RELATÓRIO SEGUNDA SEMANA

Lucas L Reis Muniz

Introdução

Nesta semana será mostrado um escaneamento de vulnerabilidades na infraestrutura e nos recursos web de um servidor criado pelo metasploitable 2. Esses testes foram feitos no SO (sistema operacional) Kali Linux.

Para visualizar melhor os resultados obtidos, recomendo que utilize, no mínimo, zoom de 300% do próprio PDF nas imagens.

Termos Importantes

VirtualBox: software que permite a instalação e utilização de um sistema operacional dentro do outro, assim como seus respectivos softwares, como dois ou mais computadores independentes, mas compartilhando o mesmo hardware. (fonte wikipedia)

Aplicações web: Os apps web são sistemas que rodam na internet. São como sistemas tradicionais que recebem uma entrada, processam informação e emitem uma saída. Contudo, eles são rodados e interpretados por um navegador. (fonte tegra.com)

Proxy: é um servidor que age como um intermediário para requisições de clientes solicitando recursos de outros servidores. Um cliente conecta-se ao servidor *proxy*, solicitando algum serviço, como um arquivo, conexão, página web ou outros recursos disponíveis de um servidor diferente, e o *proxy* avalia a solicitação como um meio de simplificar e controlar sua complexidade. (fonte wikipedia)

URL: O termo URL é a abreviação de Uniform Resource Locator, ou Localizador Uniforme de Recursos. Sendo direto, URL é a mesma coisa de endereço web, o texto que você digita na barra de endereços de seu navegador para acessar uma determinada página ou serviço. (fonte tecnoblog.net)

Spider: ferramenta usada para descobrir URLs em um site particular. Começa com uma lista de URLs para visitar, chamadas de “sementes”. O Spider visita essas URLs e identifica todos os hyperlinks na página e os adiciona às URLs visitadas. Esse processo continua recursivamente enquanto forem achados URLs.

Man-in-the-middle: é uma forma de ataque em que os dados trocados entre duas partes (por exemplo, você e o seu banco), são de alguma forma interceptados, registrados e possivelmente alterados pelo atacante sem que as vitimas percebam. (fonte wikipedia)

owasp zap: software capaz de encontrar vulnerabilidades em aplicações web. Owasp zap funciona como um man-in-the-middle proxy, o qual faz um spider e depois utiliza um escaneamento ativo, o qual procura potenciais vulnerabilidades usando ataques conhecidos contra o alvo selecionado.

Execução

Primeiramente foi executado, usando o **VirtualBox**, um servidor fictício, **metasploitable 2** (os detalhes de como foi feito serão omitidos porque nosso objetivo é focar na busca e resultado das vulnerabilidades). Depois de aberto, utilizei o endereço de **ip do servidor (192.168.0.103)** para realizar o **scan web** com a ferramenta **owasp zap**, rodando o scan no modo automático e, assim encontrando os resultados.

Depois disso foi executado o **OpenVAS**, utilizando o mesmo endereço ip anterior, o qual apresenta uma abordagem maior em termos de infraestrutura (pode nos dizer o sistema operacional usado e outras informações a mais). Os resultados classificados com severity low não foram registrados neste relatório.

Resultados owasp zap:

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	1
Medium	2
Low	7
Informational	4

