

January 8, 2026

# Vulnerability Scan Report

Prepared By

**HostedScan Security**



# Overview

<b>1 Executive Summary</b>	<b>3</b>
<b>2 Trends</b>	<b>4</b>
<b>3 Vulnerabilities By Target</b>	<b>5</b>
<b>4 Passive Web Application Vulnerabilities</b>	<b>8</b>
<b>5 Glossary</b>	<b>32</b>

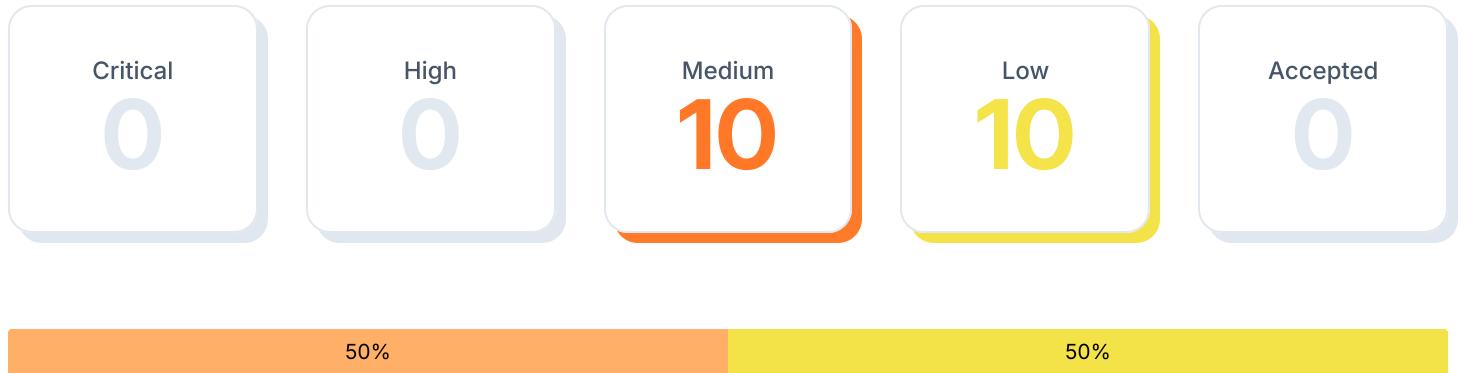


# 1 Executive Summary

Vulnerability scans were conducted on select servers, networks, websites, and applications. This report contains the discovered potential vulnerabilities from these scans. Vulnerabilities have been classified by severity. Higher severity indicates a greater risk of a data breach, loss of integrity, or availability of the targets.

