Project Report

Vulnerability Assessment

Benjamin Huang

May 2nd, 2023

Table of Contents

Table of Figures

Executive Summary

This is a report on a security assessment on a network of virtual machines. The scope of this security assessment will include the entire virtual network. The virtual machines (VM) were configured using VirtualBox. A network diagram is provided in Figure 1. The network has been divided into three sub-networks or Virtual Local Area Networks (VLAN) using a firewall. The first VLAN is the internal network that includes a VM called PC1 and VM for scanning purposes. The second VLAN is the Demilitarized Zone (DMZ) which consists of the Web Server, DNS Server, and VM for scanning purposes. The third VLAN is the external VLAN which is meant to act as a public facing network and has a VM for scanning purposes.

The network is segmented into three parts because it improves security, allows access control, and better network performance. Segmenting the network into different zones increases security because it isolates sensitive assets from the public facing network and reduces the attack surface. The firewall helps with access control because each zone can be set to only allow access from specific IP addresses, ports numbers, and destination IP. Segmentation can also improve network performance by reducing network congestion and optimizing the flow of data between different zones and preventing Denial of Service (DoS) attacks. The results of the report will help showcase the benefits of segmenting an organization's network.

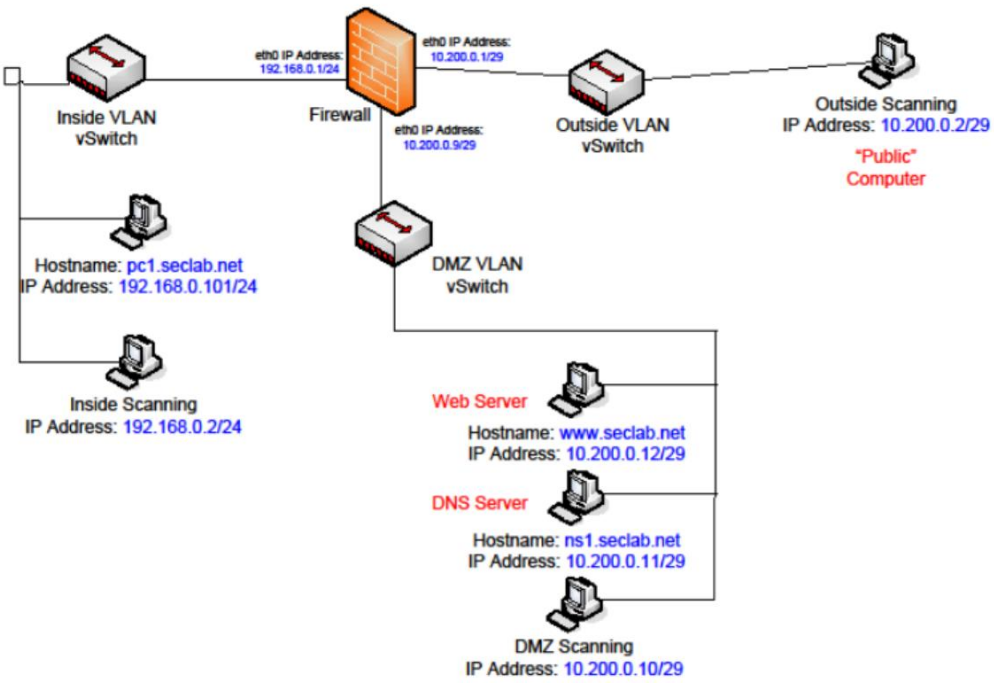

Figure 1: Network Layout

The tools that will be used in this assessment include the open-source Greenbone software and the open-source OWASP ZAP software. Greenbone is used to scan and analyze networks for security vulnerabilities and provides a web-based interface for managing and

running vulnerability scans. Greenbone can scan networks, applications, and systems for vulnerabilities and provide detailed reports on the vulnerabilities that have been identified. The software can also prioritize them based on their severity and potential impact on the organization. The ZAP tool is a web application security testing tool with a Graphical User Interface (GUI) that is designed to help identify vulnerabilities in web apps. ZAP can perform an active scan of the webserver set up in the DMZ by sending crated requests to the server and identifying vulnerabilities.

Greenbone Vulnerabilities Found

The Greenbone scan was performed on the internal, DMZ, and external network and targeted the IP 10.200.0.12 which is the IP of the server "www.seclab.net". The report will go through each finding from the highest severity for each scan. Many of these vulnerabilities could be remediated through updating to the newest release of software, closing parts, using stronger password, using stronger encryption, upgrading to stronger protocols, phasing out unsupported software, or closing ports that are unneeded. Each of the High and Medium findings are listed below. The remediations can be found in the Remediation section.

The biggest difference is that the External Scan had a lot less vulnerabilities found than the Internal and DMZ scan. The external network is where any public can access that side of the firewall, and this includes attackers which means the firewall must have more protection. That is why the external scan has the least vulnerabilities found and the vulnerabilities that are found have to do with port 80 which is used for HTTP connections.

The only difference between the DMZ and Internal scans was that the DMZ scan had the High 1099/TCP vulnerability. It is called "Java RMI Server Insecure Default Configuration RCE Vulnerability" and this allows unauthenticated remote attacked to execute code on the system with elevated privileges. It makes sense that this vulnerability showed up because the web server is on the DMZ side of the network.

