April 28, 2023

Vulnerability Scan Report

prepared by

ACME MSSP INCORPORATED

Overview

1 Executive Summary

Vulnerability scans were conducted on selected servers, networks, websites, and applications. This report contains the discovered potential risks from these scans. Risks have been classified into categories according to the level of threat and degree of potential harm they may pose.

1.1 Total Risks

Below is the total number of risks found by severity. High risks are the most severe and should be evaluated first. An accepted risk is one which has been manually reviewed and classified as acceptable to not fix at this time, such as a false positive or an intentional part of the system's architecture.

High	Medium	Low	Accepted
0	14	17	2

1.2 Report Coverage

This report includes findings for **1 target** that were scanned. Each target is a single URL, IP address, or fully qualified domain name (FQDN).

Vulnerability Categories

13 Active Web Application Vulnerabilities	15 Passive Web Application Vulnerabilities	3 Network Vulnerabilities
0 SSL/TLS Security	2 Open TCP Ports	0 Open UDP Ports

2 Risks By Target

This section contains the vulnerability findings for each target that was scanned. Prioritize the most vulnerable assets first.

2.1 Targets Summary

The total number of risks found for each target, by severity.

Target	High	Medium	Low	Accepted
example.com	0	14	17	2

2.2 Target Breakdowns

The risks discovered for each target.

Target
example.com

Total Risks

0	14	17	2

Active Web Application Vulnerabilities	Threat Level	First Detected
Absence of Anti-CSRF Tokens	Medium	114 days ago
Missing Anti-clickjacking Header	Medium	74 days ago
CSP: Wildcard Directive	Medium	114 days ago
CSP: script-src unsafe-inline	Medium	114 days ago
CSP: style-src unsafe-inline	Medium	114 days ago
Information Disclosure - Debug Error Messages	Low	114 days ago
Application Error Disclosure	Low	114 days ago
Cross-Domain JavaScript Source File Inclusion	Low	114 days ago
X-Content-Type-Options Header Missing	Low	74 days ago
Cookie Without Secure Flag	Low	114 days ago
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	114 days ago
Cookie without SameSite Attribute	Low	114 days ago
CSP: Notices	Low	114 days ago

Passive Web Application Vulnerabilities	Threat Level	First Detected
Absence of Anti-CSRF Tokens	Medium	74 days ago
Cross-Domain Misconfiguration	Medium	74 days ago

Application Error Disclosure	Medium	74 days ago
Missing Anti-clickjacking Header	Medium	74 days ago
CSP: Wildcard Directive	Medium	74 days ago
CSP: script-src unsafe-inline	Medium	74 days ago
CSP: style-src unsafe-inline	Medium	74 days ago
X-Content-Type-Options Header Missing	Low	74 days ago
Cross-Domain JavaScript Source File Inclusion	Low	74 days ago
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	74 days ago
Cookie Without Secure Flag	Low	74 days ago
Cookie No HttpOnly Flag	Low	74 days ago
Information Disclosure - Debug Error Messages	Low	74 days ago
Cookie with SameSite Attribute None	Low	74 days ago
CSP: Notices	Low	74 days ago

Network Vulnerabilities	Threat Level	First Detected
SSL/TLS: Report Weak Cipher Suites cvss score: 5.0	Medium	74 days ago
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection cvss score: 4.3	Medium	74 days ago
TCP timestamps cvss score: 2.6	Low	142 days ago

Open TCP Ports	Threat Level	First Detected
Open TCP Port: 443	Accepted	74 days ago
Open TCP Port: 80	Accepted	74 days ago

3 Active Web Application Vulnerabilities

The OWASP ZAP active web application scan crawls the pages of a web application. It scans for all of the passive scan checks and additionally makes requests and submits forms to actively test an application for even more vulnerabilities. The active scan checks for vulnerabilities such as SQL injection, remote command execution, XSS, and more.

3.1 Total Risks

Total number of risks found by severity.

High	Medium	Low	Accepted
0	5	8	0

3.2 Risks Breakdown

Summary list of all detected risks.

