



Web Application Security Assessment Report – Grey/Black Box

For the attention of:

[Client SPOC Name]

[SPOC Designation]

Assessment Start Date:	XX/XX/202X
Assessment End Date:	XX/XX/202X
Report Issue Date:	XX/XX/202X
Document Version:	1.0
Score Type:	The vulnerability score of each issue in this report has been determined and ranked using the KRYLO scoring scheme.

Contents

1	Your Assessment	3
1.1	Description	3
1.2	Scope and Objectives	3
1.3	Constraints	3
2	Executive Summary	4
3	Technical Summary	5
3.1	Risk Summary Graph	5
3.2	Summary of Vulnerabilities	6
4	Critical Risk Findings	7
4.1	Telerik Remote Code Execution (CVE-2017-9248)	7
5	Medium Risks	10
5.1	Cross-Site Scripting (XSS)	10
5.2	Username Enumeration	14
5.3	Log.txt File Accessible	20
5.4	Cross-Origin Resource Sharing (CORS) Misconfiguration	22
6	Informational Risks.....	24
6.1	Missing HTTP Security Headers	24
6.2	Verbose Response Headers	29
6.3	Unencrypted Communication	35
7	Document Control.....	38

1 Your Assessment

1.1 Description

Your security assessment was carried out by Team Krylo between XX/XX/202X and XX/XX/202X. Throughout this engagement, the team made every effort to offer as much information as possible while adhering to the objectives, scope, and constraints outlined in the following sub-sections.

1.2 Scope and Objectives

The following URL/URLs were part of scope for this engagement:

- <https://abc.com>

The following User accounts were provided:

- User Role – ABC
- Admin Role – XYZ

1.3 Constraints

This engagement examined a snapshot in time of the URLs under consideration. Changes to the underlying configuration may result in the creation of new issues or a degraded security posture. Every day, new vulnerabilities and attack paths are identified, reinforcing the necessity for more security testing. Penetration Testing is a security standard that represents both Krylo's security assessment methodology and publicly known attack tactics at the time of engagement. Because project scope and time limits do not limit real-world attackers, more security flaws that could not reasonably be found during the engagement may exist and be exploited in the future.

2 Executive Summary

Krylo conducted a security assessment against <Client>'s web application as defined in Section 1.2. The testing utilised a grey-box approach where in the web application was tested from the perspective of an external threat actor and also from an authenticated user.

The web application assessment resulted in 8 vulnerabilities comprising of 1 Critical, 4 Medium and 3 Informational findings. The critical finding enabled a malicious user to upload files on the server which can lead to execution of commands remotely.

The medium findings comprised of injecting malicious JavaScript Code, enumerating valid user accounts, accessing server log file and sharing sensitive information to non-trusted malicious web applications.

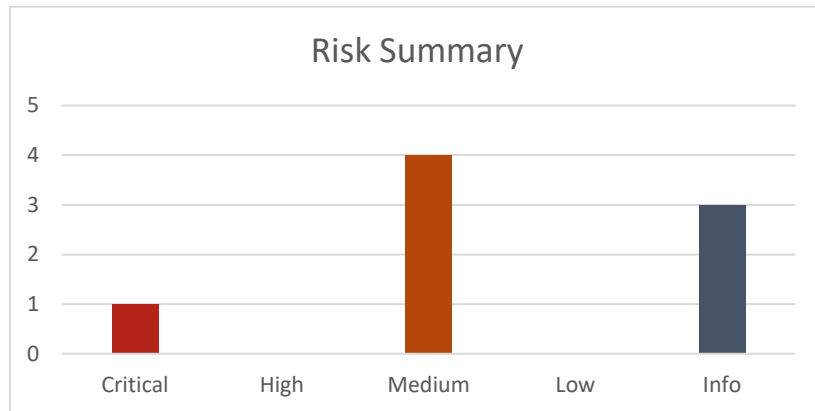
The informational findings are not considered an immediate threat but are considered security misconfigurations.

It is advised that <Client>, take adequate remediation measures against the Critical and Medium findings.

3 Technical Summary

3.1 Risk Summary Graph

The following graph represents the distribution of the risks identified by vulnerability rating.



3.2 Summary of Vulnerabilities

S. No.	Description	Remediation	Risk
1	Telerik Remote Code Execution (CVE-2017-9248)	Upgrade Telerik UI to latest version.	Critical
2	Cross-Site Scripting (XSS)	Sanitise all user inputs.	Medium
3	Username Enumeration	Install WP-Hardening plugin.	Medium
4	Log.txt File Accessible	Restrict access to file.	Medium
5	Cross-Origin Resource Sharing (CORS) Misconfiguration	Allow restricted domains in the Access-Control-Allow-Origin header.	Medium
6	Missing HTTP Security Headers	Implement appropriate Security Headers.	Informational
7	Verbose Response Headers	Remove software and versions from the web server's response headers.	Informational
8	Unencrypted Communication	Enforce communication via HTTPS.	Informational

4 Critical Risk Findings

These are the concerns that Krylo has recognized as posing a significant risk to [Client]. The consequences of an attacker exploiting these vulnerabilities would be disruptive and would drastically damage typical corporate operations. It is best to assess and prepare to resolve these concerns as soon as feasible. Krylo has supplied thorough details on how to duplicate each finding and the possible consequences of exploitation.

4.1 Telerik Remote Code Execution (CVE-2017-9248)

Risk: **Critical**

CVSS Score: 9.8

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Description: The identified server was running a version of the Progress Telerik UI that was affected by Remote Code Execution.

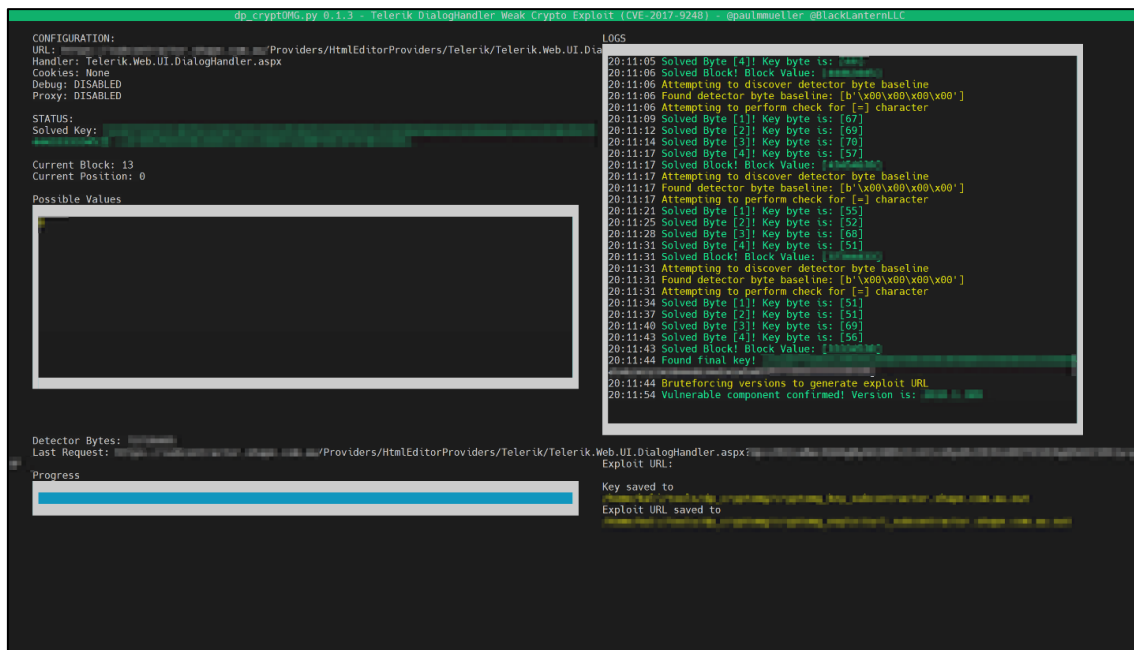
Impact: Telerik has a known cryptographic weakness in older versions of the Dialog Handler which when exploited can be used to brute force the machine key and gain access to the DNN file manager. The file manager allows for ASPX shell upload and RCE.

Proof of Concept: There was 1 instance of this issue identified at the following locations:

URL(s)

<https://subdomain.acme.corp.com/Providers/HtmlEditorProviders/Telerik/Telerik.Web.UI.DialogHandler.aspx>