CVSS Score	Internal	DMZ	External	Total
High	17	18	5	40
Medium	38	38	12	88
Low	6	6	1	13
Total	13	0	0	13

Figure 2: Greenbone Findings

Internal Findings

1) High 5432/tcp
    a. PostgreSQL weak password
    b. SSL/TLS: OpenSSL CCS Man in the Middle Security Bypass Vulnerability
2) High 3632/tcp
    a. DistCC RCE Vulnerability (CVE-2004-2687)
3) High 5900/tcp
    a. VNC Brute Force Login
4) High 1524/tcp
    a. Possible Backdoor: Ingreslock

5) High 8787/tcp
    a. Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution Vulnerabilities
6) High 21/tcp
    a. vsftpd Compromised Source Packages Backdoor Vulnerability
7) High general/tcp
    a. Operating System (OS) End of Life (EOL) Detection
8) High 6200/tcp
    a. vsftpd Compromised Source Packages Backdoor Vulnerability
9) High 6697/tcp
    a. UnrealIRCd Authentication Spoong Vulnerability
10) High 8009/tcp
    a. Apache Tomcat AJP RCE Vulnerability (Ghostcat)
11) High 512/tcp
    a. The rexec service is running.
12) High 80/tcp
    a. TWiki XSS and Command Execution Vulnerabilities
    b. phpinfo() output Reporting
    c. Test HTTP dangerous methods
    d. PHP-CGI-based setups vulnerability when parsing query string parameters from php files.
13) High 2121/tcp
    a. FTP Brute Force Logins Reporting
14) Medium 5432/tcp
    a. SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection
    b. SSL/TLS: Server Certificate / Certificate in Chain with RSA keys less than 2048 bits
    c. SSL/TLS: Renegotiation DoS Vulnerability (CVE-2011-1473, CVE-2011-5094)
    d. SSL/TLS: Certificate Expired
    e. SSL/TLS: Report Weak Cipher Suites
    f. SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection
    g. SSL/TLS: Certificate Signed Using A Weak Signature Algorithm
    h. SSL/TLS: Di-e-Hellman Key Exchange Insu-cient DH Group Strength Vulnerability
15) Medium 5900/tcp
    a. VNC Server Unencrypted Data Transmission
16) Medium 21/tcp
    a. Anonymous FTP Login Reporting
    b. FTP Unencrypted Cleartext Login
17) Medium 25/tcp
    a. Multiple Vendors STARTTLS Implementation Plaintext Arbitrary Command Injection Vulnerability
    b. SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection
    c. SSL/TLS: Server Certificate / Certificate in Chain with RSA keys less than 2048 bits
    d. Check if Mailserver answer to VRFY and EXPN requests
    e. SSL/TLS: Renegotiation DoS Vulnerability (CVE-2011-1473, CVE-2011-5094)
    f. SSL/TLS: Certificate Expired
    g. SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection
    h. SSL/TLS: RSA Temporary Key Handling 'RSA_EXPORT' Downgrade Issue (FREAK)
    i. SSL/TLS: Certificate Signed Using A Weak Signature Algorithm
    j. SSL/TLS: Di-e-Hellman Key Exchange Insu-cient DH Group Strength Vulnerability
    k. Weak Host Key Algorithm(s) (SSH)

l. Weak Key Exchange (KEX) Algorithm(s) Supported (SSH)
m. Weak Encryption Algorithm(s) Supported (SSH)

18) Medium 80/tcp
    a. TWiki Cross-Site Request Forgery Vulnerability - Sep10
    b. jQuery < 1.9.0 XSS Vulnerability
    c. TWiki < 6.1.0 XSS Vulnerability
    d. TWiki Cross-Site Request Forgery Vulnerability
    e. HTTP Debugging Methods (TRACE/TRACK) Enabled
    f. awiki <= 20100125 Multiple LFI Vulnerabilities - Active Check
    g. /doc directory browsable
    h. QWikiwiki directory traversal vulnerability
    i. Cleartext Transmission of Sensitive Information via HTTP
    j. Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability
    k. phpMyAdmin 'error.php' Cross Site Scripting Vulnerability
    l. jQuery < 1.6.3 XSS Vulnerability
19) Medium 445/tcp
    a. Samba MS-RPC Remote Shell Command Execution Vulnerability - Active Check
20) Medium 2121/tcp
    a. FTP Unencrypted Cleartext Login

DMZ Findings

1) High 21/tcp
    a. vsftpd Compromised Source Packages Backdoor Vulnerability
2) High 8787/tcp
    a. Distributed Ruby (dRuby/DRb) Multiple Remote Code Execution Vulnerabilities
3) High 1524/tcp
    a. Possible Backdoor: Ingreslock
4) High 3632/tcp
    a. DistCC RCE Vulnerability (CVE-2004-2687)
5) High 512/tcp
    a. The rexec service is running
6) High 5432/tcp
    a. PostgreSQL weak password
    b. SSL/TLS: OpenSSL CCS Man in the Middle Security Bypass Vulnerability
7) High 6200/tcp
    a. vsftpd Compromised Source Packages Backdoor Vulnerability
8) High 8009/tcp
    a. Apache Tomcat AJP RCE Vulnerability (Ghostcat)
9) High 80/tcp
    a. TWiki XSS and Command Execution Vulnerabilities
    b. phpinfo() output Reporting
    c. Test HTTP dangerous methods
    d. PHP-CGI-based setups vulnerability when parsing query string parameters from php les.
10) High 5900/tcp
    a. VNC Brute Force Login