Title	Threat Level	Open	Accepted
Absence of Anti-CSRF Tokens	Medium	1	0
Missing Anti-clickjacking Header	Medium	1	0
CSP: Wildcard Directive	Medium	1	0
CSP: script-src unsafe-inline	Medium	1	0
CSP: style-src unsafe-inline	Medium	1	0
Information Disclosure - Debug Error Messages	Low	1	0
Application Error Disclosure	Low	1	0
Cross-Domain JavaScript Source File Inclusion	Low	1	0
X-Content-Type-Options Header Missing	Low	1	0

Cookie Without Secure Flag	Low	1	0
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	1	0
Cookie without SameSite Attribute	Low	1	0
CSP: Notices	Low	1	0

3.3 Full Risk Details

Detailed information about each risk found by the scan.

Absence of Anti-CSRF Tokens

Medium

Description

No Anti-CSRF tokens were found in a HTML submission form.

A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf.

CSRF attacks are effective in a number of situations, including:

* The victim has an active session on the target site.

* The victim is authenticated via HTTP auth on the target site.

* The victim is on the same local network as the target site.

CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

Solution

Phase: Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, use anti-CSRF packages such as the OWASP CSRFGuard.

Phase: Implementation

Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script.

Phase: Architecture and Design

Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330).

Note that this can be bypassed using XSS.

Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation.

Note that this can be bypassed using XSS.

Use the ESAPI Session Management control.

This control includes a component for CSRF.

Do not use the GET method for any request that triggers a state change.

Phase: Implementation

Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

References

http://projects.webappsec.org/Cross-Site-Request-Forgery
http://cwe.mitre.org/data/definitions/352.html

Vulnerable Target	First Detected
example.com	114 days ago

Missing Anti-clickjacking Header

Medium

Description

The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'ClickJacking' attacks.

Solution

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.

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. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

References

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Vulnerable Target	First Detected
example.com	74 days ago

CSP: Wildcard Directive

Medium

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	114 days ago

CSP: script-src unsafe-inline

Medium

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	114 days ago

CSP: style-src unsafe-inline

Medium

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	114 days ago

Information Disclosure - Debug Error Messages

Low

Description

The response appeared to contain common error messages returned by platforms such as ASP.NET, and Web-servers such as IIS and Apache. You can configure the list of common debug messages.

Solution

Disable debugging messages before pushing to production.

Vulnerable Target	First Detected
example.com	114 days ago

Application Error Disclosure

Low

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.

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.

Vulnerable Target	First Detected
example.com	114 days ago

Cross-Domain JavaScript Source File Inclusion

Low

Description

The page includes one or more script files from a third-party domain.

Solution

Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application.

Vulnerable Target	First Detected
example.com	114 days ago

X-Content-Type-Options Header Missing

Low

Description

The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

Solution

Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

References

http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx
https://owasp.org/www-community/Security_Headers

Vulnerable Target	First Detected
example.com	74 days ago

Cookie Without Secure Flag

Low

Description

A cookie has been set without the secure flag, which means that the cookie can be accessed via unencrypted connections.

Solution

Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

References

https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/02-Testing_for_Cookies_Attributes.html

Vulnerable Target	First Detected
example.com	114 days ago

Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)

Low

Description

The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.

Solution

Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.

References

http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx
http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html

Vulnerable Target	First Detected
example.com	114 days ago

Cookie without SameSite Attribute

Low

Description

A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

Solution

Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

References

https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site

Vulnerable Target	First Detected
example.com	114 days ago

CSP: Notices

Low

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	114 days ago

4 Passive Web Application Vulnerabilities

The OWASP ZAP passive web application scan crawls the pages of a web application. It inspects the web pages as well as the requests and responses sent between the server. The passive scan checks for vulnerabilities such as cross-domain misconfigurations, insecure cookies, vulnerable js dependencies, and more.

4.1 Total Risks

Total number of risks found by severity.

High	Medium	Low	Accepted
0	7	8	0

4.2 Risks Breakdown

Summary list of all detected risks.

Title	Threat Level	Open	Accepted
Absence of Anti-CSRF Tokens	Medium	1	0
Cross-Domain Misconfiguration	Medium	1	0
Application Error Disclosure	Medium	1	0
Missing Anti-clickjacking Header	Medium	1	0
CSP: Wildcard Directive	Medium	1	0
CSP: script-src unsafe-inline	Medium	1	0
CSP: style-src unsafe-inline	Medium	1	0
X-Content-Type-Options Header Missing	Low	1	0
Cross-Domain JavaScript Source File Inclusion	Low	1	0

Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	Low	1	0
Cookie Without Secure Flag	Low	1	0
Cookie No HttpOnly Flag	Low	1	0
Information Disclosure - Debug Error Messages	Low	1	0
Cookie with SameSite Attribute None	Low	1	0
CSP: Notices	Low	1	0

4.3 Full Risk Details

Detailed information about each risk found by the scan.