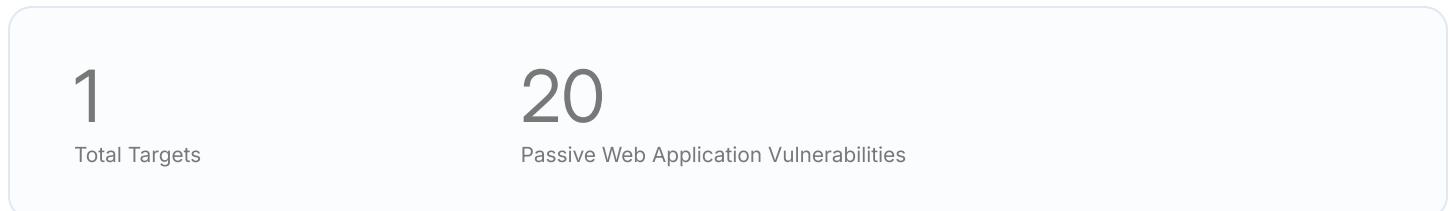
## 1.1 Total Vulnerabilities

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



## 1.2 Report Coverage

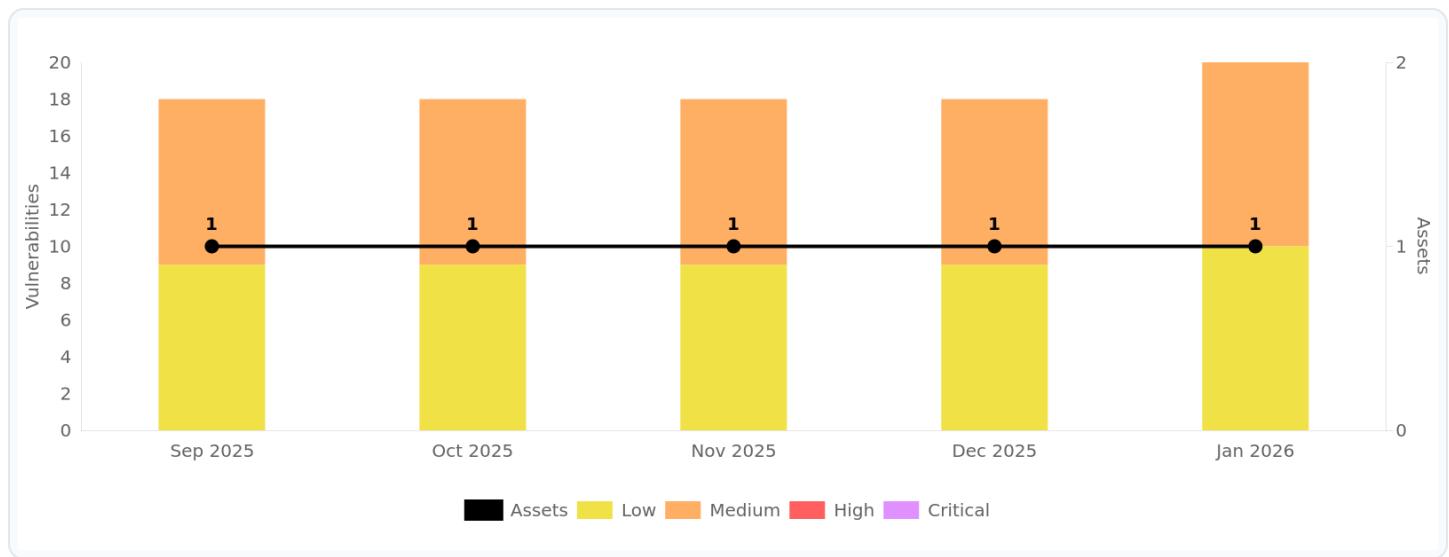
This report includes findings for **1 target** scanned. Each target is a single URL, IP address, or fully qualified domain name (FQDN).