11) High 2121/tcp
    a. FTP Brute Force Logins Reporting
12) High 6697/tcp
    a. UnrealIRCd Authentication Spoong Vulnerability
13) High 1099/tcp
    a. Java RMI Server Insecure Default Conguration RCE Vulnerability
14) High general/tcp
    a. Operating System (OS) End of Life (EOL) Detection
15) Medium 21/tc
    a. Anonymous FTP Login Reporting
    b. FTP Unencrypted Cleartext Login
16) Medium 22/tcp
    a. Weak Host Key Algorithm(s) (SSH)
    b. Weak Key Exchange (KEX) Algorithm(s) Supported (SSH)
    c. Weak Encryption Algorithm(s) Supported (SSH)
17) Medium 5432/tcp
    a. SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection
    b. SSL/TLS: Server Certificate in Chain with RSA keys less than 2048 bits
    c. SSL/TLS: Report Weak Cipher Suites
    d. SSL/TLS: Certificate Expired
    e. SSL/TLS: Renegotiation DoS Vulnerability (CVE-2011-1473, CVE-2011-5094)
    f. SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection
    g. SSL/TLS: Di-e-Hellman Key Exchange Insu-cient DH Group Strength Vulnerability
    h. SSL/TLS: Certificate Signed Using A Weak Signature Algorithm
18) Medium 445/tcp
    a. Samba MS-RPC Remote Shell Command Execution Vulnerability - Active Check
19) Medium 25/tcp
    a. Multiple Vendors STARTTLS Implementation Plaintext Arbitrary Command Injection Vulnerability
    b. SSL/TLS: Deprecated SSLv2 and SSLv3 Protocol Detection
    c. SSL/TLS: Server Certificate in Chain with RSA keys less than 2048 bits
    d. SSL/TLS: Renegotiation DoS Vulnerability (CVE-2011-1473, CVE-2011-5094)
    e. SSL/TLS: Certificate Expired
    f. Check if Mailserver answer to VRFY and EXPN requests
    g. SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection
    h. SSL/TLS: RSA Temporary Key Handling 'RSA_EXPORT' Downgrade Issue (FREAK)
    i. SSL/TLS: Certificate Signed Using A Weak Signature Algorithm
    j. SSL/TLS: Di-e-Hellman Key Exchange Insu-cient DH Group Strength Vulnerability
20) Medium 80/tc
    a. TWiki Cross-Site Request Forgery Vulnerability - Sep10
    b. jQuery < 1.9.0 XSS Vulnerability
    c. TWiki < 6.1.0 XSS Vulnerability
    d. TWiki Cross-Site Request Forgery Vulnerability
    e. HTTP Debugging Methods (TRACE/TRACK) Enabled

f. /doc directory browsable
g. QWikiwiki directory traversal vulnerability
h. awiki <= 20100125 Multiple LFI Vulnerabilities - Active Check
i. Cleartext Transmission of Sensitive Information via HTTP
j. Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability
k. jQuery < 1.6.3 XSS Vulnerability
l. phpMyAdmin 'error.php' Cross Site Scripting Vulnerability

21) Medium 5900/tcp
    a. VNC Server Unencrypted Data Transmission
22) Medium 2121/tcp
    a. FTP Unencrypted Cleartext Login
    b.

External Findings

1) High 80/tcp
    a. TWiki XSS and Command Execution Vulnerabilities
    b. Test HTTP dangerous methods
    c. PHP-CGI-based setups vulnerability when parsing query string parameters from php les.
    d. phpinfo() output Reporting
2) High general/tcp
    a. Operating System (OS) End of Life (EOL) Detection
3) Medium 80/tcp
    a. TWiki Cross-Site Request Forgery Vulnerability - Sep10
    b. TWiki < 6.1.0 XSS Vulnerability
    c. jQuery < 1.9.0 XSS Vulnerability
    d. TWiki Cross-Site Request Forgery Vulnerability
    e. HTTP Debugging Methods (TRACE/TRACK) Enabled
    f. /doc directory browsable
    g. QWikiwiki directory traversal vulnerability
    h. awiki <= 20100125 Multiple LFI Vulnerabilities - Active Check
    i. Cleartext Transmission of Sensitive Information via HTTP
    j. jQuery < 1.6.3 XSS Vulnerability
    k. Apache HTTP Server 'httpOnly' Cookie Information Disclosure Vulnerability
    l. phpMyAdmin 'error.php' Cross Site Scripting Vulnerability

Zap Vulnerabilities Found

The ZAP scan was configured to target the web server which had the web address: “http://www.seclab.net”. ZAP scan organized the findings by risk and confidence. Two scans were done, one in the DMZ and one in the external or public facing network. The first scan results will cover the DMZ scan. There was a total of 26 findings and 6 of those were information. The number of findings can be found in Figure 3. The report will go through each finding starting from the highest risk findings.