Alert Detail

High (Medium)	Path Traversal
Description	The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will execute or reveal the contents of arbitrary files anywhere on the web server. Any device that exposes an HTTP-based interface is potentially vulnerable to Path Traversal. Most web sites restrict user access to a specific portion of the file-system, typically called the "web document root" or "CGI root" directory. These directories contain the files intended for user access and the executable necessary to drive web application functionality. To access files or execute commands anywhere on the file-system, Path Traversal attacks will utilize the ability of special-characters sequences. The most basic Path Traversal attack uses the "../" special-character sequence to alter the resource location requested in the URL. Although most popular web servers will prevent this technique from escaping the web document root, alternate encodings of the "../" sequence may help bypass the security filters. These method variations include valid and invalid Unicode-encoding ("..%u2216" or "..%c0%af") of the forward slash character, backslash characters ("..\") on Windows-based servers, URL encoded characters "%2e%2e%2f"), and double URL encoding ("..%255c") of the backslash character. Even if the web server properly restricts Path Traversal attempts in the URL path, a web application itself may still be vulnerable due to improper handling of user-supplied input. This is a common problem of web applications that use template mechanisms or load static text from files. In variations of the attack, the original URL parameter value is substituted with the file name of one of the web application's dynamic scripts. Consequently, the results can reveal source code because the file is interpreted as text instead of an executable script. These techniques often employ additional special characters such as the dot (".") to reveal the listing of the current working directory, or "%00" NULL characters in order to bypass rudimentary file extension checks.
URL	http://192.168.0.103/mutillidae/?page=%2Fetc%2Fpasswd
Method	GET
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=%2Fetc%2Fpasswd
Method	GET
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?do=toggle-hints&page=%2Fetc%2Fpasswd
Method	GET
Parameter	page
Attack	/etc/passwd

Figure 1: 1zap

Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=text-file-viewer.php
Method	POST
Parameter	textfile
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?choice=nmap&initials=ZAP&page=%2Fetc%2Fpasswd&user-poll-php-submit-button=Submit+Vote
Method	GET
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=%2Fetc%2Fpasswd
Method	POST
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=source-viewer.php
Method	POST
Parameter	phpfile
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=%2Fetc%2Fpasswd&password=ZAP&user-info-php-submit-button=View+Account+Details&username=ZAP
Method	GET
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=%2Fetc%2Fpasswd&username=anonymous
Method	GET
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?page=source-viewer.php
Method	POST

Figure 2: 2zap

URL	http://192.168.0.103/mutillidae/index.php?page=source-viewer.php
Method	POST
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
URL	http://192.168.0.103/mutillidae/index.php?forwardurl=http%3A%2F%2Fwww.php.net%2F&page=%2Fetc%2Fpasswd
Method	GET
Parameter	page
Attack	/etc/passwd
Evidence	root:x:0:0
Instances	11
Solution	Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue." For filenames, use stringent whitelists that limit the character set to be used. If feasible, only allow a single "." character in the filename to avoid weaknesses, and exclude directory separators such as "/". Use a whitelist of allowable file extensions. Warning: if you attempt to cleanse your data, then do so that the end result is not in the form that can be dangerous. A sanitizing mechanism can remove characters such as '.' and ';' which may be required for some exploits. An attacker can try to fool the sanitizing mechanism into "cleaning" data into a dangerous form. Suppose the attacker injects a '.' inside a filename (e.g. "sensi.tiveFile") and the sanitizing mechanism removes the character resulting in the valid filename, "sensitiveFile". If the input data are now assumed to be safe, then the file may be compromised. Inputs should be decoded and canonicalized to the application's current internal representation before being validated. Make sure that your application does not decode the same input twice. Such errors could be used to bypass whitelist schemes by introducing dangerous inputs after they have been checked. Use a built-in path canonicalization function (such as realpath() in C) that produces the canonical version of the pathname, which effectively removes ".." sequences and symbolic links. Run your code using the lowest privileges that are required to accomplish the necessary tasks. If possible, create isolated accounts with limited privileges that are only used for a single task. That way, a successful attack will not immediately give the attacker access to the rest of the software or its environment. For example, database applications rarely need to run as the database administrator, especially in day-to-day operations. When the set of acceptable objects, such as filenames or URLs, is limited or known, create a mapping from a set of fixed input values (such as numeric IDs) to the actual filenames or URLs, and reject all other inputs. Run your code in a "jail" or similar sandbox environment that enforces strict boundaries between the process and the operating system. This may effectively restrict which files can be accessed in a particular directory or which commands can be executed by your software. OS-level examples include the Unix chroot jail, AppArmor, and SELinux. In general, managed code may provide some protection. For example, java.io.FilePermission in the Java SecurityManager allows you to specify restrictions on file operations. This may not be a feasible solution, and it only limits the impact to the operating system; the rest of your application may still be subject to compromise.
Reference	http://projects.webappsec.org/Path-Traversal http://cwe.mitre.org/data/definitions/22.html
CWE Id	22
WASC Id	33
Source ID	1