## 2 Trends

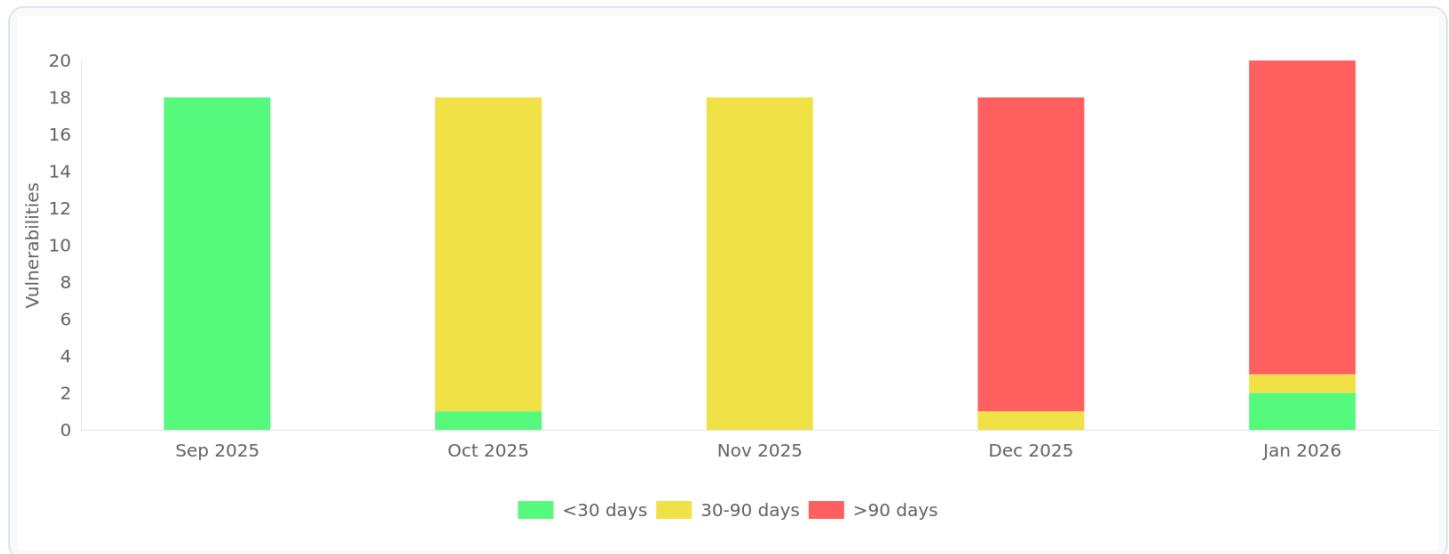
### 2.1 Open Risks

Total number of vulnerabilities grouped by severity level.



### 2.2 Exposure Window

Total number of unresolved vulnerabilities grouped by age (time since first detection).

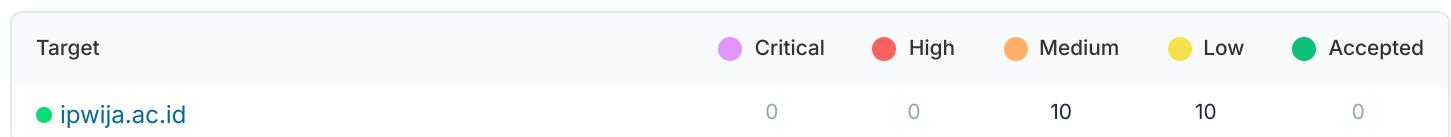


## 3 Vulnerabilities By Target

This section contains the vulnerability findings for each scanned target. Prioritization should be given to the targets with the highest severity vulnerabilities. However, it is important to take into account the purpose of each system and consider the potential impact a breach or an outage would have for the particular target.

### 3.1 Targets Summary (1)

The number of potential vulnerabilities found for each target by severity.

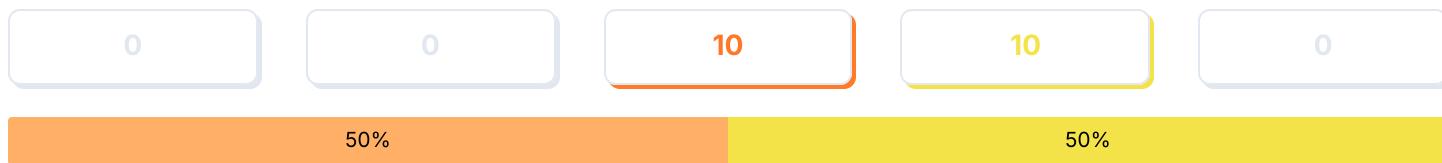


## 3.2 Target Breakdowns

Details for the potential vulnerabilities found for each target by scan type.



### Total Risks



Passive Web Application Vulnerabilities	Severity	First Detected	Last Detected
Absence of Anti-CSRF Tokens	● Medium	09/18/2025	01/08/2026
Content Security Policy (CSP) Header Not Set	● Medium	09/18/2025	01/08/2026
Weak Authentication Method	● Medium	09/18/2025	01/08/2026
CSP: Failure to Define Directive with No Fallback	● Medium	09/18/2025	01/08/2026
CSP: Wildcard Directive	● Medium	09/18/2025	01/08/2026
CSP: script-src unsafe-inline	● Medium	09/18/2025	01/08/2026
CSP: style-src unsafe-inline	● Medium	09/18/2025	01/08/2026
Missing Anti-clickjacking Header	● Medium	09/18/2025	01/08/2026
Vulnerable JS Library: bootstrap 3.2.0	● Medium	09/18/2025	01/08/2026
Sub Resource Integrity Attribute Missing	● Medium	01/08/2026	01/08/2026
Cookie No HttpOnly Flag	● Low	09/18/2025	01/08/2026
Cookie Without Secure Flag	● Low	09/18/2025	01/08/2026
Cross-Domain JavaScript Source File Inclusion	● Low	09/18/2025	01/08/2026
X-Content-Type-Options Header Missing	● Low	09/18/2025	01/08/2026
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	● Low	09/18/2025	01/08/2026
Cookie without SameSite Attribute	● Low	09/18/2025	01/08/2026