		Confidence			
		High	Medium	Low	Total
Risk	High	1	2	0	3
	Medium	2	6	1	9
	Low	1	6	1	8
	Information	0	4	2	6
	Total	4	18	4	26

Figure 3: DMZ ZAP Findings

DMZ Findings

There was a total of 3 high findings.

1) Cross Site Scripting (DOM Based)
    a. CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
2) Cross Site Scripting (Reflected Based)
    a. CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
3) External Redirect
    a. CWE-601: URL Redirection to Untrusted Site ('Open Redirect')

There was a total number of 9 medium findings.

1) Content Security Policy (CSP) Header Not Set
    a. CWE-693: Protection Mechanism Failure
2) Hidden File Found
    a. CWE-538: Insertion of Sensitive Information into Externally-Accessible File or Directory
3) .htaccess Information Leak
    a. CWE-94: Improper Control of Generation of Code ('Code Injection')
4) Application Error Disclosure
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
5) Directory Browsing - Apache 2
    a. CWE-548: Exposure of Information Through Directory Listing
6) Missing Anti-clickjacking Header
    a. CWE-1021: Improper Restriction of Rendered UI Layers or Frames
7) Vulnerable JS Library
    a. CWE-829: Inclusion of Functionality from Untrusted Control Sphere
8) XSLT Injection
    a. CWE-91: XML Injection (aka Blind XPath Injection)
9) Absence of Anti-CSRF Tokens
    a. CWE-352: Cross-Site Request Forgery (CSRF)

There was a total number of 8 low findings.

1) Server Leaks Version Information via "Server" HTTP Response Header Field
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
2) Cookie No HttpOnly Flag
   a. CWE-1004: Sensitive Cookie Without 'HttpOnly' Flag
3) Cookie without SameSite Attribute
   a. CWE-1275: Sensitive Cookie with Improper SameSite Attribute
4) Information Disclosure - Debug Error Messages
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
5) Private IP Disclosure
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
6) Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
7) X-Content-Type-Options Header Missing
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
8) Timestamp Disclosure – Unix
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

There was a total number of 6 information findings.

1) Information Disclosure - Sensitive Information in URL
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
2) Information Disclosure - Suspicious Comments
   a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
3) Modern Web Application
4) User Agent Fuzzer
5) User Controllable Charset
   a. CWE-20: Improper Input Validation
6) User Controllable HTML Element Attribute (Potential XSS)
   a. CWE-20: Improper Input Validation

External Findings

The second scan results will cover the external VLAN scan. This was configured to target “www.seclab.net”. There was a total of 33 findings and 5 of those were information. The number of findings and the risks can be found in Figure 4. The report will go through each finding starting from the highest risk findings.

		Confidence			
		High	**Medium**	**Low**	**Total**
Risk	**High**	1	13	0	14
	Medium	1	4	1	6
	Low	1	6	1	8
	Information	0	3	2	5
	Total	3	26	4	33

Figure 4: External ZAP Findings

There was a total of 14 high findings.

1) Cross Site Scripting (DOM Based)
    a. CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
2) Cross Site Scripting (Persistent)
    a. CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
3) Cross Site Scripting (Reflected)
    a. CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
4) External Redirect
    a. CWE-601: URL Redirection to Untrusted Site ('Open Redirect')
5) Path Traversal
    a. CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
6) Remote Code Execution - CVE-2012-1823
    a. CWE-20: Improper Input Validation
7) SQL Injection
    a. CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
8) SQL Injection - Hypersonic SQL - Time Based
    a. CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
9) SQL Injection – MySQL
    a. CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
10) SQL Injection - Oracle - Time Based
    a. CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
11) SQL Injection - PostgreSQL - Time Based
    a. CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
12) SQL Injection – SQLite
    a. CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
13) Server Side Include
    a. CWE-97: Improper Neutralization of Server-Side Includes (SSI) Within a Web Page
14) Source Code Disclosure - CVE-2012-1823
    a. CWE-20: Improper Input Validation

There was a total number of 6 medium findings.

1) Content Security Policy (CSP) Header Not Set
    a. CWE-693: Protection Mechanism Failure
2) Application Error Disclosure
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
3) Directory Browsing - Apache 2
    a. CWE-548: Exposure of Information Through Directory Listing
4) Missing Anti-clickjacking Header
    a. CWE-1021: Improper Restriction of Rendered UI Layers or Frames
5) Vulnerable JS Library
    a. CWE-829: Inclusion of Functionality from Untrusted Control Sphere
6) Absence of Anti-CSRF Tokens
    a. CWE-352: Cross-Site Request Forgery (CSRF)

There was a total number of 8 low findings.

1) Server Leaks Version Information via "Server" HTTP Response Header Field
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
2) Cookie No HttpOnly Flag
    a. CWE-1004: Sensitive Cookie Without 'HttpOnly' Flag
3) Cookie without SameSite Attribute
    a. CWE-1275: Sensitive Cookie with Improper SameSite Attribute
4) Information Disclosure - Debug Error Messages
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
5) Private IP Disclosure
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
6) Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
7) X-Content-Type-Options Header Missing
    a. CWE-693: Protection Mechanism Failure
8) Timestamp Disclosure – Unix
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

There was a total number of 5 information findings.