Figure 3: 3zap

Medium (Medium)	X-Frame-Options Header Not Set
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	http://192.168.0.103/twiki/bin/view/TWiki/TWikiFuncModule?rev=1.2
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/rdiff/Know/WinDoze95Crash
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/edit/TWiki/TWikiCodevTWikiDocumentation?topicparent=TWiki.TWikiHistory
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/search/Main/SearchResult?regex=on&scope=text&search=Web%20*Search%5B%5EA-Za-z%5D&web=all;
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/view/TWiki/TWikiFuncModule?rev=1.1
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/rdiff/TWiki/TWikiRegistration?rev1=1.8&rev2=1.7
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/oops/Main/WebNotify?param1=1.7&param2=1.7&template=oopsmore
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiDocGraphics?filename=attachfile.gif&revInfo=1
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/mutillidae/index.php?page=dns-lookup.php
Method	POST
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/preview/TWiki/TWikiRegistration
Method	POST
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/view/Main/OfficeLocations?raw=on&rev=1.4
Method	GET
Parameter	X-Frame-Options

Figure 4: 4zap

Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/rdiff/TWiki/TWikiFormTemplate?rev1=1.16&rev2=1.15
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/view/TWiki/GrantBow?skin=print
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/search/Know/SearchResult?regex=on&scope=text&search=Web%20*Rss%5B%5EA-Za-z%5D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/edit/Main/TWikiPreferences?topicparent=Main.TWikiGuest
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/search/Know/SearchResult?regex=on&scope=text&search=Os%20*Win%5B%5EA-Za-z%5D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/view/TWiki/ManpreetSingh?rev=1.1
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/rdiff/TWiki/KlausWriessnegger
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/rdiff/TWiki/TextFormattingFAQ?rev1=1.14&rev2=1.13
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.0.103/twiki/bin/oops/Know/WebPreferences?param1=1.11&param2=1.10&template=oopsmore
Method	GET
Parameter	X-Frame-Options
Instances	4663
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	16
WASC Id	15

Figure 5: 5zap

Medium (Medium)	Application Error Disclosure
Description	This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.
URL	http://192.168.0.103/twiki/bin/attach/Main/GrantBow
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiAccessControl
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/WikiSyntax
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/WebNotify
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiFuncModule
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/WikiName
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/Main/TWikiGuest
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiDocGraphics?filename=searchtopic.gif&revInfo=1
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiRegistration
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/ManagingTopics
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/mutillidae/index.php?page=view-someones-blog.php

Figure 6: 6zap

Evidence	Internal Server Error
URL	http://192.168.0.103/mutillidae/index.php?page=view-someones-blog.php
Method	GET
Evidence	Table 'metasploit.accounts' doesn't exist
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiUsernameVsLoginUsername
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/WebIndex
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/WebSearch
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/FileAttachment?filename=Smile.gif&revInfo=1
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/ChangePassword
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/TWikiDocumentation
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/JohnAltstadt
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/PeterFokkinga
Method	GET
Evidence	Internal Server Error
URL	http://192.168.0.103/twiki/bin/attach/TWiki/WebHome
Method	GET
Evidence	Internal Server Error
Instances	238
Solution	Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.
Reference	
CWE Id	200
WASC Id	13

Figure 7: 7zap

Resultados OpenVAS:

Figure 8: 1OpenVAS

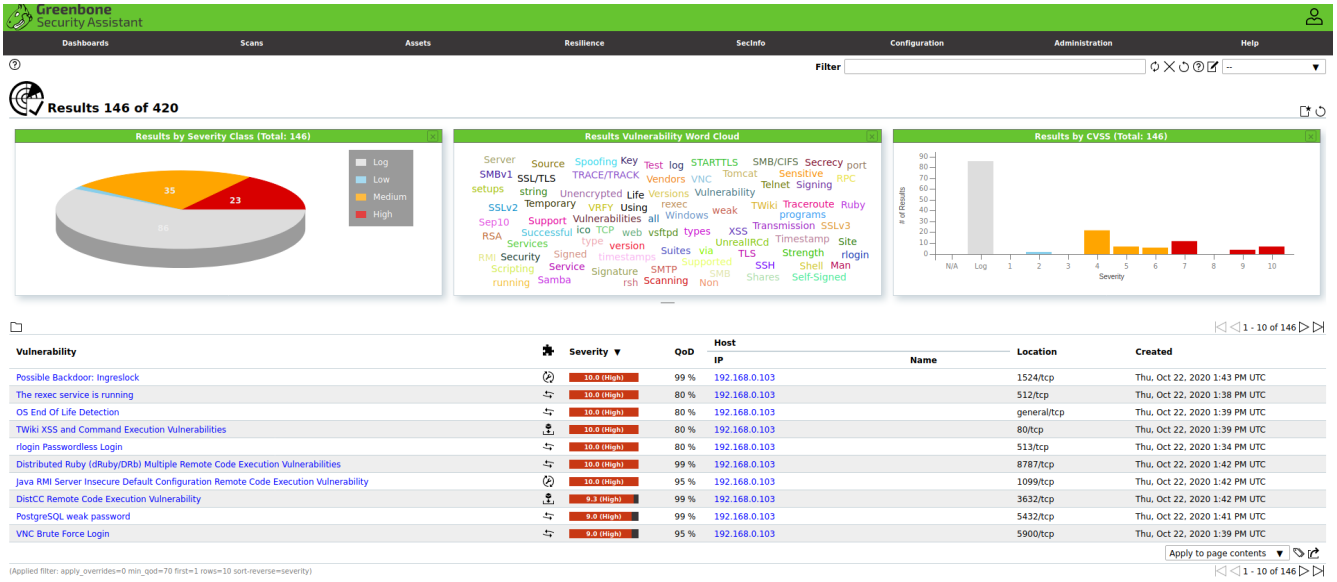

Figure 9: 2OpenVAS

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
PostgreSQL weak password	9.0 (High)	99 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:41 PM UTC
FTP Brute Force Logins Reporting	7.5 (High)	95 %	192.168.0.103		2121/tcp	Thu, Oct 22, 2020 1:48 PM UTC
Apache Tomcat AJP RCE Vulnerability (Ghostcat)	7.5 (High)	99 %	192.168.0.103		8009/tcp	Thu, Oct 22, 2020 1:43 PM UTC
Check for Backdoor in UnrealIRCd	7.5 (High)	70 %	192.168.0.103		6667/tcp	Thu, Oct 22, 2020 1:42 PM UTC
PHP-CGI-based setups vulnerability when parsing query string parameters from php files.	7.5 (High)	95 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:44 PM UTC
FTP Brute Force Logins Reporting	7.5 (High)	95 %	192.168.0.103		21/tcp	Thu, Oct 22, 2020 1:48 PM UTC
vsftpd Compromised Source Packages Backdoor Vulnerability	7.5 (High)	99 %	192.168.0.103		21/tcp	Thu, Oct 22, 2020 1:42 PM UTC
rsh Unencrypted Cleartext Login	7.5 (High)	80 %	192.168.0.103		514/tcp	Thu, Oct 22, 2020 1:38 PM UTC
Test HTTP dangerous methods	7.5 (High)	99 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:42 PM UTC
vsftpd Compromised Source Packages Backdoor Vulnerability	7.5 (High)	99 %	192.168.0.103		6200/tcp	Thu, Oct 22, 2020 1:42 PM UTC

Apply to page contents ▼

(Applied filter: apply_overrides=0 min_qod=70 first=11 rows=10 sort-reverse=severity)