Absence of Anti-CSRF Tokens

Medium

Description

No Anti-CSRF tokens were found in a HTML submission form.

A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf.

CSRF attacks are effective in a number of situations, including:

* The victim has an active session on the target site.

* The victim is authenticated via HTTP auth on the target site.

* The victim is on the same local network as the target site.

CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

Solution

Phase: Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, use anti-CSRF packages such as the OWASP CSRFGuard.

Phase: Implementation

Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script.

Phase: Architecture and Design

Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330).

Note that this can be bypassed using XSS.

Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation.

Note that this can be bypassed using XSS.

Use the ESAPI Session Management control.

This control includes a component for CSRF.

Do not use the GET method for any request that triggers a state change.

Phase: Implementation

Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.

References

http://projects.webappsec.org/Cross-Site-Request-Forgery
http://cwe.mitre.org/data/definitions/352.html

Vulnerable Target	First Detected
example.com	74 days ago

Cross-Domain Misconfiguration

Medium

Description

Web browser data loading may be possible, due to a Cross Origin Resource Sharing (CORS) misconfiguration on the web server

Solution

Ensure that sensitive data is not available in an unauthenticated manner (using IP address white-listing, for instance).

Configure the "Access-Control-Allow-Origin" HTTP header to a more restrictive set of domains, or remove all CORS headers entirely, to allow the web browser to enforce the Same Origin Policy (SOP) in a more restrictive manner.

References

https://vulncat.fortify.com/en/detail?id=desc.config.dotnet.html5_overly_permissive_cors_policy

Vulnerable Target	First Detected
example.com	74 days ago

Application Error Disclosure

Medium

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.

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.

Vulnerable Target	First Detected
example.com	74 days ago

Missing Anti-clickjacking Header

Medium

Description

The response does not include either Content-Security-Policy with 'frame-ancestors' directive or X-Frame-Options to protect against 'ClickJacking' attacks.

Solution

Modern Web browsers support the Content-Security-Policy and X-Frame-Options HTTP headers. Ensure one of them is set on all web pages returned by your site/app.

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. Alternatively consider implementing Content Security Policy's "frame-ancestors" directive.

References

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Vulnerable Target	First Detected
example.com	74 days ago

CSP: Wildcard Directive

Medium

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	74 days ago

CSP: script-src unsafe-inline

Medium

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	74 days ago

CSP: style-src unsafe-inline

Medium

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	74 days ago

X-Content-Type-Options Header Missing

Low

Description

The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.

Solution

Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages.

If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.

References

http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx
https://owasp.org/www-community/Security_Headers

Vulnerable Target	First Detected
example.com	74 days ago

Cross-Domain JavaScript Source File Inclusion

Low

Description

The page includes one or more script files from a third-party domain.

Solution

Ensure JavaScript source files are loaded from only trusted sources, and the sources can't be controlled by end users of the application.

Vulnerable Target	First Detected
example.com	74 days ago

Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)

Low

Description

The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.

Solution

Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.

References

http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx
http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html

Vulnerable Target	First Detected
example.com	74 days ago

Cookie Without Secure Flag

Low

Description

A cookie has been set without the secure flag, which means that the cookie can be accessed via unencrypted connections.

Solution

Whenever a cookie contains sensitive information or is a session token, then it should always be passed using an encrypted channel. Ensure that the secure flag is set for cookies containing such sensitive information.

References

https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/06-Session_Management_Testing/02-Testing_for_Cookies_Attributes.html

Vulnerable Target	First Detected
example.com	74 days ago

Cookie No HttpOnly Flag

Low

Description

A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.

Solution

Ensure that the HttpOnly flag is set for all cookies.

References

https://owasp.org/www-community/HttpOnly

Vulnerable Target	First Detected
example.com	74 days ago

Information Disclosure - Debug Error Messages

Low

Description

The response appeared to contain common error messages returned by platforms such as ASP.NET, and Web-servers such as IIS and Apache. You can configure the list of common debug messages.

Solution

Disable debugging messages before pushing to production.

Vulnerable Target	First Detected
example.com	74 days ago

Cookie with SameSite Attribute None

Low

Description

A cookie has been set with its SameSite attribute set to "none", which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

Solution

Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.

References

https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site

Vulnerable Target	First Detected
example.com	74 days ago

CSP: Notices

Low

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks. Including (but not limited to) Cross Site Scripting (XSS), and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

Solution

Ensure that your web server, application server, load balancer, etc. is properly configured to set the Content-Security-Policy header.