1) Information Disclosure - Sensitive Information in URL
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
2) Information Disclosure - Suspicious Comments
    a. CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
3) Modern Web Application
4) User Controllable Charset
    a. CWE-20: Improper Input Validation
5) User Controllable HTML Element Attribute (Potential XSS)
    a. CWE-20: Improper Input Validation

Greenbone Recommended Mitigations

Internal Findings Remediation

1) High 5432/tcp
    a. Change the password as soon as possible.
    b. Updates are available. OpenSSL does not properly restrict processing of ChangeCipherSpec messages, which allows man-in-the-middle attackers to trigger use of a zero-length master key in certain OpenSSL-toOpenSSL communications, and consequently hijack sessions or obtain sensitive information, via a crafted TLS handshake, aka the 'CCS Injection' vulnerability.
2) High 3632/tcp
    a. Vendor updates are available.
3) High 5900/tcp
    a. Change the password to something hard to guess or enable password protection at all.
4) High 1524/tcp
    a. A whole cleanup of the infected system is recommended.
5) High 8787/tcp
    a. Administrators of environments that rely on Distributed Ruby should ensure that appropriate controls are in place. Code-level controls may include: Implementing taint on untrusted input, Setting $SAFE levels appropriately (>=2 is recommended if untrusted hosts are allowed to submit Ruby commands, and >=3 may be appropriate), Including drb/acl.rb to set ACLEntry to restrict access to trusted hosts
6) High 21/tcp
    a. The repaired package can be downloaded. Please validate the package with its signature.
7) High general/tcp
    a. Upgrade the OS on the remote host to a version which is still supported and receiving security updates by the vendor
8) High 6200/tcp
    a. The repaired package can be downloaded. Please validate the package with its signature
9) High 6697/tcp
    a. Upgrade to UnrealIRCd 3.2.10.7, or 4.0.6, or later.
10) High 8009/tcp
    a. Update Apache Tomcat to version 7.0.100, 8.5.51, 9.0.31 or later.
11) High 512/tcp
    a. Disable the rexec service and use alternatives like SSH instead
12) High 80/tcp
    a. Upgrade to version 4.2.4 or later
    b. Delete the listed les or restrict access to them.
    c. Use access restrictions to these dangerous HTTP methods or disable them completely.

d. PHP has released version 5.4.3 and 5.3.13 to address this vulnerability. PHP is recommending that users upgrade to the latest version of PHP.

13) High 2121/tcp
    a. Change the password as soon as possible
14) Medium 5432/tcp
    a. It is recommended to disable the deprecated SSLv2 and/or SSLv3 protocols in favor of the TLSv1.2+ protocols.
    b. Replace the certificate with a stronger key and reissue the certificate it signed
    c. Users should contact their vendors for specific patch information. A general solution is to remove/disable renegotiation capabilities altogether from/in the affected SSL/TLS service.
    d. Replace the SSL/TLS certificate by a new one
    e. The configuration of this services should be changed so that it does not accept the listed weak cipher suites anymore.
    f. It is recommended to disable the deprecated TLSv1.0 and/or TLSv1.1 protocols in favor of the TLSv1.2+ protocols.
    g. Servers that use SSL/TLS certificates signed with a weak SHA-1, MD5, MD4 or MD2 hashing algorithm will need to obtain new SHA-2 signed SSL/TLS certificate to avoid web browser SSL/TLS certificate warnings.
    h. Deploy (Ephemeral) Elliptic-Curve Di-e-Hellman (ECDHE) or use a 2048-bit or stronger Di-eHellman group. For Apache Web Servers: Beginning with version 2.4.7, mod_ssl will use DH parameters which include primes with lengths of more than 1024 bits
15) Medium 5900/tcp
    a. Run the session over an encrypted channel provided by IPsec [RFC4301] or SSH [RFC4254]. Some VNC server vendors are also providing more secure Security Types within their products.
16) Medium 21/tcp
    a. If you do not want to share les, you should disable anonymous logins.
    b. Enable FTPS or enforce the connection via the 'AUTH TLS' command.
17) Medium 25/tcp
    a. Updates are available.
    b. It is recommended to disable the deprecated SSLv2 and/or SSLv3 protocols in favor of the TLSv1.2+ protocols.
    c. Replace the certificate with a stronger key and reissue the certificates it signed.
    d. Disable VRFY and/or EXPN on your Mailserver. For postx add 'disable_vrfy_command=yes' in 'main.cf'. For Sendmail add the option 'O PrivacyOptions=goaway'. It is suggested that, if you really want to publish this type of information, you use a mechanism that legitimate users actually know about, such as Finger or HTTP.
    e. Users should contact their vendors for specific patch information. A general solution is to remove/disable renegotiation capabilities altogether from/in the affected SSL/TLS service.
    f. Replace the SSL/TLS certificate by a new one

g. It is recommended to disable the deprecated TLSv1.0 and/or TLSv1.1 protocols in favor of the TLSv1.2+ protocols
h. Remove support for 'RSA_EXPORT' cipher suites from the service. If running OpenSSL update to version 0.9.8zd or 1.0.0p or 1.0.1k or later
i. Servers that use SSL/TLS certificates signed with a weak SHA-1, MD5, MD4 or MD2 hashing algorithm will need to obtain new SHA-2 signed SSL/TLS certificates to avoid web browser SSL/TLS certificate warnings.
j. Deploy (Ephemeral) Elliptic-Curve Di-e-Hellman (ECDHE) or use a 2048-bit or stronger Di-eHellman group. For Apache Web Servers: Beginning with version 2.4.7, mod_ssl will use DH parameters which include primes with lengths of more than 1024 bits
k. Disable the reported weak host key algorithm(s).
l. Disable the reported weak KEX algorithm(s). 1024-bit MODP group / prime KEX algorithms: Alternatively use elliptic-curve Di-e-Hellmann in general, e.g. Curve 25519.
m. Disable the reported weak encryption algorithm(s).