Step 1: The team brute forced the machine key as shown below:



```
dp_crypto.py 0.1.3 - Telerik DialogHandler Weak Crypto Exploit (CVE-2017-9248) - @paulmueller_@blacklanternllc

CONFIGURATION:
URL: https://subdomain.acme.corp.com/Providers/HtmlEditorProviders/Telerik/Telerik.Web.UI.DialogHandler.aspx
Handler: Telerik.Web.UI.DialogHandler.aspx
Cookies: None
Debug: DISABLED
Proxy: DISABLED

STATUS:
Solved Key: [REDACTED]

Current Block: 13
Current Position: 0

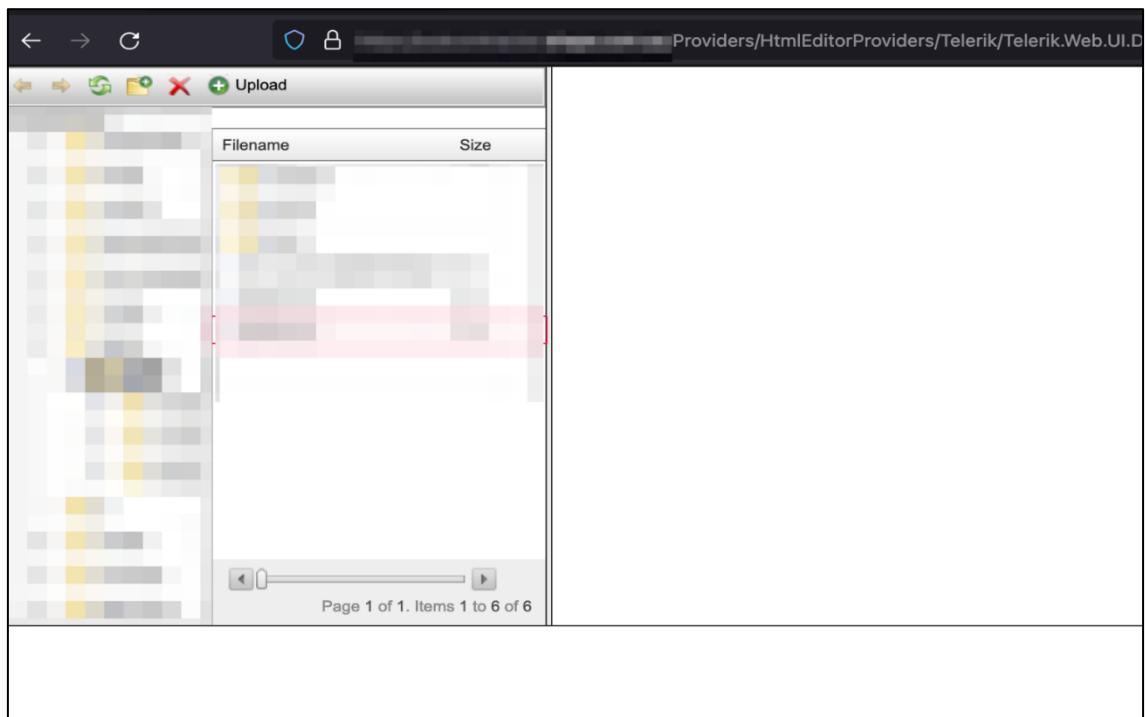
Possible Values
[Progress Bar]

Detector Bytes: [REDACTED]
Last Request: https://subdomain.acme.corp.com/Providers/HtmlEditorProviders/Telerik/Telerik.Web.UI.DialogHandler.aspx
Exploit URL: [REDACTED]

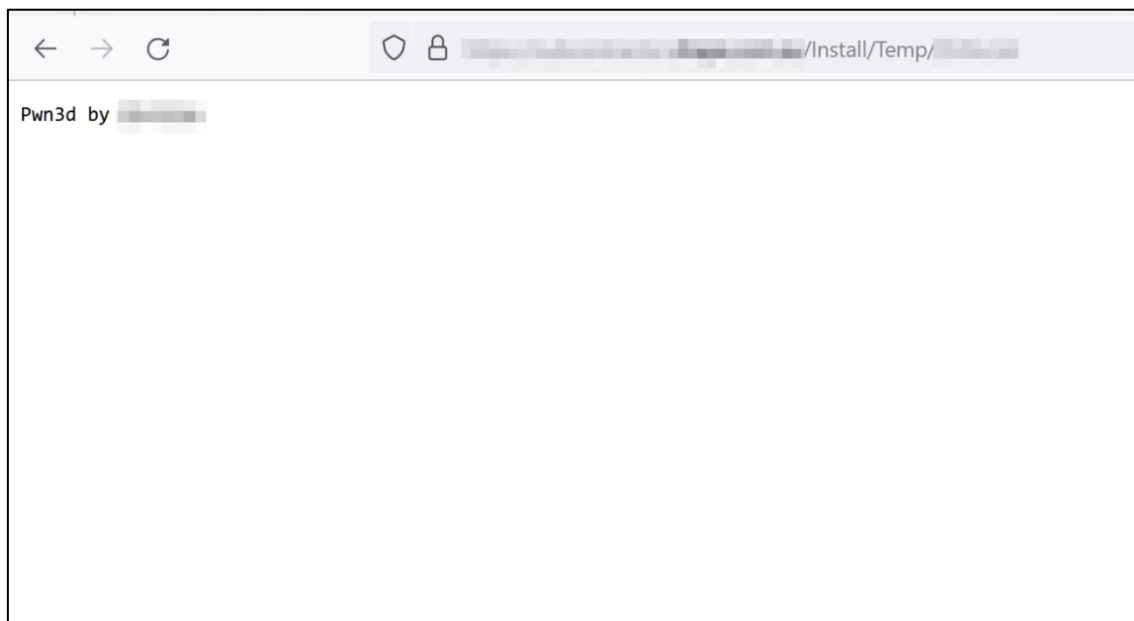
Key saved to [REDACTED]
Exploit URL saved to [REDACTED]

LOGS
20:11:05 Solved Byte [4]! Key byte is: [REDACTED]
20:11:06 Solved Block! Block Value: [REDACTED]
20:11:06 Attempting to discover detector byte baseline
20:11:06 Found detector byte baseline: [b'\x00\x00\x00\x00']
20:11:06 Attempting to perform check for [=] character
20:11:09 Solved Byte [1]! Key byte is: [67]
20:11:12 Solved Byte [2]! Key byte is: [69]
20:11:14 Solved Byte [3]! Key byte is: [76]
20:11:17 Solved Byte [4]! Key byte is: [57]
20:11:17 Solved Block! Block Value: [REDACTED]
20:11:17 Attempting to discover detector byte baseline
20:11:17 Found detector byte baseline: [b'\x00\x00\x00\x00']
20:11:17 Attempting to perform check for [=] character
20:11:21 Solved Byte [1]! Key byte is: [55]
20:11:25 Solved Byte [2]! Key byte is: [52]
20:11:28 Solved Byte [3]! Key byte is: [68]
20:11:31 Solved Byte [4]! Key byte is: [51]
20:11:31 Solved Block! Block Value: [REDACTED]
20:11:31 Attempting to discover detector byte baseline
20:11:31 Found detector byte baseline: [b'\x00\x00\x00\x00']
20:11:31 Attempting to perform check for [=] character
20:11:34 Solved Byte [1]! Key byte is: [51]
20:11:37 Solved Byte [2]! Key byte is: [51]
20:11:40 Solved Byte [3]! Key byte is: [69]
20:11:43 Solved Byte [4]! Key byte is: [56]
20:11:43 Solved Block! Block Value: [REDACTED]
20:11:44 Found final key!
20:11:44 Brute forcing versions to generate exploit URL
20:11:54 Vulnerable component confirmed! Version is: [REDACTED]
```

Step 2: Using the key, gained access to DNN file manager and uploaded an acme.txt in the /Install/Temp/ directory as shown below:



The following screenshot shows that the team was able to access the file from browser which were uploaded.



Recommendation: We recommend that the server must be upgraded to the latest vendor supported version of Progress Telerik UI.

Reference(s): <https://nvd.nist.gov/vuln/detail/CVE-2017-9248>

5 Medium Risks

These are the concerns Krylo has recognized as posing a moderate but significant danger to [Client]. If an attacker exploited these vulnerabilities, the impact would be disruptive but unlikely to seriously interrupt typical corporate operations. It is prudent to examine and prepare for resolution of these difficulties, but only after more serious hazards have been handled. Krylo has supplied thorough details on how to duplicate each and the possible consequences of exploitation.

5.1 Cross-Site Scripting (XSS)

Risk: Medium

CVSS Score: 6.5

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N

Description: The web applications were found to be vulnerable to Cross-Site Scripting (XSS). XSS occurs where user supplied input is stored and returned every time a user visited a vulnerable page, it would be known as Stored Cross-Site Scripting.

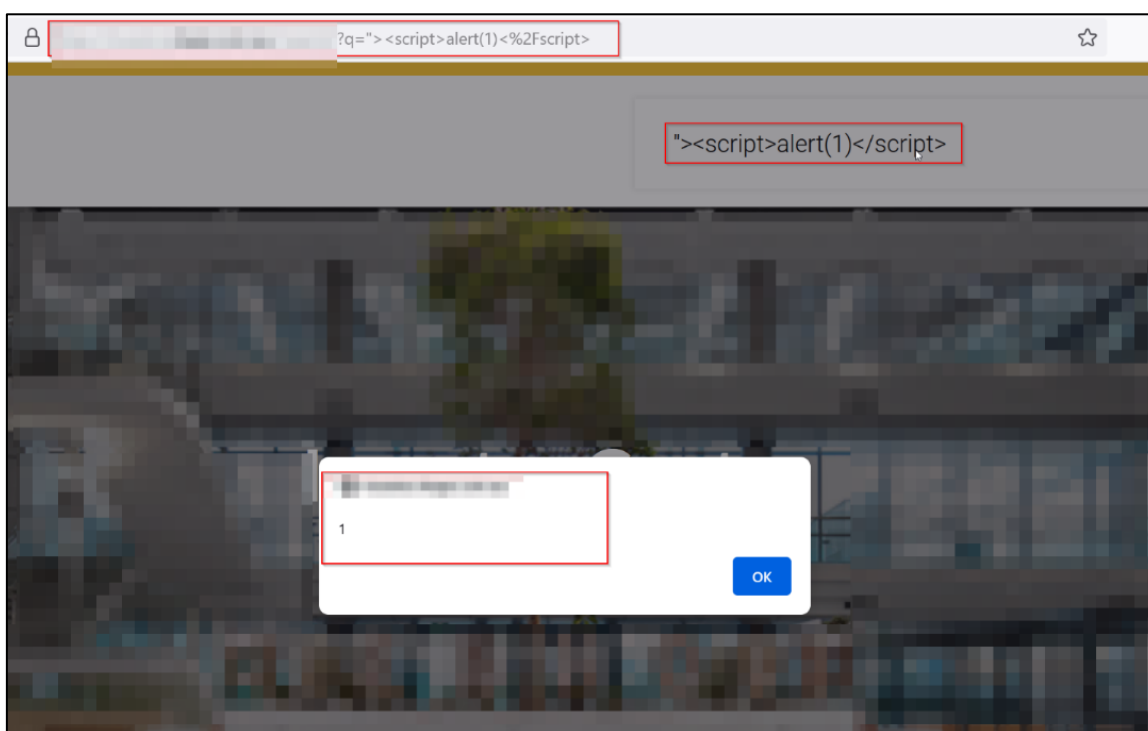
Impact: Where user input is insecurely included, it is generally possible to include malicious content such as HTML or JavaScript. HTML can be used to virtually deface the application, capturing user credentials through a fake login screen, or distributing political or offensive messages. JavaScript can be used for a range of functions, such as theft of confidential data, theft of user credentials, or the distribution of malicious software.