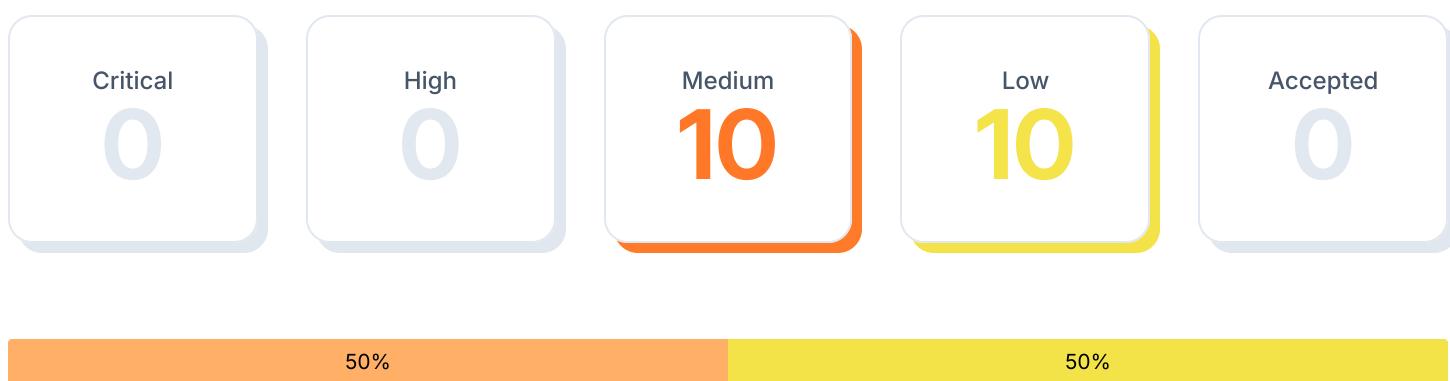
Strict-Transport-Security Header Not Set	<span style="color: yellow;">●</span> Low	09/18/2025	01/08/2026
Server Leaks Version Information via "Server" HTTP Response Header Field	<span style="color: yellow;">●</span> Low	10/18/2025	01/08/2026
In Page Banner Information Leak	<span style="color: yellow;">●</span> Low	01/08/2026	01/08/2026

## 4 Passive Web Application Vulnerabilities

The OWASP ZAP Passive Web Application scan crawls the pages of a website or web application. The passive scan inspects each page 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 Javascript dependencies, and more.

### 4.1 Total Vulnerabilities

Total number of vulnerabilities found by severity.



### 4.2 Vulnerabilities Breakdown

Summary list of all detected vulnerabilities.

Title	Severity	Open	Accepted
Absence of Anti-CSRF Tokens	Medium	1	0
Content Security Policy (CSP) Header Not Set	Medium	1	0
Weak Authentication Method	Medium	1	0
CSP: Failure to Define Directive with No Fallback	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
Missing Anti-clickjacking Header	Medium	1	0
Vulnerable JS Library: bootstrap 3.2.0	Medium	1	0

Sub Resource Integrity Attribute Missing	<span style="color: orange;">●</span> Medium	1	0
Cookie No HttpOnly Flag	<span style="color: yellow;">●</span> Low	1	0
Cookie Without Secure Flag	<span style="color: yellow;">●</span> Low	1	0
Cross-Domain JavaScript Source File Inclusion	<span style="color: yellow;">●</span> Low	1	0
X-Content-Type-Options Header Missing	<span style="color: yellow;">●</span> Low	1	0
Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)	<span style="color: yellow;">●</span> Low	1	0
Cookie without SameSite Attribute	<span style="color: yellow;">●</span> Low	1	0
Big Redirect Detected (Potential Sensitive Information Leak)	<span style="color: yellow;">●</span> Low	1	0
Strict-Transport-Security Header Not Set	<span style="color: yellow;">●</span> Low	1	0
Server Leaks Version Information via "Server" HTTP Response Header Field	<span style="color: yellow;">●</span> Low	1	0
In Page Banner Information Leak	<span style="color: yellow;">●</span> Low	1	0

## 4.3 Vulnerability Details

Detailed information about each potential vulnerability found by the scan.

