts and thorough tests, control report verbosity and paranoia as well as set a custom HTTP User-Agent string to be used during testing. If a client side SSL certificate is required to interact with an application, it can be specified here.



Nessus uses a native spider process to crawl a web server and its associated applications. This process allows it to examine the technology present and more efficiently conduct tests. Users can control the maximum number of pages to mirror, define multiple start pages and choose to follow dynamic pages as well as list pages or directories to be excluded from the mirror process.

The "Excluded items regex" is a powerful method for establishing granular mirroring exemptions. For example, if you do not want to crawl (or scan) "/manual" and do not want to test any Perl-based CGI, set this field to: **(^/manual)|(\.pl(\?.*)?$)**.



The "**Web Application Tests Settings**" allows users to enable custom application testing. This directs Nessus to check applications for a wide range of vulnerability classes including:

- SQL injection (SQLi) – A code injection method targeting the application's database software (e.g., MySQL, Oracle). SQLi can result in the disclosure of sensitive

information, manipulation of private data or potentially gaining unrestricted access to the machine hosting the database.

- Cross-site scripting (XSS) – A code injection flaw allowing an attacker to inject arbitrary script code into a web page that will be executed in the context of the security relationship between the victim and the application. This is frequently used to steal authentication credentials of unsuspecting users.
- HTTP header injection – A type of injection attack that targets the application's use and reliance on HTTP headers. Such attacks can be used to set arbitrary headers (e.g., CRLF injection) or bypass access controls (e.g., Referer header).
- Directory Traversal – An input manipulation attack that uses directory traversal sequences to access or manipulate arbitrary files and resources on the remote web server.
- Remote File Inclusion – A style of cross-server attack that manipulates an application to load executable content from a third-party server. This method allows an attacker to execute arbitrary commands with the same privileges as the targeted web server.
- Command Execution – A code injection flaw in which an application does not properly sanitize user input, allowing for the injection or arbitrary operating system commands. Such flaws typically allow an attacker to quickly and easily take complete control of a server.

The "**Maximum run time**" setting limits how much time Nessus will spend performing these tests. Large complex applications can take as long as a day or more to complete automated testing. In addition to testing an application via GET requests, Nessus can expand testing to include POST requests as well.

The "**Combinations of arguments values**" controls how each parameter of an application is tested. This choice may have the most significant impact on application testing; both in the time required and how thorough the testing is performed:

- **one value** – This will test one parameter at a time with an attack string, without trying non-attack variations for additional parameters. For example, Nessus would attempt "`/test.php?arg1=XSS&b=1&c=1`" where "b" and "c" allows other values, without testing each combination. This is the quickest method of testing with the smallest data set generated.
- **all pairs (slower but efficient)** – This form of testing tries a representative data set of tests based on the "All-pairs testing method". While testing multiple parameters, it will test an attack string, variations for a single variable and then use the first value for all other variables. For example, Nessus would attempt "`/test.php?a=XSS&b=1&c=1&d=1`" and then cycle through the variables so that one is given the attack string, one is cycled through all possible values (as discovered during the mirror process) and any other variables are given the first value. In this case, Nessus would never test for "`/test.php?a=XSS&b=3&c=3&d=3`" when the first value of each variable is "1".
- **some pairs** – Like "all pairs" testing, this will try to test a representative data set based on the "All-pairs" method. However, for each parameter discovered, Nessus will only test using a maximum of three valid input variables.
- **all combinations (extremely slow)** – This method of testing will do a fully exhaustive test of all possible combinations of attack strings and valid input to variables. Where "All-pairs" testing seeks to create a smaller data set as a tradeoff for speed, "all combinations" makes no compromise on time and uses a complete data set of tests.

- **some combinations** – Like "all combinations" testing, this will perform tests using a combination of attack strings and valid input. However, for each parameter discovered, Nessus will only test using a maximum of three valid input variables.

