Acunetix
by Invicti

Comprehensive Report

Medium

Acunetix Threat Level 2

One or more medium-severity type vulnerabilities have been discovered by the scanner. You should investigate each of these vulnerabilities to ensure they will not escalate to more severe problems.

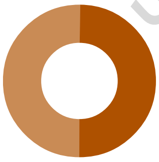
Scan Detail

Target	http://10.13.193.92:999
Scan Type	Full Scan
Start Time	Nov 21, 2025, 10:31:47 AM GMT+8
Scan Duration	1 minute
Requests	5097
Average Response Time	16ms
Maximum Response Time	21044ms
Application Build	v25.5.250613157
Authentication Profile	-

0	0	2	0	3
Critical	High	Medium	Low	Informational

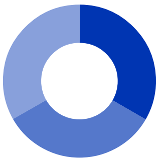
Severity	Vulnerabilities	Instances
Critical	0	0
High	0	0
Medium	2	2
Low	0	0
Informational	3	3
Total	5	5

Medium Severity



	Instances
Insecure HTTP Usage	1
SSL/TLS Not Implemented	1

Informational



	Instances
Content Security Policy (CSP) Not Impleme...	1
Permissions-Policy header not implemented	1
X-Content-Type-Options (XCTO) Not Imple...	1

Impacts

SEVERITY	IMPACT	
Medium	1	Insecure HTTP Usage
Medium	1	SSL/TLS Not Implemented
Informational	1	Content Security Policy (CSP) Not Implemented
Informational	1	Permissions-Policy header not implemented
Informational	1	X-Content-Type-Options (XCTO) Not Implemented

Insecure HTTP Usage

It was detected that your web application uses HTTP protocol, but doesn't automatically redirect users to HTTPS.

Impact

In some circumstances, it could be used for a man-in-the-middle (MitM) attack

http://10.13.193.92:999/

Request

```
GET / HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
Host: 10.13.193.92:999
Connection: Keep-alive
```

Recommendation

It's recommended to implement best practices of HTTP Redirection into your web application. Consult web references for more information

References

HTTP Redirections
https://infosec.mozilla.org/guidelines/web_security#http-redirections

SSL/TLS Not Implemented

This scan target was connected to over an unencrypted connection. A potential attacker can intercept and modify data sent and received from this site.

Impact

Possible information disclosure.

http://10.13.193.92:999/ Verified

Request

```
GET / HTTP/1.1
Referer: http://10.13.193.92:999/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate
```

```
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
Host: 10.13.193.92:999
Connection: Keep-alive
```

Recommendation

The site should send and receive data over a secure (HTTPS) connection.

References

[SSL/TLS Not Implemented | Invicti](https://www.invicti.com/web-vulnerability-scanner/vulnerabilities/ssltls-not-implemented/)
https://www.invicti.com/web-vulnerability-scanner/vulnerabilities/ssltls-not-implemented/

Content Security Policy (CSP) Not Implemented

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.

Content Security Policy (CSP) can be implemented by adding a **Content-Security-Policy** header. The value of this header is a string containing the policy directives describing your Content Security Policy. To implement CSP, you should define lists of allowed origins for the all of the types of resources that your site utilizes. For example, if you have a simple site that needs to load scripts, stylesheets, and images hosted locally, as well as from the jQuery library from their CDN, the CSP header could look like the following:

```
Content-Security-Policy:
default-src ’self’;
script-src ’self’ https://code.jquery.com;
```

It was detected that your web application doesn't implement Content Security Policy (CSP) as the CSP header is missing from the response. It's recommended to implement Content Security Policy (CSP) into your web application.

Impact

CSP can be used to prevent and/or mitigate attacks that involve content/code injection, such as cross-site scripting/XSS attacks, attacks that require embedding a malicious resource, attacks that involve malicious use of iframes, such as clickjacking attacks, and others.

http://10.13.193.92:999/

Paths without CSP header:

- http://10.13.193.92:999/
- http://10.13.193.92:999/clientaccesspolicy.xml
- http://10.13.193.92:999/login
- http://10.13.193.92:999/crossdomain.xml

Request

```
GET / HTTP/1.1
Referer: http://10.13.193.92:999/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
Host: 10.13.193.92:999
Connection: Keep-alive
```

Recommendation

It's recommended to implement Content Security Policy (CSP) into your web application. Configuring Content Security Policy involves adding the **Content-Security-Policy** HTTP header to a web page and giving it values to control resources the user agent is allowed to load for that page.