Instance(s): There were 3 instances of this issue identified at the following locations:

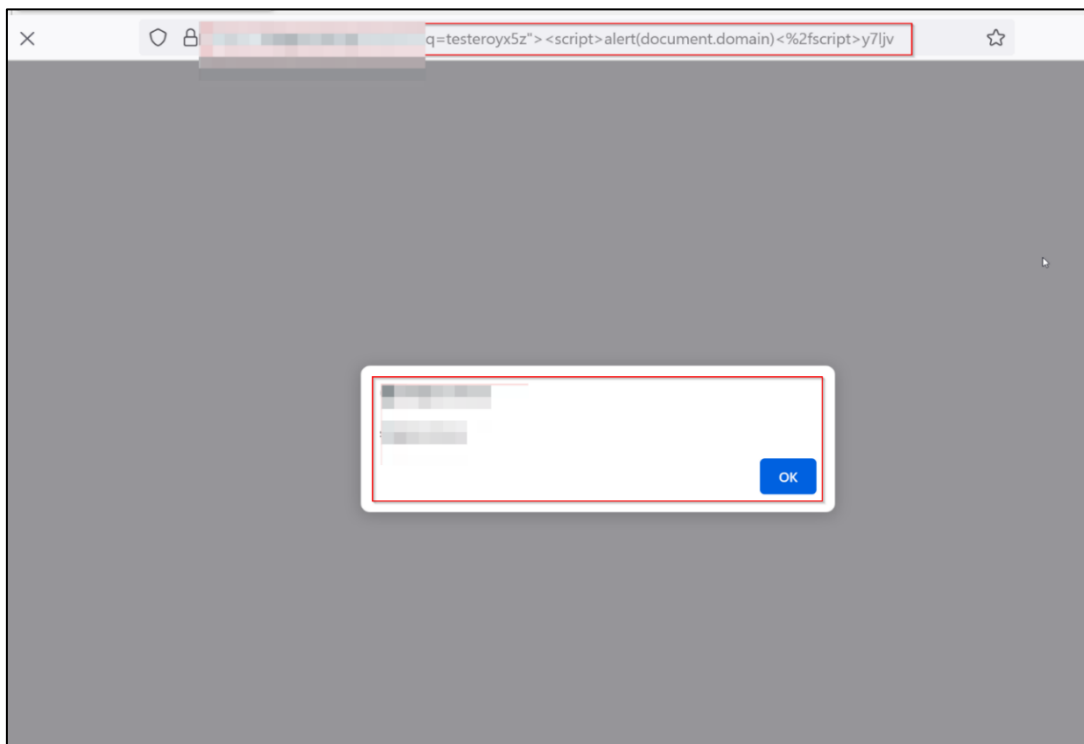
URL(s)
<code>https://subdomain.acme.corp.com//search/?q=<payload></code>
<code>https://acme.corp.com/search/?q=<payload></code>
<code>https://acme.corp.com/?s=<payload></code>

Proof of Concept:

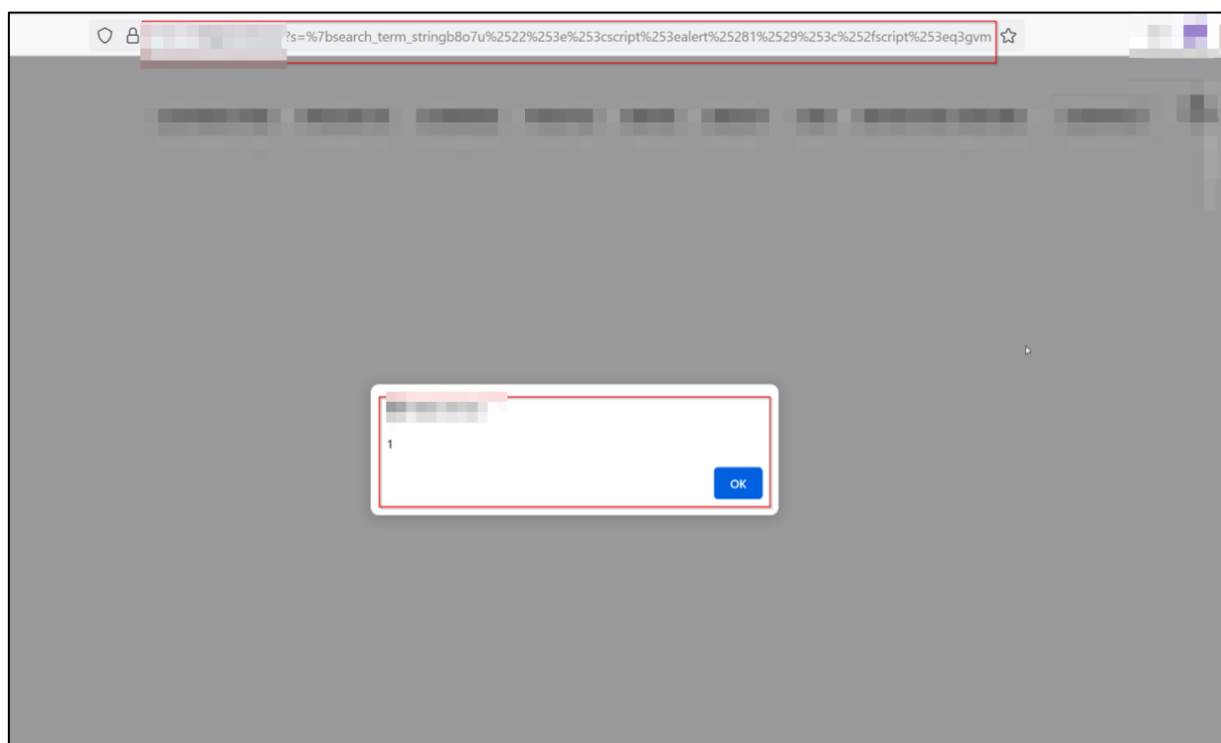
Instance 1: The payload "><script>alert(1)<%2Fscript>" was entered in the q parameter.



Instance 2: The payload `testeroyx5z"><script>alert (document. Domain) <%2Fscript>y7ljv` was entered in the q parameter.



Instance 3: The payload `%7bsearch_term_stringb8o7u%2522%253cscript%253ealert%25281%2529%253c%252fscript%253eq3gvm` was submitted in the s parameter.



Recommendation: To remediate Cross-Site Scripting effectively, a layered approach of using filtering and encoding is recommended. An allow list should be implemented based on the content the parameter should accept, filtering unwanted user input.

All user input should be entity-encoded, rendering any potentially dangerous characters such as angular brackets into their HTML entities.

Modifying the Content-Security-Policy header would also remediate the XSS. Only trusted script sources should be allowed, and options such as “unsafe-eval” and “unsafe-inline” should not be included.

Reference(s): <https://owasp.org/www-community/attacks/xss/>

5.2 Username Enumeration

Risk: Medium

CVSS Score: 5.3

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Description: WordPress allows username enumeration by utilising multiple techniques such as REST API calls, Various plugins like Yoast SEO, WordPress Login Feature, etc.

CVE-2017-5487 – The application hosting WordPress 4.7 does not restrict listing of post authors, which allow remote attackers to obtain user information.

Impact: Username enumeration vulnerabilities typically arise when the applications logic or responses indicate that a supplied account name or email is known by the application. By comparing the responses or the indication the application returns, an attacker can compile a list of valid accounts on the application and in turn use the list in other targeted attack such as a brute force attack or targeted phishing emails.