---



# Absence of Anti-CSRF Tokens

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Medium	1 target	0 days ago

## 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 CSRGuard.

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.

### Instances (1 of 11)

uri: <https://pmb.ipwija.ac.id/>

method: GET

evidence: <form id="form\_filter" method="post">

otherinfo: No known Anti-CSRF token [anticsrf, CSRFToken, \_\_RequestVerificationToken, csrfmiddlewaretoken, authenticity\_token, OWASP\_CSRFTOKEN, anoncsrf, csrf\_token, \_csrf, \_csrfSecret, \_\_csrf\_magic, CSRF,\_token, \_csrf\_token, \_csrfToken, data[\_Token][key], \_wpnonce] was found in the following HTML form: [Form 1: "act" ].

### References

[https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site\\_Request\\_Forgery\\_Prevention\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html)

<https://cwe.mitre.org/data/definitions/352.html>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Content Security Policy (CSP) Header Not Set

**SEVERITY**

Medium

**AFFECTED TARGETS**

1 target

**LAST DETECTED**

0 days ago

## Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including 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 configured to set the Content-Security-Policy header.

## Instances (1 of 10)

uri: <https://pmb.ipwija.ac.id/>

method: GET

## References

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/CSP>

[https://cheatsheetseries.owasp.org/cheatsheets/Content\\_Security\\_Policy\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html)

<https://www.w3.org/TR/CSP/>

<https://w3c.github.io/webappsec-csp/>

<https://web.dev/articles/csp>

<https://caniuse.com/#feat=contentsecuritypolicy>

<https://content-security-policy.com/>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Weak Authentication Method

SEVERITY

Medium

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## Description

HTTP basic or digest authentication has been used over an unsecured connection. The credentials can be read and then reused by someone with access to the network.

## Solution

Protect the connection using HTTPS or use a stronger authentication mechanism.

## Instances (1 of 3)

uri: <http://repository.ipwija.ac.id/cgi/users/home>

method: GET

evidence: www-authenticate: Basic realm="UNIVERSITAS IPWIJA REPOSITORY"

## References

[https://cheatsheetseries.owasp.org/cheatsheets/Authentication\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html)

Vulnerable Target	First Detected	Last Detected
<a href="http://ipwija.ac.id">ipwija.ac.id</a>	09/18/2025	01/08/2026

# CSP: Failure to Define Directive with No Fallback

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Medium	1 target	0 days ago

## Description

The Content Security Policy fails to define one of the directives that has no fallback. Missing/excluding them is the same as allowing anything.

## Solution

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

## Instances (1 of 5)

uri: <https://ipwija.ac.id/>  
method: GET  
param: Content-Security-Policy  
evidence: upgrade-insecure-requests  
otherinfo: The directive(s): frame-ancestors, form-action is/are among the directives that do not fallback to default-src.

## References

<https://www.w3.org/TR/CSP/>  
<https://caniuse.com/#search=content+security+policy>  
<https://content-security-policy.com/>  
<https://github.com/HtmlUnit/htmlunit-csp>  
<https://web.dev/articles/csp#resource-options>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# CSP: Wildcard Directive

SEVERITY

Medium

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## 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.

## Instances (1 of 5)

uri: <https://ipwija.ac.id/>

method: GET

param: Content-Security-Policy

evidence: upgrade-insecure-requests

otherinfo: The following directives either allow wildcard sources (or ancestors), are not defined, or are overly broadly defined: script-src, style-src, img-src, connect-src, frame-src, font-src, media-src, object-src, manifest-src, worker-src

## References

<https://www.w3.org/TR/CSP/>

<https://caniuse.com/#search=content+security+policy>

<https://content-security-policy.com/>

<https://github.com/HtmlUnit/htmlunit-csp>

<https://web.dev/articles/csp#resource-options>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# CSP: script-src unsafe-inline

SEVERITY

Medium

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## 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.

## Instances (1 of 5)

uri: <https://ipwija.ac.id/>  
method: GET  
param: Content-Security-Policy  
evidence: upgrade-insecure-requests  
otherinfo: script-src includes unsafe-inline.