18) Medium 80/tcp

a. Upgrade to TWiki version 4.3.2 or later.
b. Update to version 1.9.0 or later.
c. Update to version 6.1.0 or later.
d. Upgrade to version 4.3.1 or later.
e. Disable the TRACE and TRACK methods in your web server configuration.
f. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
g. Use access restrictions for the /doc directory. If you use Apache you might use this in your access.conf: <Directory /usr/doc> AllowOverride None order deny, allow deny from all allow from localhost </Directory>
h. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
i. Enforce the transmission of sensitive data via an encrypted SSL/TLS connection. Additionally make sure the host / application is redirecting all users to the secured SSL/TLS connection before allowing to input sensitive data into the mentioned functions.
j. Update to Apache HTTP Server version 2.2.22 or later.
k. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
l. Update to version 1.6.3 or later or apply the patch.

19) Medium 445/tcp

a. Updates are available.

20) Medium 2121/tcp
    a. Enable FTPS or enforce the connection via the 'AUTH TLS' command.

DMZ Findings Remediation

1) High 21/tcp
    a. The repaired package can be downloaded. Please validate the package with its signature.
2) High 8787/tcp
    a. Administrators of environments that rely on Distributed Ruby should ensure that appropriate controls are in place. Code-level controls may include: Implementing taint on untrusted input. Setting $SAFE levels appropriately (>=2 is recommended if untrusted hosts are allowed to submit Ruby commands, and >=3 may be appropriate). Including drb/acl.rb to set ACLEntry to restrict access to trusted hosts
3) High 1524/tcp
    a. A whole cleanup of the infected system is recommended.
4) High 3632/tcp
    a. Vendor updates are available.
5) High 512/tcp
    a. Disable the rexec service and use alternatives like SSH instead
6) High 5432/tcp
    a. Change the password as soon as possible.
    b. Updates are available.
7) High 6200/tcp
    a. The repaired package can be downloaded. Please validate the package with its signature.
8) High 8009/tcp
    a. Update Apache Tomcat to version 7.0.100, 8.5.51, 9.0.31 or later.
9) High 80/tcp
    a. Upgrade to version 4.2.4 or later.
    b. Delete the listed les or restrict access to them.
    c. Use access restrictions to these dangerous HTTP methods or disable them completely.
    d. PHP has released version 5.4.3 and 5.3.13 to address this vulnerability. PHP is recommending that users upgrade to the latest version of PHP
10) High 5900/tcp
    a. Change the password to something hard to guess
11) High 2121/tcp
    a. Change the password as soon as possible.
12) High 6697/tcp
    a. Upgrade to UnrealIRCd 3.2.10.7, or 4.0.6, or later.
13) High 1099/tcp
    a. Disable class-loading.

14) High general/tcp
    a. Upgrade the OS on the remote host to a version which is still supported and receiving security updates by the vendor
15) Medium 21/tc
    a. If you do not want to share les, you should disable anonymous logins.
    b. Enable FTPS or enforce the connection via the 'AUTH TLS' command.
16) Medium 22/tcp
    a. Disable the reported weak host key algorithm(s).
    b. Disable the reported weak KEX algorithm(s). 1024-bit MODP group / prime KEX algorithms: Alternatively use elliptic-curve Di-e-Hellmann in general, e.g. Curve 25519.
    c. Disable the reported weak encryption algorithm(s).
17) Medium 5432/tcp
    a. It is recommended to disable the deprecated SSLv2 and/or SSLv3 protocols in favor of the TLSv1.2+ protocols.
    b. Replace the certificate with a stronger key and reissue the certificates it signed
    c. The configuration of these services should be changed so that it does not accept the listed weak cipher suites anymore.
    d. Replace the SSL/TLS certificate by a new one
    e. Users should contact their vendors for specific patch information. A general solution is to remove/disable renegotiation capabilities altogether from/in the affected SSL/TLS service.
    f. It is recommended to disable the deprecated TLSv1.0 and/or TLSv1.1 protocols in favor of the TLSv1.2+ protocols
    g. Deploy (Ephemeral) Elliptic-Curve Di-e-Hellman (ECDHE) or use a 2048-bit or stronger Di-eHellman group
    h. Servers that use SSL/TLS certificates signed with a weak SHA-1, MD5, MD4 or MD2 hashing algorithm will need to obtain new SHA-2 signed SSL/TLS certificates to avoid web browser SSL/TLS certificate warnings
18) Medium 445/tcp
    a. Updates are available.
19) Medium 25/tcp
    a. Updates are available
    b. It is recommended to disable the deprecated SSLv2 and/or SSLv3 protocols in favor of the TLSv1.2+ protocols.
    c. Replace the certificate with a stronger key and reissue the certificates it signed.
    d. Users should contact their vendors for specific patch information. A general solution is to remove/disable renegotiation capabilities altogether from/in the affected SSL/TLS service.
    e. Replace the SSL/TLS certificate by a new one.
    f. Disable VRFY and/or EXPN on your Mailserver.
    g. It is recommended to disable the deprecated TLSv1.0 and/or TLSv1.1 protocols in favor of the TLSv1.2+ protocols.
    h. Remove support for 'RSA_EXPORT' cipher suites from the service.