Instance(s): There were 5 instances of this issue identified at the following location:

URL(s)
https://acme.corp.com
https://subdomain.acme.corp.com
https://acme.corp.com
https://subdomain.acme.corp.com
https://subdomain.acme.corp.com/?rest_route=/wp/v2/users/ (CVE-2017-5487)

Proof of Concept:

Instance 1: Utilising the WPScan, we were able to enumerate users from the server as shown below:

```
[i] User(s) Identified:

[+] 
| Found By: Rss Generator (Passive Detection)
| Confirmed By: Rss Generator (Aggressive Detection)

[+] 
| Found By: Rss Generator (Passive Detection)
| Confirmed By:
| Wp Json Api (Aggressive Detection)
| Rss Generator (Aggressive Detection)
| Yoast Seo Author Sitemap (Aggressive Detection)
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Login Error Messages (Aggressive Detection)

[+] 
| Found By: Wp Json Api (Aggressive Detection)
| Confirmed By:
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Login Error Messages (Aggressive Detection)

[+] 
| Found By: Wp Json Api (Aggressive Detection)
| Confirmed By:
| Oembed API - Author URL (Aggressive Detection)
| Yoast Seo Author Sitemap (Aggressive Detection)
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Login Error Messages (Aggressive Detection)
```

Instance 2: Utilising the WPScan, we were able to enumerate users on the server as shown below:

```
[i] User(s) Identified:
```

```
| Found By: Wp Json Api (Aggressive Detection)
```

```
| Confirmed By:
```

```
| Oembed API - Author URL (Aggressive Detection)
```

```
| Rss Generator (Aggressive Detection)
```

```
| Yoast Seo Author Sitemap (Aggressive Detection)
```

```
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

```
| Found By: Wp Json Api (Aggressive Detection)
```

```
| Confirmed By:
```

```
| Yoast Seo Author Sitemap (Aggressive Detection)
```

```
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

```
| Found By: Wp Json Api (Aggressive Detection)
```

```
| Confirmed By:
```

```
| Yoast Seo Author Sitemap (Aggressive Detection)
```

```
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

```
| Found By: Wp Json Api (Aggressive Detection)
```

```
| Confirmed By:
```

```
| Rss Generator (Aggressive Detection)
```

```
| Yoast Seo Author Sitemap (Aggressive Detection)
```

```
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

```
| Found By: Wp Json Api (Aggressive Detection)
```

```
| Confirmed By:
```

```
| Yoast Seo Author Sitemap (Aggressive Detection)
```

```
| Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

```
| Found By: Wp Json Api (Aggressive Detection)
```

Instance 3: Utilising the WPScan, we were able to enumerate users on the server as shown below:

```
[i] User(s) Identified:

[+] [REDACTED]
| Found By: Wp Json Api (Aggressive Detection)
| [REDACTED]
| Confirmed By:
| | Rss Generator (Aggressive Detection)
| | Yoast Seo Author Sitemap (Aggressive Detection)
| | [REDACTED]
| | Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|
[+] [REDACTED]
| Found By: Wp Json Api (Aggressive Detection)
| [REDACTED]
| Confirmed By:
| | Yoast Seo Author Sitemap (Aggressive Detection)
| | [REDACTED]
| | Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|
[+] [REDACTED]
| Found By: Wp Json Api (Aggressive Detection)
| [REDACTED]
| Confirmed By:
| | Yoast Seo Author Sitemap (Aggressive Detection)
| | [REDACTED]
| | Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|
[+] [REDACTED]
| Found By: Wp Json Api (Aggressive Detection)
| [REDACTED]
| Confirmed By:
| | Oembed API - Author URL (Aggressive Detection)
| | [REDACTED]
| | Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|
[+] [REDACTED]
| Found By: Wp Json Api (Aggressive Detection)
| [REDACTED]
| Confirmed By:
| | Yoast Seo Author Sitemap (Aggressive Detection)
| | [REDACTED]
| | Author Id Brute Forcing - Author Pattern (Aggressive Detection)
|
[+] [REDACTED]
| Found By: Wp Json Api (Aggressive Detection)
| [REDACTED]
| Confirmed By:
```

Instance 4: Utilising the WPScan, we were able to enumerate users on the server as shown below:

```
[i] User(s) Identified:

| Found By: Wp Json Api (Aggressive Detection)
| Confirmed By:
|   Dembed API - Author URL (Aggressive Detection)
|   Author Sitemap (Aggressive Detection)
|   Author Id Brute Forcing - Author Pattern (Aggressive Detection)

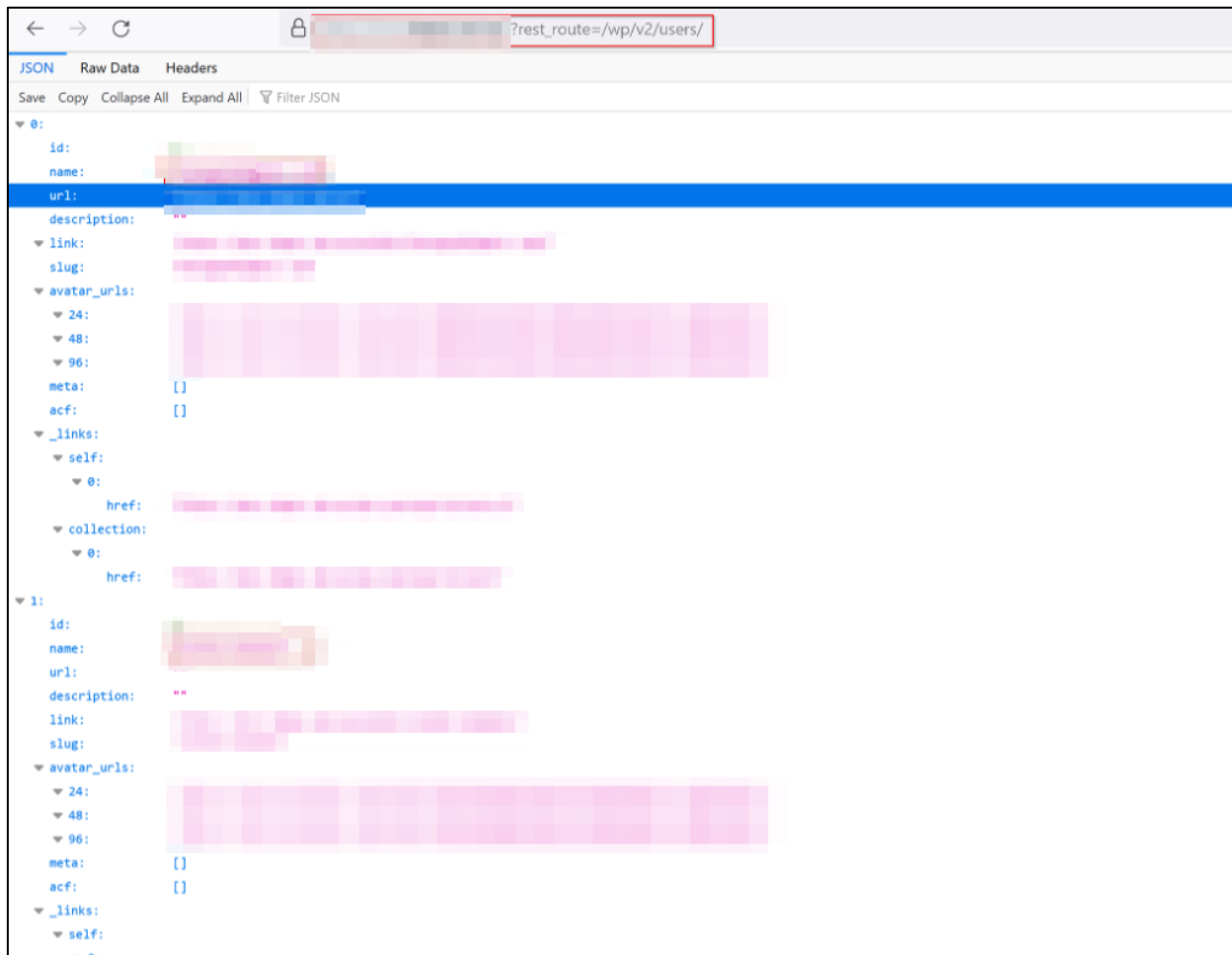
| Found By: Wp Json Api (Aggressive Detection)
| Confirmed By: Author Sitemap (Aggressive Detection)

| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

Instance 5: Utilising CVE-2017-5487, we were able to enumerate users by visiting the following



Recommendation: We recommend, Install WP-Hardening plugin to harden WP security.

URL: <https://wordpress.org/plugins/wp-security-hardening/>

Reference(s):

<https://wordpress.org/plugins/stop-user-enumeration/>

<https://www.acunetix.com/vulnerabilities/web/wordpress-username-enumeration/>

5.3 Log.txt File Accessible

Risk: **Medium**

CVSS Score: 4.8

CVSS Vector: CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:L

Description: It was observed that sensitive log file (log.txt) was publicly accessible.

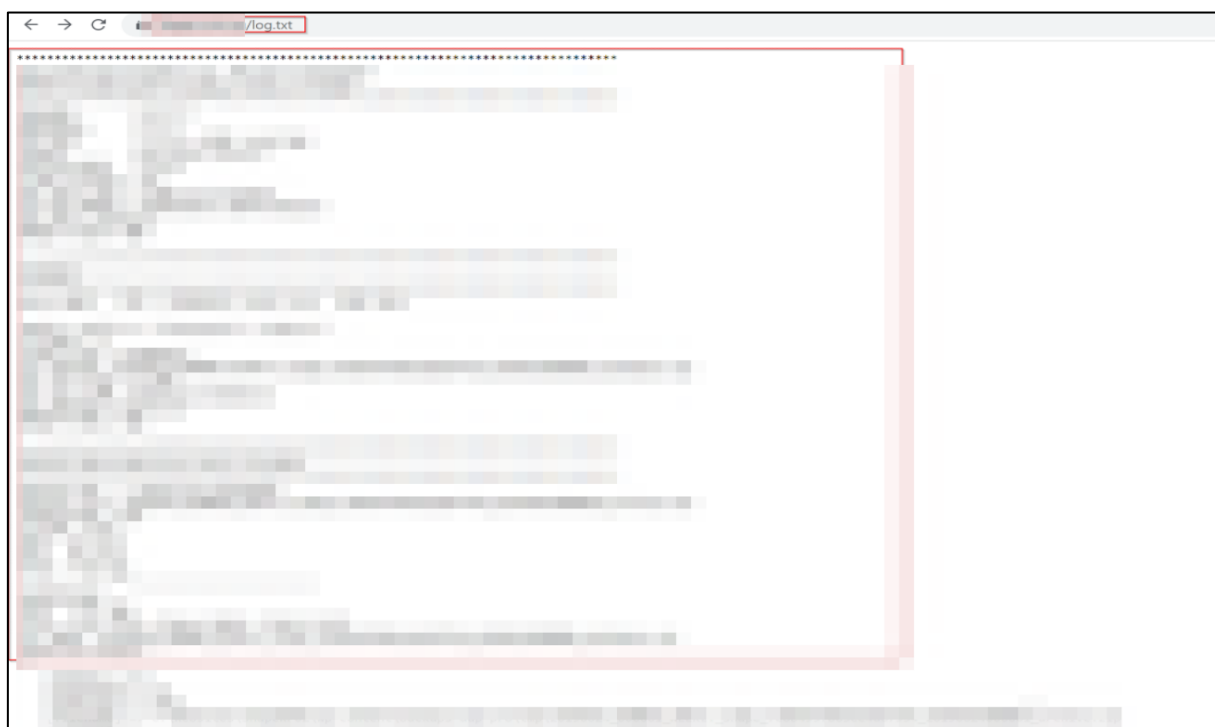
Information written to log files can be of a sensitive nature and give valuable guidance to an attacker or expose sensitive user information. While logging all information may be helpful during development stages, it is important that logging levels be set appropriately before a product ship so that sensitive user data and system information are not accidentally exposed to potential attackers. The weakness is introduced during Architecture and Design, Implementation, Operation stages.