11 - 20 of 146

Figure 10: 3OpenVAS

21 - 30 of 146

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
FTP Brute Force Logins Reporting	7.5 (High)	95 %	192.168.0.103		2121/tcp	Thu, Oct 22, 2020 1:48 PM UTC
SSH Brute Force Logins With Default Credentials Reporting	7.5 (High)	95 %	192.168.0.103		22/tcp	Thu, Oct 22, 2020 1:48 PM UTC
Test HTTP dangerous methods	7.5 (High)	99 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:42 PM UTC
Multiple Vendors STARTTLS Implementation Plaintext Arbitrary Command Injection Vulnerability	6.8 (Medium)	99 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:43 PM UTC
UnrealIRCd Authentication Spoofing Vulnerability	6.8 (Medium)	80 %	192.168.0.103		6667/tcp	Thu, Oct 22, 2020 1:35 PM UTC
TWiki Cross-Site Request Forgery Vulnerability - Sep10	6.8 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:39 PM UTC
Anonymous FTP Login Reporting	6.4 (Medium)	80 %	192.168.0.103		21/tcp	Thu, Oct 22, 2020 1:34 PM UTC
TWiki Cross-Site Request Forgery Vulnerability	6.0 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:39 PM UTC
Samba MS-RPC Remote Shell Command Execution Vulnerability (Active Check)	6.0 (Medium)	99 %	192.168.0.103		445/tcp	Thu, Oct 22, 2020 1:42 PM UTC
SSL/TLS: OpenSSL CCS Man in the Middle Security Bypass Vulnerability	5.8 (Medium)	70 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:43 PM UTC

Apply to page contents ▼

(Applied filter: apply_overrides=0 min_qod=70 sort-reverse=severity rows=10 first=21)

21 - 30 of 146

Figure 11: 4OpenVAS

31 - 40 of 146

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
SSL/TLS: OpenSSL CCS Man in the Middle Security Bypass Vulnerability	5.8 (Medium)	70 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:43 PM UTC
awiki Multiple Local File Include Vulnerabilities	5.0 (Medium)	99 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:43 PM UTC
SSL/TLS: Certificate Expired	5.0 (Medium)	99 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:37 PM UTC
/doc directory browsable	5.0 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:38 PM UTC
SSL/TLS: Certificate Expired	5.0 (Medium)	99 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
Check if Mailserver answer to VRFY and EXPN requests	5.0 (Medium)	99 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:38 PM UTC
Cleartext Transmission of Sensitive Information via HTTP	4.8 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:39 PM UTC
FTP Unencrypted Cleartext Login	4.8 (Medium)	70 %	192.168.0.103		21/tcp	Thu, Oct 22, 2020 1:34 PM UTC
VNC Server Unencrypted Data Transmission	4.8 (Medium)	70 %	192.168.0.103		5900/tcp	Thu, Oct 22, 2020 1:35 PM UTC
FTP Unencrypted Cleartext Login	4.8 (Medium)	70 %	192.168.0.103		2121/tcp	Thu, Oct 22, 2020 1:34 PM UTC

Apply to page contents ▼

(Applied filter: apply_overrides=0 min_qod=70 sort-reverse=severity rows=10 first=31)

31 - 40 of 146

Figure 12: 5OpenVAS

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
Cleartext Transmission of Sensitive Information via HTTP	4.8 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:39 PM UTC
SSL/TLS: RSA Temporary Key Handling 'RSA_EXPORT' Downgrade Issue (FREAK)	4.3 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
jQuery < 1.6.3 XSS Vulnerability	4.3 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:39 PM UTC
SSL/TLS: Report Weak Cipher Suites	4.3 (Medium)	98 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: 'DHE_EXPORT' Man in the Middle Security Bypass Vulnerability (LogJam)	4.3 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection	4.3 (Medium)	98 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection	4.3 (Medium)	98 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability	4.3 (Medium)	99 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:45 PM UTC
SSL/TLS: SSLv3 Protocol CBC Cipher Suites Information Disclosure Vulnerability (POODLE)	4.3 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
jQuery < 1.9.0 XSS Vulnerability	4.3 (Medium)	80 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:39 PM UTC

(Applied filter: apply_overrides=0 min_qod=70 rows=10 first=41 sort-reverse=severity)