i. Servers that use SSL/TLS certificates signed with a weak SHA-1, MD5, MD4 or MD2 hashing algorithm will need to obtain new SHA-2 signed SSL/TLS certificates to avoid web browser SSL/TLS certificate warnings.
j. Deploy (Ephemeral) Elliptic-Curve Di-e-Hellman (ECDHE) or use a 2048-bit or stronger Di-eHellman group. For Apache Web Servers: Beginning with version 2.4.7, mod_ssl will use DH parameters which include primes with lengths of more than 1024 bits.

20) Medium 80/tc
a. Upgrade to TWiki version 4.3.2 or later.
b. Update to version 1.9.0 or later
c. Update to version 6.1.0 or later
d. Upgrade to version 4.3.1 or later.
e. Disable the TRACE and TRACK methods in your web server configuration.
f. Use access restrictions for the /doc directory. If you use Apache you might use this in your access.conf: <Directory /usr/doc> AllowOverride None order deny, allow deny from all allow from localhost </Directory>
g. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
h. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
i. Enforce the transmission of sensitive data via an encrypted SSL/TLS connection. Additionally make sure the host / application is redirecting all users to the secured SSL/TLS connection before allowing to input sensitive data into the mentioned functions.
j. Update to Apache HTTP Server version 2.2.22 or later
k. Update to version 1.6.3 or later or apply the patch.
l. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.

21) Medium 5900/tcp
a. Run the session over an encrypted channel provided by IPsec [RFC4301] or SSH [RFC4254]. Some VNC server vendors are also providing more secure Security Types within their products.

22) Medium 2121/tcp
a. Enable FTPS or enforce the connection via the 'AUTH TLS' command.
b.

External Findings Remediation

1) High 80/tcp
    a. Upgrade to version 4.2.4 or later.
    b. Use access restrictions to these dangerous HTTP methods or disable them completely
    c. PHP has released version 5.4.3 and 5.3.13 to address this vulnerability. PHP is recommending that users upgrade to the latest version of PHP.
    d. Delete the listed les or restrict access to them.
2) High general/tcp
    a. Upgrade the OS on the remote host to a version which is still supported and receiving security updates by the vendor.
3) Medium 80/tcp
    a. Upgrade to TWiki version 4.3.2 or later
    b. Update to version 6.1.0 or later.
    c. Update to version 1.9.0 or later.
    d. Upgrade to version 4.3.1 or later.
    e. Disable the TRACE and TRACK methods in your web server configuration.
    f. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
    g. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
    h. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one.
    i. Enforce the transmission of sensitive data via an encrypted SSL/TLS connection. Additionally make sure the host / application is redirecting all users to the secured SSL/TLS connection before allowing to input sensitive data into the mentioned functions.
    j. Update to version 1.6.3 or later or apply the patch.
    k. Update to Apache HTTP Server version 2.2.22 or later
    l. No known solution was made available for at least one year since the disclosure of this vulnerability. Likely none will be provided anymore. General solution options are to upgrade to a newer release, disable respective features, remove the product or replace the product by another one

Zap Recommended Mitigations

DMZ Findings Remediations

Cross-site scripting attacks occur when an attacker can inject malicious code into the webpage or webserver, which is then executed by the victim's web browser. To prevent the cross-site scripting vulnerabilities (High findings 1-2), the webserver will perform the following:

- Make sure that all user input, including that entered through forms and URLs, is validated, and sanitized to remove any dangerous code. Input filters and regular expressions are two examples of server-side validation methods that can be used for this.
- Before it is displayed on the page, encode all user-generated material using encoding methods like HTML entity encoding or JavaScript encoding. This stops the content from being interpreted as code by the browser.
- Implement a Content Security Policy that specifies which sources are allowed to execute scripts on the page. This can prevent inline scripts and other sources of malicious code from executing.
- Enforce the same-origin policy, which limits scripts on a page to only interact with resources from the same origin (i.e., the same domain, port, and protocol).
- document.write() can be used to insert untrusted data into a page, leading to XSS vulnerabilities. Use safer alternatives such as document.createElement() and document.appendChild() instead.
- Keep your web server, web application, and all plugins and libraries up-to-date with the latest security patches and updates.

External redirect CWE-601 vulnerabilities occur when a web application allows user-controlled data to be used to construct a redirect URL to an external website. To prevent High finding 3 the webserver should:

- Input validation and sanitization
- Create a whitelist of allowed URLs and only allow redirects to those URLs. This can help prevent attackers from redirecting users to malicious websites.
- Use the HTTP 303 redirect method instead of the HTTP 302 method. The HTTP 303 method tells the browser to request the redirect URL using a GET request, which helps prevent attackers from embedding malicious code in the redirect URL.
- Avoid using user input in redirect URLs: Do not allow user input to be used in constructing redirect URLs. Instead, use pre-defined URLs or parameters to redirect users
- Implement a click-through warning: Before redirecting users to an external website, implement a click-through warning that alerts users that they are leaving your website and provides them with an option to cancel the redirect

Medium findings:

1) When the Content Security Policy (CSP) header is not set in a web application, it can leave the application vulnerable to various types of attacks. Remediate by adding CSP

header to the HTTP response which specifies sources of content such as scripts and images that are allowed to load.