Impact: An attacker can utilise such information to understand the application's design, path and other sensitive information for further attacks.

Proof of Concept: There was 1 instance of this issue identified at the following location:

URL(s)
https://acme.corp.com/log.txt

The payload log.txt was submitted in the URL and in response sensitive information related to the backend server and associated SQL server was revealed.



Recommendation: We recommend, restrict such files from unauthorised access.

Reference(s):

<https://portswigger.net/web-security/information-disclosure>

<https://www.tenable.com/plugins/nessus/121041>

5.4 Cross-Origin Resource Sharing (CORS) Misconfiguration

Risk: **Medium**

CVSS Score: 4.8

CVSS Vector: CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:L

Description: It was observed that Cross Origin Header was Misconfigured in the application. HTML5 makes it possible for JavaScript to access data across domains if a new HTTP header called Access-Control-Allow-Origin is defined. With this header, a Web server defines which other domains can access its domain using cross-origin requests. It was observed that using arbitrary domain as the value of the Access-Control-Allow-Origin header indicates that the application's data is accessible to JavaScript running on that domain.

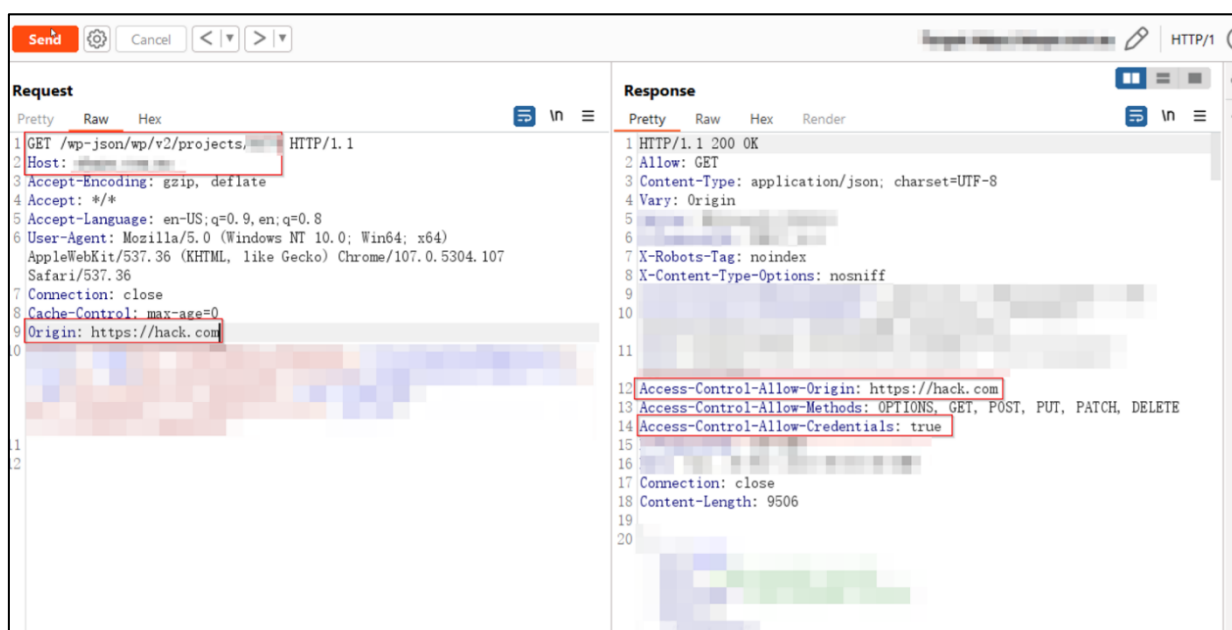
Impact: A CORS misconfiguration can leave the application at a high-risk of compromise resulting in an impact on the confidentiality and integrity of data by allowing third-party sites to carry out privileged requests through your web site's authenticated users such as retrieving user information.

Proof of Concept: There was 1 instance of this issue identified at the following location:

URL(s)

`https://acme.corp.com/wp-json/wp/v2/projects/1234`

Enter payload Origin: `https://hack.com` in the application's request header. Following screenshot shows that the entered payload is getting reflected in response header `Access-Control-Allow-Origin`. Further, `Access-Control-Allow-Credentials` is set to `true`.



Recommendation: It is recommended to allow only selected, trusted domains in the `Access-Control-Allow-Origin` header. Prefer whitelisting domains over blacklisting and do not use `*` wildcard nor blindly return the Origin header content without any checks.

Reference(s):

<https://owasp.org/www-community/attacks/CORS-OriginHeaderScrutiny>

<https://portswigger.net/web-security/cors>

6 Informational Risks

These are the concerns found by Krylo as part of your evaluation that do not pose a direct or immediate danger to [Client]. These are frequently technical or operational realities that reflect a considerable variation from normal suggested practice but do not constitute a significant vulnerability or risk in and of themselves. These should be assessed and considered as part of your overall risk profile, and appropriately reduced once the more serious hazards have been addressed.

6.1 Missing HTTP Security Headers

Risk: Informational

CVSS Score: 0.0

CVSS Vector: N/A

Description: HTTP security headers are a method of hardening the default level of security which is offered by web browsers. It was identified during the assessment that certain HTTP security headers were not implemented, this does not constitute a vulnerability in itself but lowers the level of security provided by the web browser.

Impact: The following headers are missing from the response:

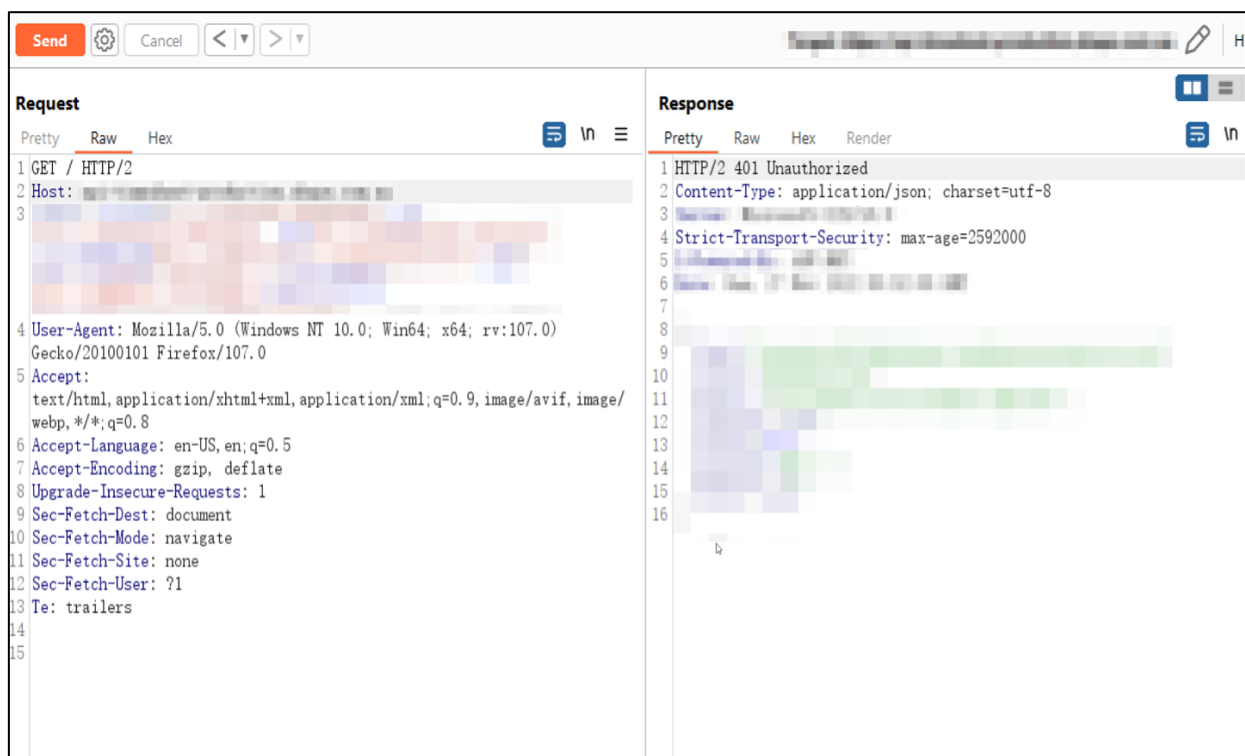
1. **Strict-Transport-Security** - Enforces the use of HTTPS within the web browser, no data will be sent over HTTP. This also helps to fix cookies set without the secure flag.
2. **Content-Security-Policy** - Allows whitelisting of web application resource locations, such as where scripts can be loaded from, and where the application can be framed. This helps to mitigate attacks such as clickjacking and cross-site scripting.
3. **X-Content-Type-Options** - Helps to prevent cross-site scripting by preventing the browser from attempting to enumerate MIME types within the server response.
4. **Referrer-Policy** - Reduces when the referrer header is set, preventing sensitive data within URLs being transmitted to third parties or over insecure connections.
5. **X-Frame-Options** - Not implemented in Chrome in favour of CSP. However, is used to control which sites may frame the web application. Preventing framing can help mitigate clickjacking attacks.