## References

<https://www.w3.org/TR/CSP/>  
<https://caniuse.com/#search=content+security+policy>  
<https://content-security-policy.com/>  
<https://github.com/HtmlUnit/htmlunit-csp>  
<https://web.dev/articles/csp#resource-options>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# CSP: style-src unsafe-inline

SEVERITY

Medium

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## 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.

## Instances (1 of 5)

uri: <https://ipwija.ac.id/>  
method: GET  
param: Content-Security-Policy  
evidence: upgrade-insecure-requests  
otherinfo: style-src includes unsafe-inline.

## References

<https://www.w3.org/TR/CSP/>  
<https://caniuse.com/#search=content+security+policy>  
<https://content-security-policy.com/>  
<https://github.com/HtmlUnit/htmlunit-csp>  
<https://web.dev/articles/csp#resource-options>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Missing Anti-clickjacking Header

SEVERITY

Medium

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## Description

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

## 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.

## Instances (1 of 20)

uri: <https://pmb.ipwija.ac.id/>

method: GET

param: x-frame-options

## References

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

Vulnerable Target	First Detected	Last Detected
<a href="https://pmb.ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Vulnerable JS Library: bootstrap 3.2.0

**SEVERITY**

Medium

**AFFECTED TARGETS**

1 target

**LAST DETECTED**

0 days ago

## Description

The identified library appears to be vulnerable.

## Solution

Upgrade to the latest version of the affected library.

### Instances (1 of 1)

uri: <https://ipwija.ac.id/wp-content/themes/eduma/assets/js/main.min.js?ver=5.6.6>

method: GET

evidence: \* Bootstrap v3.2.0

otherinfo: The identified library bootstrap, version 3.2.0 is vulnerable. CVE-2018-14041 CVE-2019-8331 CVE-2018-20677 CVE-2018-20676 CVE-

2018-14042 CVE-2016-10735 CVE-2024-6485 <https://nvd.nist.gov/vuln/detail/CVE-2024-6485> <https://github.com/twbs/bootstrap/issues/28236>

<https://www.herodevs.com/vulnerability-directory/cve-2024-6485> <https://github.com/advisories/GHSA-pj7m-g53m-7638>

<https://github.com/twbs/bootstrap/issues/20184> <https://github.com/advisories/GHSA-vxmc-5x29-h64v> <https://github.com/advisories/GHSA-ph58-4vrj-w6hr> <https://github.com/twbs/bootstrap> <https://github.com/twbs/bootstrap/issues/20631> <https://github.com/advisories/GHSA-4p24-vmcr-4gqj> <https://github.com/advisories/GHSA-9v3m-8fp8-mj99> <https://nvd.nist.gov/vuln/detail/CVE-2018-20676>

## References

[https://owasp.org/Top10/A06\\_2021-Vulnerable\\_and\\_Outdated\\_Components/](https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/)

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Sub Resource Integrity Attribute Missing

**SEVERITY**

Medium

**AFFECTED TARGETS**

1 target

**LAST DETECTED**

0 days ago

**Description**

The integrity attribute is missing on a script or link tag served by an external server. The integrity tag prevents an attacker who have gained access to this server from injecting a malicious content.

**Solution**

Provide a valid integrity attribute to the tag.

**Instances (1 of 17)**

uri: <https://pmb.ipwija.ac.id/>

method: GET

evidence: <link href="https://fonts.googleapis.com/css2?family=Material+Icons" rel="stylesheet" />

**References**

[https://developer.mozilla.org/en-US/docs/Web/Security/Subresource\\_Integrity](https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity)

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id">ipwija.ac.id</a>	01/08/2026	01/08/2026



# Cookie No HttpOnly Flag

**SEVERITY**

Low

**AFFECTED TARGETS**

1 target

**LAST DETECTED**

0 days ago

**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.

**Instances (1 of 1)**uri: <https://pmb.ipwija.ac.id/>

method: GET

param: SIAKAD\_CLOUD\_FRONT\_ACCESS

evidence: Set-Cookie: SIAKAD\_CLOUD\_FRONT\_ACCESS

**References**<https://owasp.org/www-community/HttpOnly>

Vulnerable Target	First Detected	Last Detected
<a href="#">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Cookie Without Secure Flag