2) Sensitive information in files can be remediated by making sure the directory listing is disabled on the webserver to stop attackers from seeing a list of files I the web root. Limit access permissions so only authorized personnel can view files. Avoid storing sensitive information that could be exposed in logs or other web outputs. Implement HTTPs to encrypt all web traffic.
3) Improper control of generation of code allowed code injections. Input validation and sanitization. Parameterize queries when constructing database queries. Use a secure programming language or framework. Limit the privileges of the user account that is used to execute code on the target system.
4) Information Exposure occurs when an application exposes sensitive information without proper authentication or authorization. Implement strong authentication and authorization mechanisms to ensure that only authorized users can access sensitive information or perform certain actions. Use SSL/TLS and HTTPS encryption to protect sensitive data in transit and at rest.
5) Exposure of Information Through Directory: Disable directory listing on the web server. This can be done by configuring the web server to prevent directory listings from being displayed to users. Use access control mechanisms such as file permissions, user groups, or firewall rules to restrict access to sensitive files and directories on the web server. Remove any sensitive information that may be exposed through error messages, logs, or other output. Encrypt traffic using HTTPS.
6) Improper Restriction of Rendered UI Layers or Frames: Implement CSP. Use the frame-ancestors directive in your CSP to restrict the domains that can embed your application in an iframe to stop clickjacking. Validate input and sanitize output.
7) Inclusion of Functionality from Untrusted Control Sphere: Use trusted sources for all third-party libraries and modules. Verify the integrity of all third-party code. Limit the access of third-party code. Input validation and sanitization.
8) Use XML libraries and frameworks that have built-in security features to prevent XML injection attacks. Validate and sanitize input. If XML documents are constructed based on database queries, use parameterized queries to prevent SQL injection attacks. Limit access to sensitive resources.
9) To prevent CSRF, use anti-CSRF tokens to prevent attackers from creating fake requests. Anti-CSRF tokens are unique tokens that are generated for each user session and are included in all form submissions. When a form is submitted, the server verifies that the token is valid before processing the request. Use same-origin policy to prevent attackers from accessing or manipulating user data from another domain. Same-origin policy is a security feature that prevents scripts from one domain from accessing resources from another domain. Use HTTP-only cookies to prevent attackers from stealing user session cookies. Implement access controls to prevent unauthorized access.

External Findings Remediations

Most of these remediations are the same as the section above DMZ Findings Remediation. The ones that have not been remediated will be mentioned below.

5) Path Traversal: Improper Limitation of a Pathname to a Restricted Directory occurs when an attacker is able to access files outside of the intended directory by manipulating a path or directory traversal. Implement input validation. Use file system APIs that restrict access to directories and prevent path traversal attacks. Use absolute paths to access files and directories instead of relative paths. This ensures that files and directories can only be accessed from the intended location. Limit access to sensitive directories with access controls.

6) Improper Input Validation: Implement input validation to ensure that user input is valid and meets the expected format. Validate all input from users and ensure that it meets the required format and length. Use secure input handling functions that prevent SQL injection, cross-site scripting (XSS), and other types of attacks. Use whitelisting to ensure that input data contains only valid characters. Enforce strict data typing to ensure that input data is of the expected data type.

7 – 12) Use parameterized queries or prepared statements to separate SQL commands from user input. Use input sanitization to remove any characters that may be used in SQL injection attacks. Use least privilege access to limit the access to databases. Be careful about the error messages returned by the application. Ensure that error messages do not reveal sensitive information that could be used by attackers.

13) Improper Neutralization of Server-Side Includes (SSI) within a Web Page, occurs when an application does not properly sanitize user input that is used in server-side includes (SSI) directives. Remediate by disabling it if not needed. Implement input validation. Use output encoding to ensure that any user input that is displayed in the web page is properly encoded.

Conclusion

In conclusion, completing a thorough cyber security assessment is essential to identifying weaknesses and potential security threats in the systems and apps used by an organization. In our case, we utilized Greenbone Vulnerability Management and ZAP Application Security tool to scan our segmented webserver network. The quantity of vulnerabilities can be considerably decreased by adhering to security best practices, including the use of strong passwords, frequent system/software updates, access control mechanisms, and network segmentation. This is because many of these remediations, once implemented, apply to multiple of the findings and greatly reduce the attack surface of the network. It is also important to have regular security audits and assessments to help in maintaining the high level of security hardness and make sure new vulnerabilities are quickly found and patched. To avoid unauthorized access, data breaches, and other cyberthreats that could cause serious financial and reputational harm, cybersecurity must be prioritized as a crucial component of any organization's operations. Organizations can reduce the risks associated with cyberattacks and safeguard sensitive data from potential breaches by following industry-standard security measures and adhering to cybersecurity best practices, like the ones mentioned in the remediation section.