Acunetix by Invicti

Comprehensive Report

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	stream.ssbmultiservices.com
Scan Type	Full Scan
Start Time	Feb 15, 2024, 7:51:39 PM GMT+8
Scan Duration	2 minutes
Requests	6693
Average Response Time	31ms
Maximum Response Time	1291ms
Application Build	v23.7.230728157

0	1	1	4
High	Medium	Low	Informational

Severity	Vulnerabilities	Instances
High	0	0
Medium	1	1
Low	1	1
Informational	4	4
Total	6	6

Informational

		Instances
🟩	Content Security Policy (CSP) not implement…	1
🟩	No HTTP Redirection	1
🟩	Permissions-Policy header not implemented	1
🟩	Others	1

Low Severity

		Instances
🟦	Clickjacking: X-Frame-Options header	1

Medium Severity

		Instances
🟧	Unencrypted connection	1

Impacts

SEVERITY	IMPACT	
Medium	1	Unencrypted connection
Low	1	Clickjacking: X-Frame-Options header
Informational	1	Content Security Policy (CSP) not implemented
Informational	1	No HTTP Redirection
Informational	1	Permissions-Policy header not implemented
Informational	1	Subresource Integrity (SRI) not implemented

Unencrypted connection

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://stream.ssbmultiservices.com/ Verified

Request

```
GET / HTTP/1.1
Referer: http://stream.ssbmultiservices.com/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/114.0.0.0 Safari/537.36
Host: stream.ssbmultiservices.com
Connection: Keep-alive
```

Recommendation

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

Clickjacking: X-Frame-Options header

Clickjacking (User Interface redress attack, UI redress attack, UI redressing) is a malicious technique of tricking a Web user into clicking on something different from what the user perceives they are clicking on, thus potentially revealing confidential information or taking control of their computer while clicking on seemingly innocuous web pages.

The server did not return an **X-Frame-Options** header with the value DENY or SAMEORIGIN, which means that this website could be at risk of a clickjacking attack. The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page inside a frame or iframe. Sites can use this to avoid clickjacking attacks, by ensuring that their content is not embedded into untrusted sites.

Impact

The impact depends on the affected web application.

http://stream.ssbmultiservices.com/

Paths without secure XFO header:

- http://stream.ssbmultiservices.com/
- http://stream.ssbmultiservices.com/index.html
- http://stream.ssbmultiservices.com/mailman/archives/

Request

```
GET / HTTP/1.1
Referer: http://stream.ssbmultiservices.com/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/114.0.0.0 Safari/537.36
Host: stream.ssbmultiservices.com
Connection: Keep-alive
```

Recommendation

Configure your web server to include an X-Frame-Options header and a CSP header with frame-ancestors directive. Consult Web references for more information about the possible values for this header.

References

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

Clickjacking
https://en.wikipedia.org/wiki/Clickjacking

OWASP Clickjacking
https://cheatsheetseries.owasp.org/cheatsheets/Clickjacking_Defense_Cheat_Sheet.html

Frame Buster Buster
https://stackoverflow.com/questions/958997/frame-buster-buster-buster-code-needed

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://stream.ssbmultiservices.com/

Paths without CSP header:

- http://stream.ssbmultiservices.com/
- http://stream.ssbmultiservices.com/index.html
- http://stream.ssbmultiservices.com/mailman/archives/

Request

```
GET / HTTP/1.1
Referer: http://stream.ssbmultiservices.com/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
```

```
Chrome/114.0.0.0 Safari/537.36
Host: stream.ssbmultiservices.com
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/

No HTTP Redirection

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://stream.ssbmultiservices.com/

Request

```
GET / HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/114.0.0.0 Safari/537.36
Host: stream.ssbmultiservices.com
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

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://stream.ssbmultiservices.com/

Locations without Permissions-Policy header:

- http://stream.ssbmultiservices.com/
- http://stream.ssbmultiservices.com/index.html
- http://stream.ssbmultiservices.com/mailman/
- http://stream.ssbmultiservices.com/mailman/archives/

Request

```
GET / HTTP/1.1
Referer: http://stream.ssbmultiservices.com/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/114.0.0.0 Safari/537.36
Host: stream.ssbmultiservices.com
Connection: Keep-alive
```

References

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

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

Subresource Integrity (SRI) not implemented

Subresource Integrity (SRI) is a security feature that enables browsers to verify that third-party resources they fetch (for example, from a CDN) are delivered without unexpected manipulation. It works by allowing developers to provide a cryptographic hash that a fetched file must match.

Third-party resources (such as scripts and stylesheets) can be manipulated. An attacker that has access or has hacked the hosting CDN can manipulate or replace the files. SRI allows developers to specify a base64-encoded cryptographic hash of the resource to be loaded. The integrity attribute containing the hash is then added to the <script> HTML element tag. The integrity string consists of a base64-encoded hash, followed by a prefix that depends on the hash algorithm. This prefix can either be sha256, sha384 or sha512.

The script loaded from the external URL specified in the Details section doesn't implement Subresource Integrity (SRI). It's recommended to implement Subresource Integrity (SRI) for all the scripts loaded from external hosts.

Impact

An attacker that has access or has hacked the hosting CDN can manipulate or replace the files.

http://stream.ssbmultiservices.com/

Pages where SRI is not implemented:

- http://stream.ssbmultiservices.com/
  Script SRC: **https://vjs.zencdn.net/7.15.4/video.min.js**

- http://stream.ssbmultiservices.com/
  Script SRC: **https://cdnjs.cloudflare.com/ajax/libs/videojs-flash/2.1.3/videojs-flash.min.js**

Request

```
GET / HTTP/1.1
Referer: http://stream.ssbmultiservices.com/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,br
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/114.0.0.0 Safari/537.36
Host: stream.ssbmultiservices.com
Connection: Keep-alive
```

Recommendation

Use the SRI Hash Generator link (from the References section) to generate a <script> element that implements Subresource Integrity (SRI).

For example, you can use the following <script> element to tell a browser that before executing the https://example.com/example-framework.js script, the browser must first compare the script to the expected hash, and verify that there's a match.

```
<script src="https://example.com/example-framework.js"
integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"
crossorigin="anonymous"></script>
```

References

Subresource Integrity
https://developer.mozilla.org/en-US/docs/Web/Security/Subresource_Integrity

SRI Hash Generator
https://www.srihash.org/

Coverage

- http://stream.ssbmultiservices.com
  - cgi-sys
  - mailman
    - archives
  - index.html