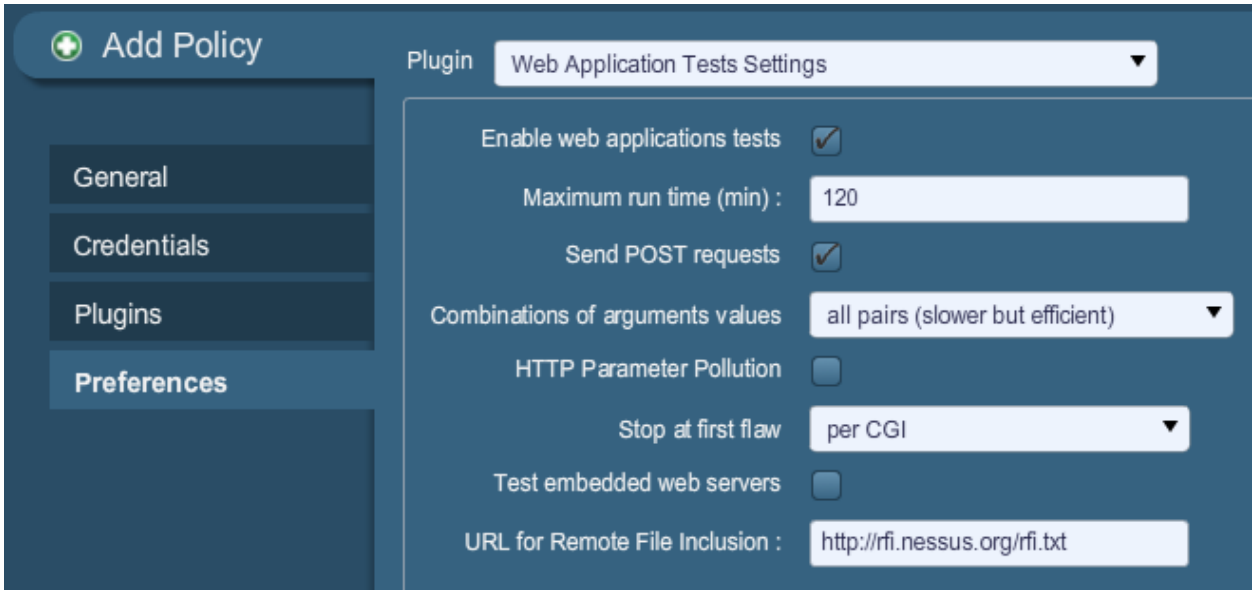
If enabled, Nessus can use a technique called "HTTP Parameter Pollution" that attempts to break up application test requests. This method may allow Nessus to bypass some types of filtering (e.g., Web Application Firewalls) or circumvent an application's logic for expected input.

Nessus has the ability to stop testing at specified points:

- **per port (quicker)** – For CGI testing, once Nessus finds a web application flaw, it will stop testing all applications on this port for that host. This is handy for PCI DSS testing, as you will fail the test if just one flaw is found.
- **per CGI** – Nessus will stop once it has found a flaw in a particular CGI script. You can save time by having Nessus stop at each CGI, then go back and perform manual testing and/or source code review on the applications that failed.
- **look for all flaws (slower)** – This options will cause Nessus to continue testing until it exhausts all options as defined in your settings, regardless of the number of flaws found.

If enabled, Nessus will test embedded web servers and associated applications. Often, embedded web servers are static and cannot be configured with custom CGI applications. In addition, scanning embedded web servers can be very slow and/or cause problems on the device. Therefore, it is recommended that they be tested separately.

The final option allows a user to specify the location of a file hosted on a third-party web site, to be used while testing for remote file inclusion. While Nessus will attempt to test using a safe file hosted on Tenable's web site, this may not work if the systems being tested do not have Internet connectivity or subjected to some kind of content filtering.



Once configured and a scan launched, Nessus will report any issues found along with the relevant details required to manually validate the results and determine how best to

remediate the issue. For example, if cross-site scripting flaws are found on a web site, the output may look like the following:

CGI Generic Cross-Site Scripting Vulnerability

Synopsis :

The remote web server is prone to cross-site scripting attacks.

Description :

The remote web server hosts CGI scripts that fail to adequately sanitize request strings with malicious JavaScript. By leveraging this issue, an attacker may be able to cause arbitrary HTML and script code to be executed in a user's browser within the security context of the affected site.

See also :

http://en.wikipedia.org/wiki/Cross-site_scripting

Solution :

Restrict access to the vulnerable application. Contact the vendor for a patch or upgrade.

Risk factor :

Medium / CVSS Base Score : 4.3
(CVSS2#AV:N/AC:M/Au:N/C:N/I:P/A:N)

Plugin output :

Using the GET HTTP method, Nessus found that :

+ The following resources may be vulnerable to Cross site scripting :

/php-ids/w3af/audit/xss/no_tag_xss.php?text=<<<<<<<<<<foobar>>>>>&country=CA&language=en
-------- output --------

Start--
<img src="<<<<<<<<<<foobar>>>>>">
--End

/mod_security/w3af/audit/xss/no_tag_xss.php?text=<<<<<<<<<<foobar>>>>>&1853133142=16-36-1
-------- output --------

Start--
<img src="<<<<<<<<<<foobar>>>>>">
--End

/mod_security/w3af/audit/xss/simple_xss_no_quotes.php?text=<<<<<<<<<<foobar>>>>>&18531331
-------- output --------

About Tenable Network Security

Tenable, headquartered in Columbia, Md., USA, is the world leader in Unified Security Monitoring. Tenable provides agent-less solutions for continuous monitoring of vulnerabilities, configurations, data leakage, log analysis and compromise detection. For more information, please visit us at http://www.tenablesecurity.com/.

TENABLE Network Security, Inc.
7063 Columbia Gateway Drive
Suite 100
Columbia, MD 21046
TEL: 410-872-0555
http://www.tenablesecurity.com/