Instance(s): There were 5 instances of this issue identified at the following location:

URL(s)
https://api-secret-data.acme.corp.com
https://acme.corp.com
https://subdomain.acme.corp.com
https://acme.corp.com
https://subdomain.acme.corp.com

Proof of Concept:

Instance 1:

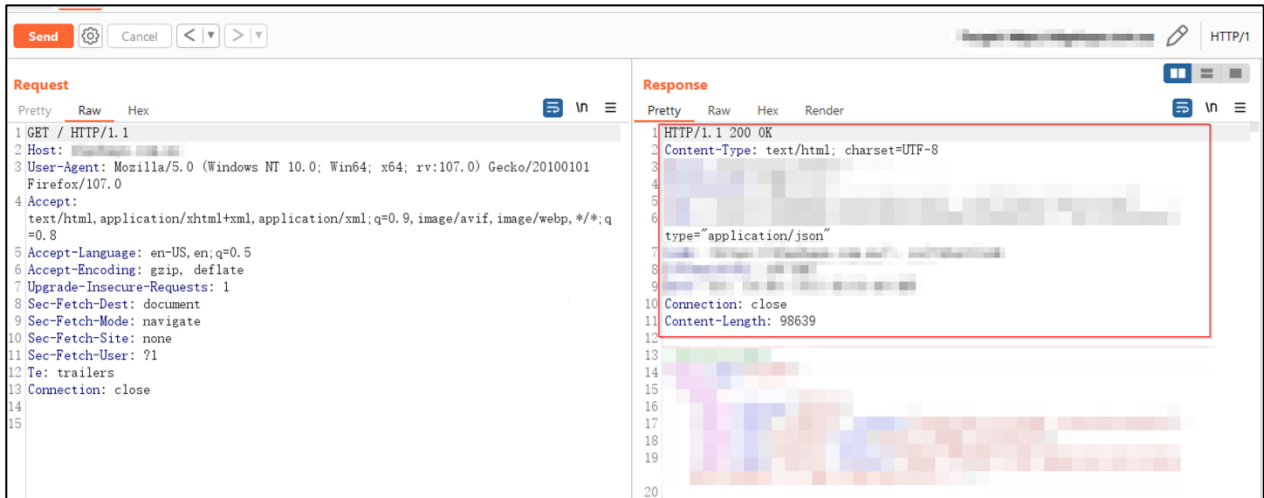


The screenshot shows a network request and response in a browser's developer tools. The request is a GET / HTTP/2 to https://api-secret-data.acme.corp.com. The response is a 401 Unauthorized status with Content-Type: application/json; charset=utf-8 and Strict-Transport-Security: max-age=2592000. The response body is empty.

Request	Response
1 GET / HTTP/2	1 HTTP/2 401 Unauthorized
2 Host: https://api-secret-data.acme.corp.com	2 Content-Type: application/json; charset=utf-8
3	3
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0) Gecko/20100101 Firefox/107.0	4 Strict-Transport-Security: max-age=2592000
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	5
6 Accept-Language: en-US,en;q=0.5	6
7 Accept-Encoding: gzip, deflate	7
8 Upgrade-Insecure-Requests: 1	8
9 Sec-Fetch-Dest: document	9
10 Sec-Fetch-Mode: navigate	10
11 Sec-Fetch-Site: none	11
12 Sec-Fetch-User: ?1	12
13 Te: trailers	13
14	14
15	15
	16

From the response above, it can be seen that Content-Security-Policy, X-Content-Type-Options, Referrer-Policy and X-Frame-Options headers are missing.

Instance 2:



Request

```

1 GET / HTTP/1.1
2 Host: [redacted]
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0) Gecko/20100101 Firefox/107.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13 Connection: close
14
15

```

Response

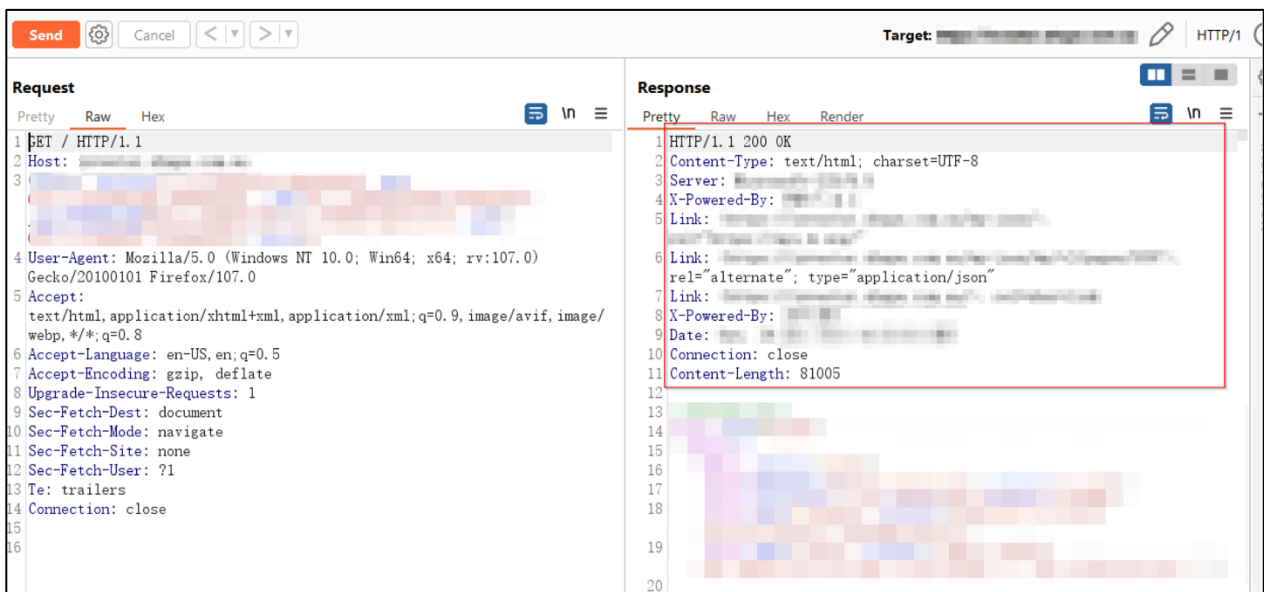
```

1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=UTF-8
3
4 [redacted]
5
6 type="application/json"
7 [redacted]
8 [redacted]
9 [redacted]
10 Connection: close
11 Content-Length: 98639
12
13 [redacted]
14 [redacted]
15 [redacted]
16 [redacted]
17 [redacted]
18 [redacted]
19 [redacted]
20 [redacted]

```

From the response above, it can be seen that Strict-Transport-Security, Content-Security-Policy, X-Content-Type-Options, Referrer-Policy and X-Frame-Options headers are missing.

Instance 3:



Request

```

1 GET / HTTP/1.1
2 Host: [redacted]
3 [redacted]
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0) Gecko/20100101 Firefox/107.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Upgrade-Insecure-Requests: 1
9 Sec-Fetch-Dest: document
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-Site: none
12 Sec-Fetch-User: ?1
13 Te: trailers
14 Connection: close
15
16

```

Response

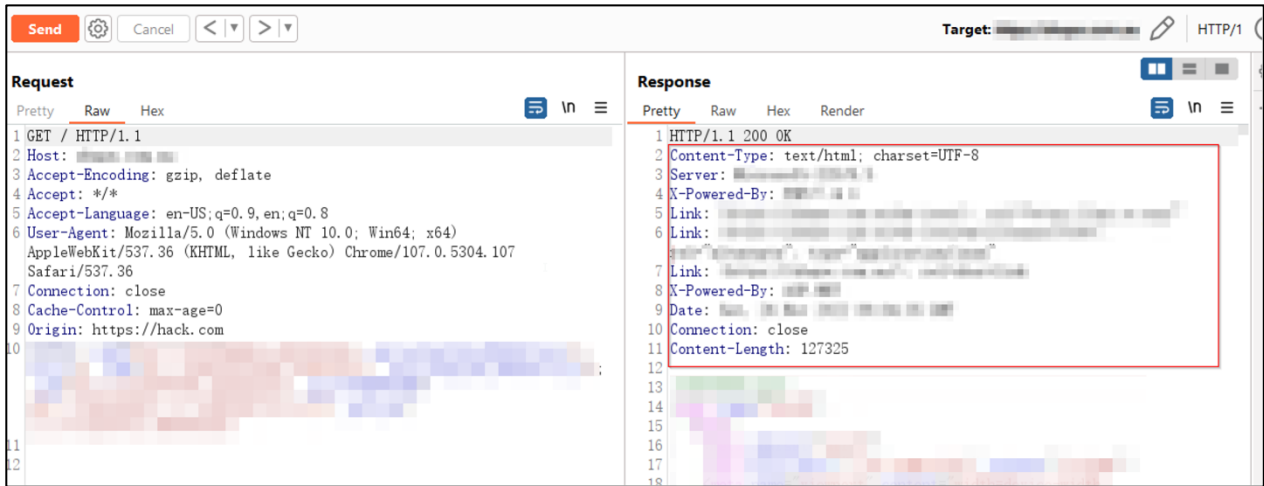
```

1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=UTF-8
3 Server: [redacted]
4 X-Powered-By: [redacted]
5 Link: [redacted]
6 Link: [redacted]
7 Link: [redacted]
8 X-Powered-By: [redacted]
9 Date: [redacted]
10 Connection: close
11 Content-Length: 81005
12
13 [redacted]
14 [redacted]
15 [redacted]
16 [redacted]
17 [redacted]
18 [redacted]
19 [redacted]
20 [redacted]