References

Content Security Policy (CSP)
https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

Implementing Content Security Policy
https://hacks.mozilla.org/2016/02/implementing-content-security-policy/

Permissions-Policy header not implemented

The Permissions-Policy header allows developers to selectively enable and disable use of various browser features and APIs.

Impact

http://10.13.193.92:999/

Locations without Permissions-Policy header:

- http://10.13.193.92:999/
- http://10.13.193.92:999/clientaccesspolicy.xml
- http://10.13.193.92:999/login
- http://10.13.193.92:999/logo/
- http://10.13.193.92:999/crossdomain.xml
- http://10.13.193.92:999/assets/

Request

```
GET / HTTP/1.1
Referer: http://10.13.193.92:999/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
Host: 10.13.193.92:999
```

```
Connection: Keep-alive
```

References

[Permissions-Policy / Feature-Policy (MDN)](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy)
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Feature-Policy

[Permissions Policy (W3C)](https://www.w3.org/TR/permissions-policy-1/)
https://www.w3.org/TR/permissions-policy-1/

X-Content-Type-Options (XCTO) Not Implemented

MIME type sniffing is a standard functionality in browsers to find an appropriate way to render data where the HTTP headers sent by the server are either inconclusive or missing.
This allows web browsers 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 intended content type.

X-Content-Type-Options (XCTO) is an HTTP header that can be used to prevent MIME type sniffing, which can help to mitigate certain types of attacks, including Cross Site Scripting (XSS). It also enables Cross-Origin Read Blocking (CORB) for sensitive resources, helping protect against Cross-Site Script Inclusion (XSSI) and side channel attacks.

Impact

XCTO header can be used as an additional layer of defense to prevent various attacks, such as Cross-Site Scripting (XSS), Cross-Site Script Inclusion (XSSI), side channel attacks, and others.

http://10.13.193.92:999/

Paths without X-Content-Type-Options header:

- http://10.13.193.92:999/
- http://10.13.193.92:999/assets/LoginView-BaTE-wlW.css
- http://10.13.193.92:999/clientaccesspolicy.xml
- http://10.13.193.92:999/assets/LoginView-SnWWjr6B.js
- http://10.13.193.92:999/login
- http://10.13.193.92:999/crossdomain.xml
- http://10.13.193.92:999/assets/_plugin-vue_export-helper-DlAUqK2U.js
- http://10.13.193.92:999/assets/index-51h8plvk.css
- http://10.13.193.92:999/assets/index-D7nFzLnk.js
- http://10.13.193.92:999/assets/validation-Df079fv5.js

Request

```
GET / HTTP/1.1
Referer: http://10.13.193.92:999/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
Host: 10.13.193.92:999
Connection: Keep-alive
```

Recommendation

Add the X-Content-Type-Options header with a value of "nosniff" to inform the browser to trust what the site has sent is the appropriate content-type, and to not attempt "sniffing" the real content-type.

References

X-Content-Type-Options header
https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/X-Content-Type-Options

Cross-Origin Read Blocking (CORB)
https://chromium.googlesource.com/chromium/src/+/HEAD/services/network/cross_origin_read_blocking_explainer.md

Coverage

- http://10.13.193.92:999
  - Inputs
    - GET /[*]/<s>, /<s>/[*]
    - GET Accept, Referer, User-Agent, X-Forwarded-For
  - assets
    - Inputs
      - GET Accept, Referer, User-Agent, X-Forwarded-For
    - _plugin-vue_export-helper-DlAUqK2U.js
    - index-51h8plvk.css
      - #fragments
        - elCarouselHorizontal
        - elCarouselVertical
    - index-D7nFzLnk.js
    - LoginView-BaTE-wlW.css
    - LoginView-SnWWjr6B.js
    - validation-Df079fv5.js
  - logo
    - Inputs
      - GET Accept, Referer, User-Agent, X-Forwarded-For
  - clientaccesspolicy.xml
    - Inputs
      - GET Accept, Referer, User-Agent, X-Forwarded-For
  - crossdomain.xml
    - Inputs
      - GET Accept, Referer, User-Agent, X-Forwarded-For
  - login
    - Inputs
      - GET Accept, Referer, User-Agent, X-Forwarded-For
  - robots.txt
    - Inputs
      - GET Accept, Referer, User-Agent, X-Forwarded-For