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Low	1 target	0 days ago

## 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.

## Instances (1 of 1)

uri: <https://pmb.ipwija.ac.id/>  
method: GET  
param: SIAKAD\_CLOUD\_FRONT\_ACCESS  
evidence: Set-Cookie: SIAKAD\_CLOUD\_FRONT\_ACCESS

## 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](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	Last Detected
<a href="#">ipwija.ac.id</a>	09/18/2025	01/08/2026

# Cross-Domain JavaScript Source File Inclusion

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Low	1 target	0 days ago

## 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.

### Instances (1 of 7)

uri: <https://pmb.ipwija.ac.id/>  
method: GET  
param: <https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js>  
evidence: <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id">ipwija.ac.id</a>	09/18/2025	01/08/2026



# X-Content-Type-Options Header Missing

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Low	1 target	0 days ago

## 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.

## Instances (1 of 9)

uri: <https://pmb.ipwija.ac.id/>

method: GET

param: x-content-type-options

otherinfo: This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.

## References

[https://learn.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/gg622941\(v=vs.85\)](https://learn.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/compatibility/gg622941(v=vs.85))

[https://owasp.org/www-community/Security\\_Headers](https://owasp.org/www-community/Security_Headers)

Vulnerable Target	First Detected	Last Detected
<a href="https://ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



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

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Low	1 target	0 days ago

## 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.

## Instances (1 of 6)

uri: <https://lp2m.ipwija.ac.id/>

method: GET

evidence: X-Powered-By: PHP/8.2.28

## References

[https://owasp.org/www-project-web-security-testing-guide/v42/4-Web\\_Application\\_Security\\_Testing/01-Information\\_Gathering/08-Fingerprint\\_Web\\_Application\\_Framework](https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/01-Information_Gathering/08-Fingerprint_Web_Application_Framework)  
<https://www.troyhunt.com/shhh-dont-let-your-response-headers/>

Vulnerable Target	First Detected	Last Detected
<a href="#">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Cookie without SameSite Attribute

SEVERITY

Low

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## 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.

## Instances (1 of 2)

uri: <https://pmb.ipwija.ac.id/>

method: GET

param: SIAKAD\_CLOUD\_FRONT\_ACCESS

evidence: Set-Cookie: SIAKAD\_CLOUD\_FRONT\_ACCESS

## References

<https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-cookie-same-site>

Vulnerable Target	First Detected	Last Detected
<a href="#">ipwija.ac.id</a>	09/18/2025	01/08/2026

# Big Redirect Detected (Potential Sensitive Information Leak)

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Low	1 target	0 days ago

## Description

The server has responded with a redirect that seems to provide a large response. This may indicate that although the server sent a redirect it also responded with body content (which may include sensitive details, PII, etc.).

## Solution

Ensure that no sensitive information is leaked via redirect responses. Redirect responses should have almost no content.

### Instances (1 of 3)

uri: <http://ipwija.ac.id/informasi-beasiswa/>

method: GET

otherinfo: Location header URI length: 40 [<https://ipwija.ac.id/informasi-beasiswa/>]. Predicted response size: 340. Response Body Length: 795.

Vulnerable Target	First Detected	Last Detected
<a href="#">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Strict-Transport-Security Header Not Set

SEVERITY

Low

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## Description

HTTP Strict Transport Security (HSTS) is a web security policy mechanism whereby a web server declares that complying user agents (such as a web browser) are to interact with it using only secure HTTPS connections (i.e. HTTP layered over TLS/SSL). HSTS is an IETF standards track protocol and is specified in RFC 6797.

## Solution

Ensure that your web server, application server, load balancer, etc. is configured to enforce Strict-Transport-Security.