```

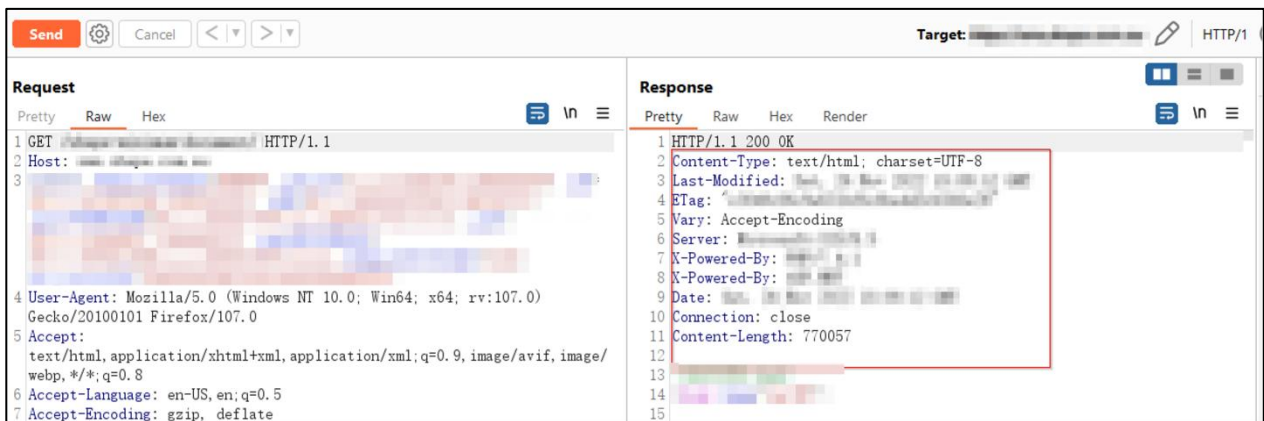
From the response above, it can be seen that Strict-Transport-Security, Content-Security-Policy, X-Content-Type-Options, Referrer-Policy and X-Frame-Options headers are missing.

Instance 4:



From the response above, it can be seen that Strict-Transport-Security, Content-Security-Policy, X-Content-Type-Options, Referrer-Policy and X-Frame-Options headers are missing.

Instance 5:



From the response above, it can be seen that Strict-Transport-Security, Content-Security-Policy, X-Content-Type-Options, Referrer-Policy and X-Frame-Options headers are missing.

Recommendation: For each of the security headers following the recommendations:

1. **Strict-Transport-Security** - It is recommended that this header be set with a max-age of at least 180 days and the preload directive. For example: Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
2. **Content-Security-Policy** - Enforce a whitelist of trusted script sources and ensure that "unsafe-eval" and "unsafe-inline" are blocked, for example: Content-Security-Policy "default-src https: <url>"
3. **X-Content-Type-Options** - Content sniffing is a behaviour of certain browsers when they try to determine the content type and encoding of the response even when

these properties are defined correctly. This behaviour can be explicitly forbidden with the additional HTTP response header: X-Content-Type-Options: nosniff

4. Referrer-Policy - Consider preventing the “Referrer” header from being transmitted to third-parties and over insecure links, using this header. For example, a referrer policy of “same-origin” would only set the referrer header if the request was made from the same origin: Referrer-Policy: same-origin
5. X-Frame-Options - Setting this header to “DENY” or “SAMEORIGIN” can mitigate clickjacking attacks. For example: X-Frame-Options: DENY

Reference(s):

<https://owasp.org/www-community/Security-Headers>

<https://cheatsheetseries.owasp.org/cheatsheets/HTTP-Headers-Cheat-Sheet.html>

6.2 Verbose Response Headers

Risk: Informational

CVSS Score: 0.0

CVSS Vector: N/A

Description: The Web applications were revealing “Server” and “X-Powered-By” in the response headers.

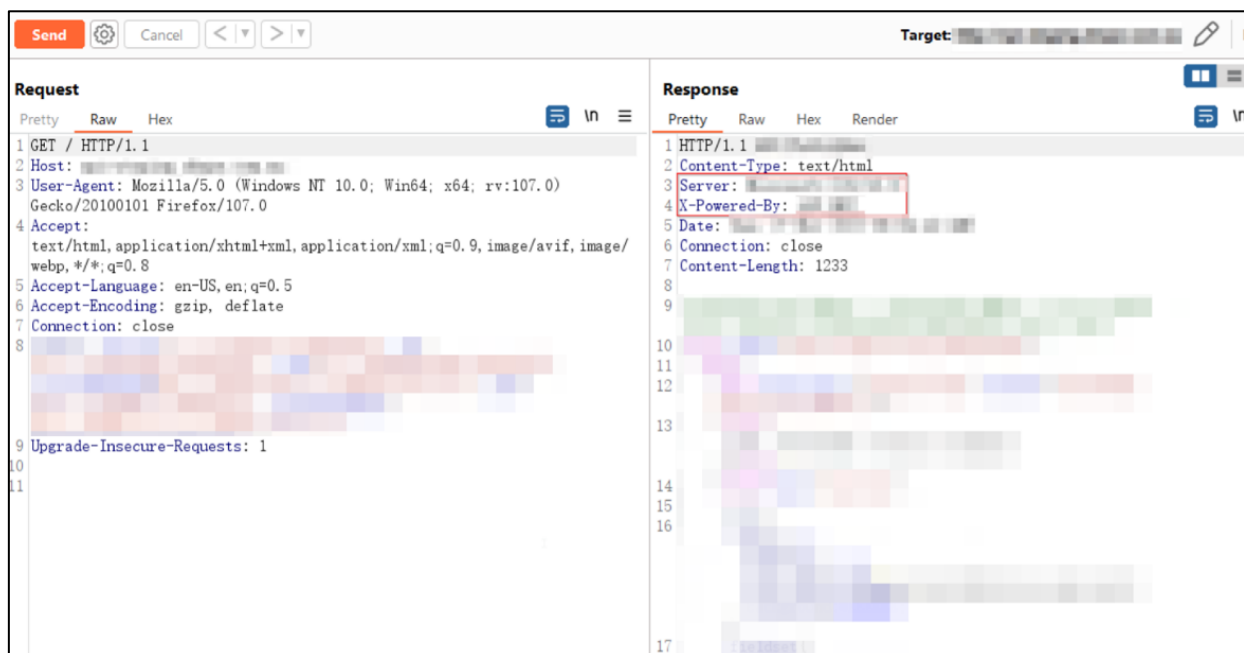
Impact: Knowing the version and type of a running web server and software installed, allows attackers to determine known vulnerabilities and issue targeted attacks against the application.

Instance(s): There were 7 instances of this issue identified at the following location:

URL(s)
https://api-secret.acme.corp.com
https://api-secret-data.acme.corp.com
https://acme.corp.com
https://subdomain.acme.corp.com
https://acme.corp.com
https://subdomain.acme.corp.com
https://subdomain.acme.corp.com

Proof of Concept:

Instance 1:

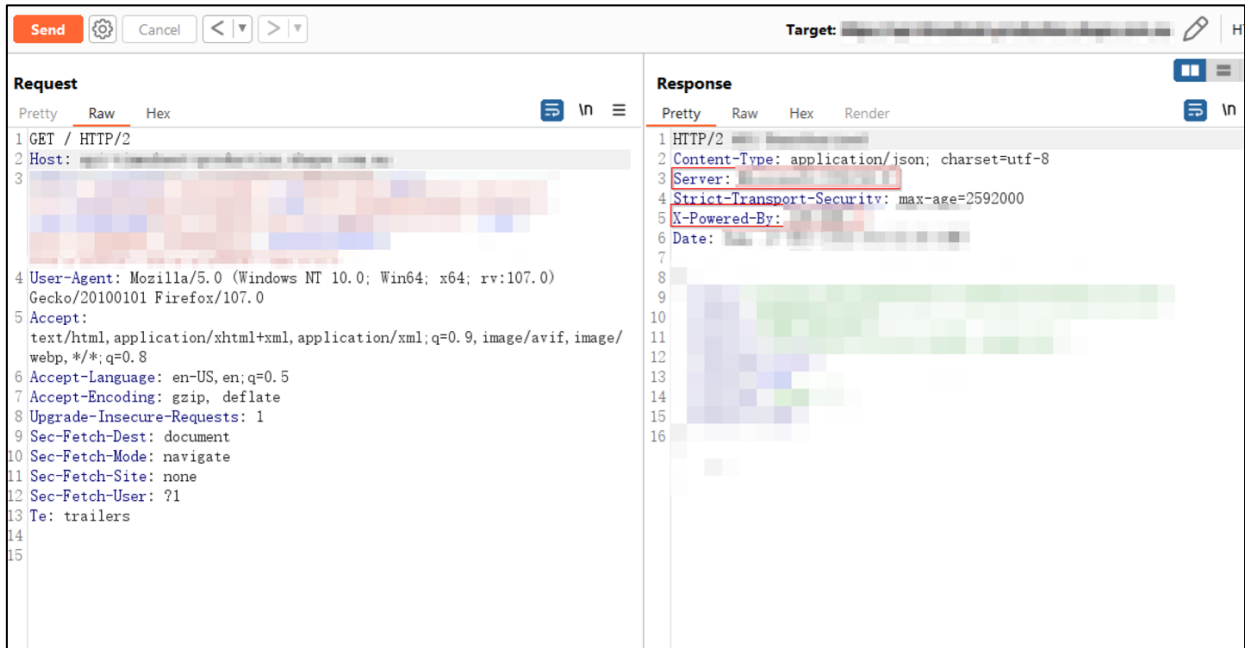


The screenshot displays the 'Request' and 'Response' tabs of a web browser's developer tools. The 'Request' tab shows a GET request to `http://api-secret.acme.corp.com`. The 'Response' tab shows the server's response, which includes the following headers:

- `Content-Type: text/html`
- `Server: secret-ABC/00.0`
- `X-Powered-By: Secret.Tech`

In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech.