References

http://www.w3.org/TR/CSP2/
http://www.w3.org/TR/CSP/
http://caniuse.com/#search=content+security+policy
http://content-security-policy.com/
https://github.com/shapesecurity/salvation
https://developers.google.com/web/fundamentals/security/csp#policy_applies_to_a_wide_variety_of_resources

Vulnerable Target	First Detected
example.com	74 days ago

5 SSL/TLS Security

The SSLyze security scan checks for misconfigured SSL/TLS certificates, expired certificates, weak ciphers, and SSL/TLS vulnerabilities such as Heartbleed.

5.1 Total Risks

Total number of risks found by severity.

High	Medium	Low	Accepted
0	0	0	0

5.2 Risks Breakdown

Summary list of all detected risks.

Title	Threat Level	Open	Accepted
No risks detected			

6 Network Vulnerabilities

The OpenVAS network vulnerability scan tests servers and internet connected devices for over 50,000 vulnerabilities. OpenVAS uses the Common Vulnerability Scoring System (CVSS) to quantify the severity of findings. 0.0 is the lowest severity and 10.0 is the highest.

6.1 Total Risks

Total number of risks found by severity.

High	Medium	Low	Accepted
0	2	1	0

6.2 Risks Breakdown

Summary list of all detected risks.

Title	Threat Level	CVSS Score	Open	Accepted
SSL/TLS: Report Weak Cipher Suites	Medium	5.0	1	0
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	Medium	4.3	1	0
TCP timestamps	Low	2.6	1	0

6.3 Full Risk Details

Detailed information about each risk found by the scan.

SSL/TLS: Report Weak Cipher Suites

Medium
cvss score: 5.0

Description

This routine reports all Weak SSL/TLS cipher suites accepted by a service.

NOTE: No severity for SMTP services with 'Opportunistic TLS' and weak cipher suites on port 25/tcp is reported. If too strong cipher suites are configured for this service the alternative would be to fall back to an even more insecure cleartext communication.

These rules are applied for the evaluation of the cryptographic strength:

- RC4 is considered to be weak (CVE-2013-2566, CVE-2015-2808)

- Ciphers using 64 bit or less are considered to be vulnerable to brute force methods and therefore considered as weak (CVE-2015-4000)

- 1024 bit RSA authentication is considered to be insecure and therefore as weak

- Any cipher considered to be secure for only the next 10 years is considered as medium

- Any other cipher is considered as strong

Solution

The configuration of this services should be changed so that it does not accept the listed weak cipher suites anymore.

Please see the references for more resources supporting you with this task.

References

CVE-2013-2566
CVE-2015-2808
CVE-2015-4000
https://www.bsi.bund.de/SharedDocs/Warnmeldungen/DE/CB/warnmeldung_cb-k16-1465_update_6.html
https://bettercrypto.org/
https://mozilla.github.io/server-side-tls/ssl-config-generator/

Vulnerable Target	First Detected
example.com	74 days ago

SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection

Medium
cvss score: 4.3

Description

It was possible to detect the usage of the deprecated TLSv1.0 and/or TLSv1.1 protocol on this system.

The TLSv1.0 and TLSv1.1 protocols contain known cryptographic flaws like:

- CVE-2011-3389: Browser Exploit Against SSL/TLS (BEAST)

- CVE-2015-0204: Factoring Attack on RSA-EXPORT Keys Padding Oracle On Downgraded Legacy Encryption (FREAK)

An attacker might be able to use the known cryptographic flaws to eavesdrop the connection between clients and the service to get access to sensitive data transferred within the secured connection.

Furthermore newly uncovered vulnerabilities in this protocols won't receive security updates anymore.

Solution

It is recommended to disable the deprecated TLSv1.0 and/or TLSv1.1 protocols in favor of the TLSv1.2+ protocols. Please see the references for more information.

References

CVE-2011-3389
CVE-2015-0204
https://ssl-config.mozilla.org/
https://bettercrypto.org/
https://datatracker.ietf.org/doc/rfc8996/
https://vnhacker.blogspot.com/2011/09/beast.html
https://web.archive.org/web/20201108095603/https://censys.io/blog/freak
https://www.enisa.europa.eu/publications/algorithms-key-size-and-parameters-report-2014

Vulnerable Target	First Detected
example.com	74 days ago

TCP timestamps

Low
cvss score: 2.6

Description

The remote host implements TCP timestamps and therefore allows to compute the uptime.