## Instances (1 of 8)

uri: <https://pmb.ipwija.ac.id/>

method: GET

## References

[https://cheatsheetseries.owasp.org/cheatsheets/HTTP\\_Strict\\_Transport\\_Security\\_Cheat\\_Sheet.html](https://cheatsheetseries.owasp.org/cheatsheets/HTTP_Strict_Transport_Security_Cheat_Sheet.html)

[https://owasp.org/www-community/Security\\_Headers](https://owasp.org/www-community/Security_Headers)

[https://en.wikipedia.org/wiki/HTTP\\_Strict\\_Transport\\_Security](https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security)

<https://caniuse.com/stricttransportsecurity>

<https://datatracker.ietf.org/doc/html/rfc6797>

Vulnerable Target	First Detected	Last Detected
<a href="https://pmb.ipwija.ac.id/">ipwija.ac.id</a>	09/18/2025	01/08/2026



# Server Leaks Version Information via "Server" HTTP Response Header Field

SEVERITY	AFFECTED TARGETS	LAST DETECTED
Low	1 target	0 days ago

## Description

The web/application server is leaking version information via the "Server" HTTP response header. Access to such information may facilitate attackers identifying other vulnerabilities your web/application server is subject to.

## Solution

Ensure that your web server, application server, load balancer, etc. is configured to suppress the "Server" header or provide generic details.

### Instances (1 of 5)

uri: <http://repository.ipwija.ac.id/>  
method: GET  
evidence: Apache/2.4.29 (Ubuntu)

## References

<https://httpd.apache.org/docs/current/mod/core.html#servertokens>  
[https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552\(v=pandp.10\)](https://learn.microsoft.com/en-us/previous-versions/msp-n-p/ff648552(v=pandp.10))  
<https://www.troyhunt.com/shhh-dont-let-your-response-headers/>

Vulnerable Target	First Detected	Last Detected
<a href="#">ipwija.ac.id</a>	10/18/2025	01/08/2026



# In Page Banner Information Leak

SEVERITY

Low

AFFECTED TARGETS

1 target

LAST DETECTED

0 days ago

## Description

The server returned a version banner string in the response content. Such information leaks may allow attackers to further target specific issues impacting the product and version in use.

## Solution

Configure the server to prevent such information leaks. For example:

Under Tomcat this is done via the "server" directive and implementation of custom error pages.

Under Apache this is done via the "ServerSignature" and "ServerTokens" directives.

## Instances (1 of 3)

uri: <http://repository.ipwija.ac.id/cgi/users/home>

method: GET

evidence: Apache/2.4.29

otherinfo: There is a chance that the highlight in the finding is on a value in the headers, versus the actual matched string in the response body.

## References

[https://owasp.org/www-project-web-security-testing-guide/v41/4-Web\\_Application\\_Security\\_Testing/08-Testing\\_for\\_Error\\_Handling/](https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/08-Testing_for_Error_Handling/)

Vulnerable Target	First Detected	Last Detected
<a href="http://ipwija.ac.id">ipwija.ac.id</a>	01/08/2026	01/08/2026

# 5 Glossary

## Accepted Vulnerability

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

## 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 website or web application. The passive scan inspects each page 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 Javascript dependencies, and more.

## Vulnerability

A weakness in the computational logic (e.g., code) found in software and hardware components that, when exploited, results in a negative impact to confidentiality, integrity, or availability. Mitigation of the vulnerabilities in this context typically involves coding changes, but could also include specification changes or even specification deprecations (e.g., removal of affected protocols or functionality in their entirety).

## Target

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

## Severity

Severity represents the estimated impact potential of a particular vulnerability. Severity is divided into 5 categories: Critical, High, Medium, Low and Accepted.

## CVSS Score

The CVSS 3.0 score is a global standard for evaluating vulnerabilities with a 0 to 10 scale. CVSS maps to threat levels:

0.1 - 3.9 = Low

4.0 - 6.9 = Medium

7.0 - 8.9 = High

9.0 - 10.0 = Critical

## EPSS Score

The EPSS score is the estimated probability that a given vulnerability will be exploited in the wild within the next 30 days, on a 0% to 100% scale.

This report was prepared using

## HostedScan Security ®

For more information, visit [hostedscan.com](https://hostedscan.com)

Founded in Seattle, Washington in 2019, HostedScan, LLC. is dedicated to making continuous vulnerability scanning and risk management much more easily accessible to more businesses.



HostedScan, LLC.

2212 Queen Anne Ave N  
Suite #521  
Seattle, WA 98109

Terms & Policies  
[hello@hostedscan.com](mailto:hello@hostedscan.com)