41 - 50 of 146

Figure 13: 6OpenVAS

Vulnerability	Severity ▼	QoD	Host IP	Host Name	Location	Created
phpMyAdmin 'error.php' Cross Site Scripting Vulnerability	4.3 (Medium)	99 %	192.168.0.103		80/tcp	Thu, Oct 22, 2020 1:44 PM UTC
SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection	4.3 (Medium)	98 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: 'DHE_EXPORT' Man in the Middle Security Bypass Vulnerability (Logjam)	4.3 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: RSA Temporary Key Handling 'RSA_EXPORT' Downgrade Issue (FREAK)	4.3 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: Diffie-Hellman Key Exchange Insufficient DH Group Strength Vulnerability	4.0 (Medium)	80 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: Diffie-Hellman Key Exchange Insufficient DH Group Strength Vulnerability	4.0 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: Certificate Signed Using A Weak Signature Algorithm	4.0 (Medium)	80 %	192.168.0.103		5432/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSL/TLS: Certificate Signed Using A Weak Signature Algorithm	4.0 (Medium)	80 %	192.168.0.103		25/tcp	Thu, Oct 22, 2020 1:37 PM UTC
SSH Weak MAC Algorithms Supported	2.6 (Low)	95 %	192.168.0.103		22/tcp	Thu, Oct 22, 2020 1:35 PM UTC
TCP timestamps	2.6 (Low)	80 %	192.168.0.103		general/tcp	Thu, Oct 22, 2020 1:26 PM UTC

(Applied filter: apply_overrides=0 min_qod=70 rows=10 first=51 sort-reverse=severity)