The remote host implements TCP timestamps, as defined by RFC1323/RFC7323.

A side effect of this feature is that the uptime of the remote host can sometimes be computed.

Solution

To disable TCP timestamps on linux add the line 'net.ipv4.tcp_timestamps

References

http://www.ietf.org/rfc/rfc1323.txt
http://www.ietf.org/rfc/rfc7323.txt
https://web.archive.org/web/20151213072445/http://www.microsoft.com/en-us/download/details.aspx?id=9152

Vulnerable Target	First Detected
example.com	142 days ago

7 Open TCP Ports

The NMAP TCP port scan discovers open TCP ports with a complete scan of ports 0 to 65535.

7.1 Total Risks

Total number of risks found by severity.

High	Medium	Low	Accepted
0	0	0	2

100%

7.2 Risks Breakdown

Summary list of all detected risks.

Title	Threat Level	Open	Accepted
Open TCP Port: 443	Medium	0	1
Open TCP Port: 80	Medium	0	1

7.3 Full Risk Details

Detailed information about each risk found by the scan.

Open TCP Port: 443

Medium

Description

An open port may be an expected configuration. For example, web servers use port 80 to serve websites over http and port 443 to serve websites over https. For a list of commonly used ports see https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers.

An unexpected open port could give unintended access to applications, data, and private networks. Open ports can also be dangerous when expected services are out of date and exploited through security vulnerabilities.

Vulnerable Target	First Detected
example.com Accepted:	74 days ago

Open TCP Port: 80

Medium

Description

An open port may be an expected configuration. For example, web servers use port 80 to serve websites over http and port 443 to serve websites over https. For a list of commonly used ports see https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers.

An unexpected open port could give unintended access to applications, data, and private networks. Open ports can also be dangerous when expected services are out of date and exploited through security vulnerabilities.

Vulnerable Target	First Detected
example.com Accepted:	74 days ago

8 Open UDP Ports

The NMAP UDP port scan discovers open ports of common UDP services

8.1 Total Risks

Total number of risks found by severity.

High	Medium	Low	Accepted
0	0	0	0

8.2 Risks Breakdown

Summary list of all detected risks.

Title	Threat Level	Open	Accepted
No risks detected			

9 Glossary

Accepted Risk

An accepted risk is one which has been manually reviewed and classified as acceptable to not fix at this time, such as a false positive or an intentional part of the system's architecture.

Active Web Application Vulnerabilities

The OWASP ZAP active web application scan crawls the pages of a web application. It scans for all of the passive scan checks and additionally makes requests and submits forms to actively test an application for even more vulnerabilities. The active scan checks for vulnerabilities such as SQL injection, remote command execution, XSS, and more.

Fully Qualified Domain Name (FQDN)

A fully qualified domain name is a complete domain name for a specific website or service on the internet. This includes not only the website or service name, but also the top-level domain name, such as .com, .org, .net, etc. For example, 'www.example.com' is an FQDN.

Passive Web Application Vulnerabilities

The OWASP ZAP passive web application scan crawls the pages of a web application. It inspects the web pages as well as the requests and responses sent between the server. The passive scan checks for vulnerabilities such as cross-domain misconfigurations, insecure cookies, vulnerable js dependencies, and more.

Network Vulnerabilities

The OpenVAS network vulnerability scan tests servers and internet connected devices for over 50,000 vulnerabilities. OpenVAS uses the Common Vulnerability Scoring System (CVSS) to quantify the severity of findings. 0.0 is the lowest severity and 10.0 is the highest.

Open TCP Ports

The NMAP TCP port scan discovers open TCP ports with a complete scan of ports 0 to 65535.

Open UDP Ports

The NMAP UDP port scan discovers open ports of common UDP services

Risk

A risk is a finding from a vulnerability scan. Each risk is a potential security issue that needs review. Risks are assigned a threat level which represents the potential severity.

SSL/TLS Security

The SSLyze security scan checks for misconfigured SSL/TLS certificates, expired certificates, weak ciphers, and SSL/TLS vulnerabilities such as Heartbleed.

Target

A target represents target is a single URL, IP address, or fully qualified domain name (FQDN) that was scanned.

Threat Level

The threat level represents the estimated potential severity of a particular risk. Threat level is divided into 4 categories: High, Medium, Low and Accepted.