51 - 60 of 146

Figure 14: 7OpenVAS

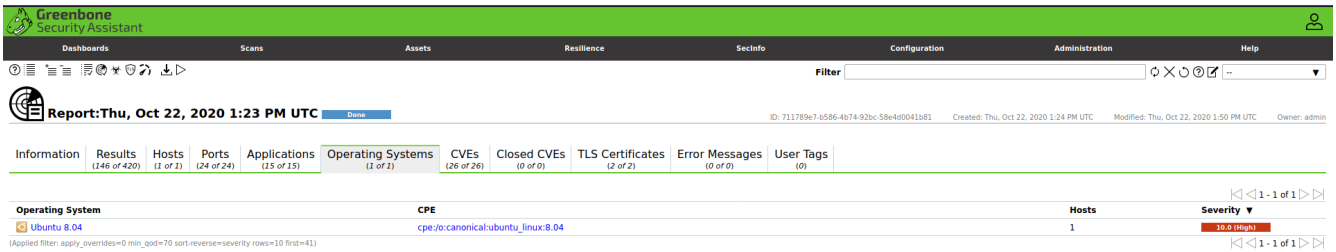

Figure 15: 8OpenVAS

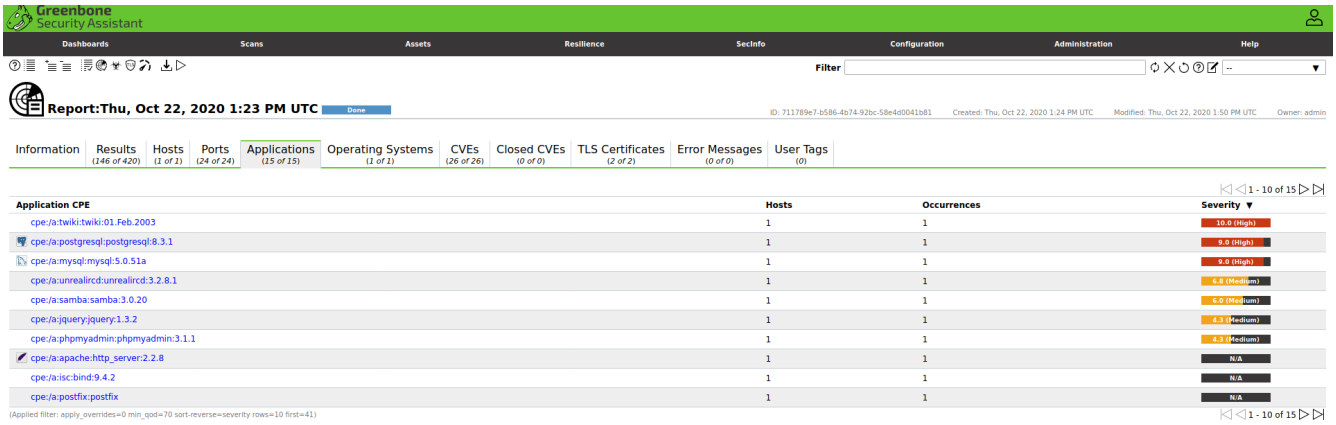

Figure 16: 9OpenVAS

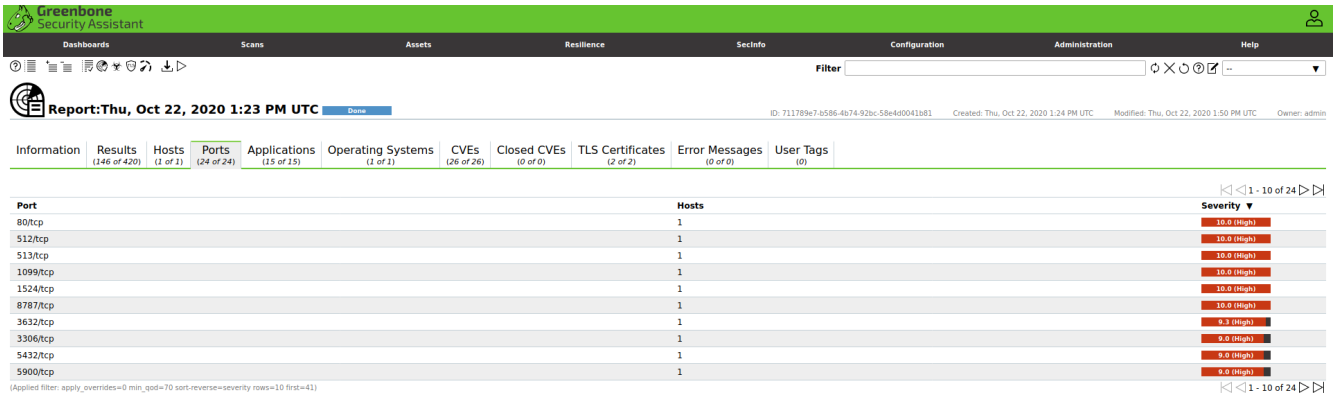

Figure 17: 10OpenVAS

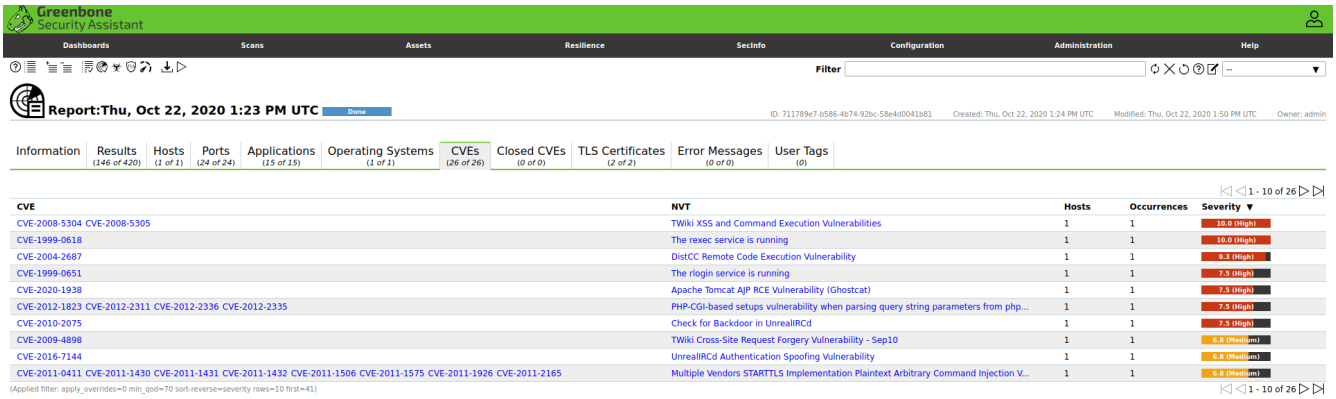

Figure 18: 11OpenVAS