Instance 2:



Request

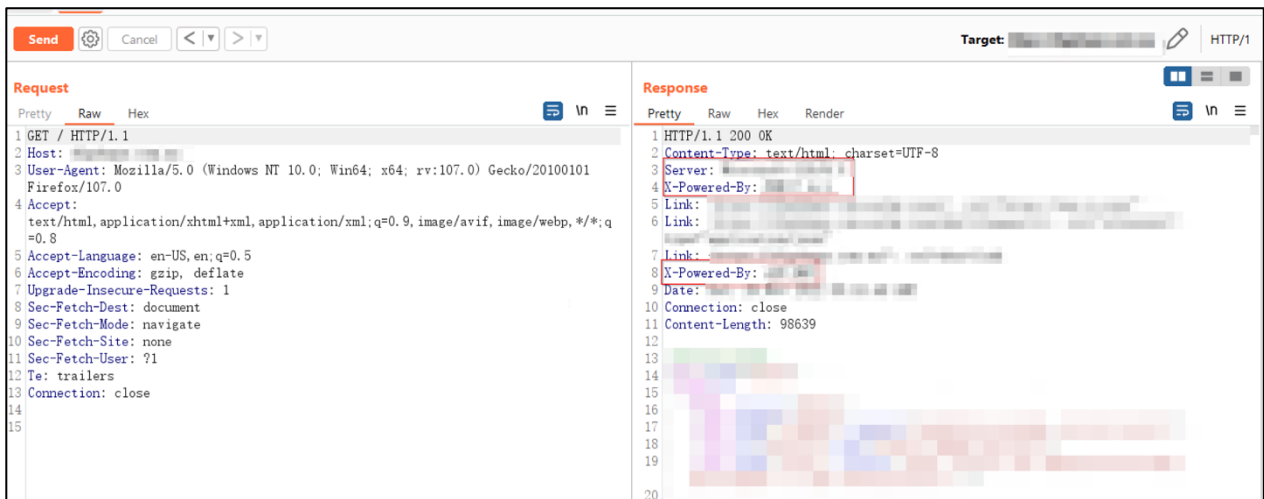
```
1 GET / HTTP/2
2 Host: [redacted]
3
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0)
  Gecko/20100101 Firefox/107.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
  webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Upgrade-Insecure-Requests: 1
9 Sec-Fetch-Dest: document
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-Site: none
12 Sec-Fetch-User: ?1
13 Te: trailers
14
15
```

Response

```
1 HTTP/2
2 Content-Type: application/json; charset=utf-8
3 Server: secret-ABC/00.0
4 Strict-Transport-Security: max-age=2592000
5 X-Powered-By: Secret.Tech
6 Date: [redacted]
7
8 [redacted]
9
10 [redacted]
11 [redacted]
12 [redacted]
13 [redacted]
14 [redacted]
15 [redacted]
16 [redacted]
```

In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech

Instance 3:



Request

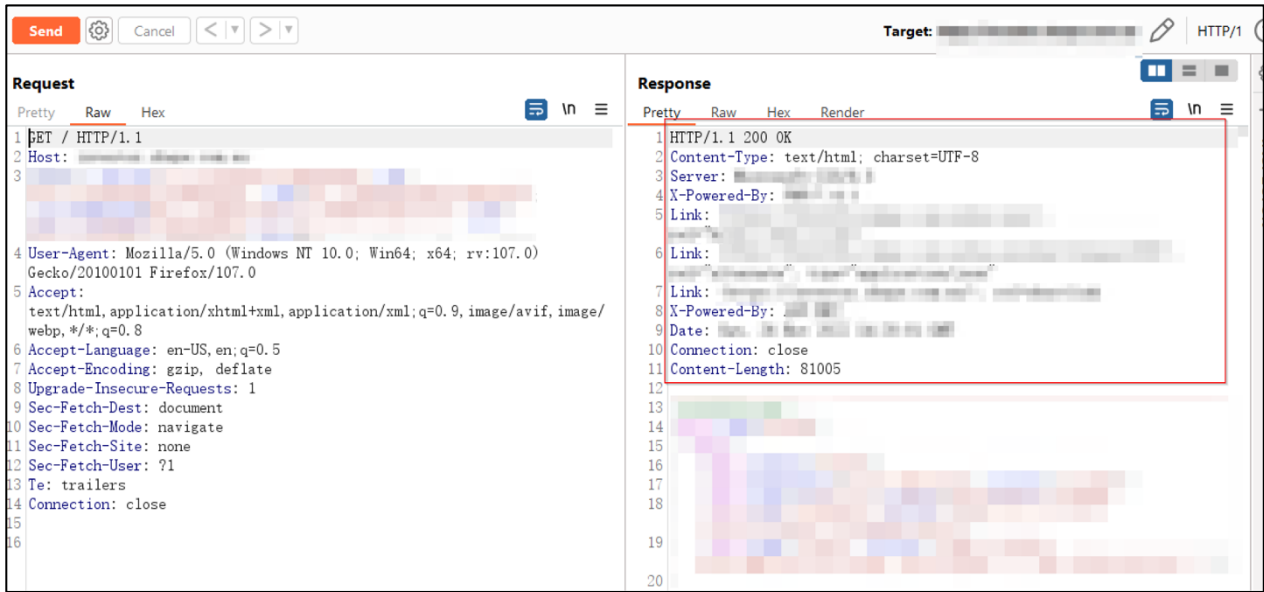
```
1 GET / HTTP/1.1
2 Host: [redacted]
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0) Gecko/20100101
  Firefox/107.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q
  =0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Upgrade-Insecure-Requests: 1
8 Sec-Fetch-Dest: document
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-Site: none
11 Sec-Fetch-User: ?1
12 Te: trailers
13 Connection: close
14
15
```

Response

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=UTF-8
3 Server: secret-ABC/00.0
4 X-Powered-By: Secret.Tech & Secret/0.0.0
5 Link: [redacted]
6 Link: [redacted]
7 Link: [redacted]
8 X-Powered-By: Secret.Tech & Secret/0.0.0
9 Date: [redacted]
10 Connection: close
11 Content-Length: 98639
12
13 [redacted]
14 [redacted]
15 [redacted]
16 [redacted]
17 [redacted]
18 [redacted]
19 [redacted]
20 [redacted]
```

In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech & Secret/0.0.0.

Instance 4:



Request

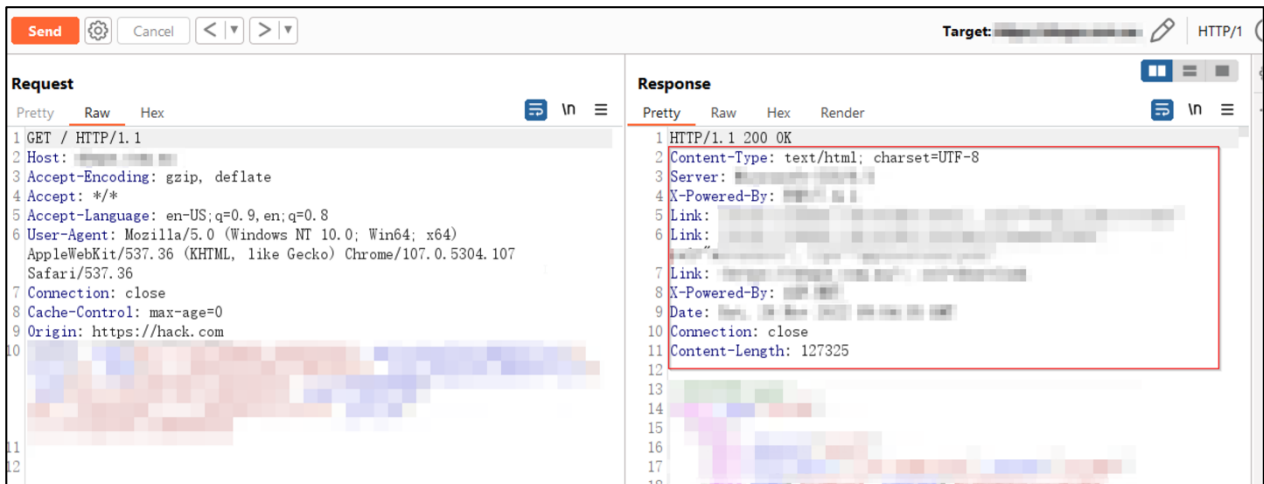
```
1 POST / HTTP/1.1
2 Host: 
3 
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0)
5 Gecko/20100101 Firefox/107.0
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: none
13 Te: trailers
14 Connection: close
```

Response

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=UTF-8
3 Server: secret-ABC/00.0
4 X-Powered-By: Secret.Tech & Secret/0.0.0
5 Link: 
6 Link: 
7 Link: 
8 X-Powered-By: 
9 Date: 
10 Connection: close
11 Content-Length: 81005
```

In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech & Secret/0.0.0.

Instance 5:



Request

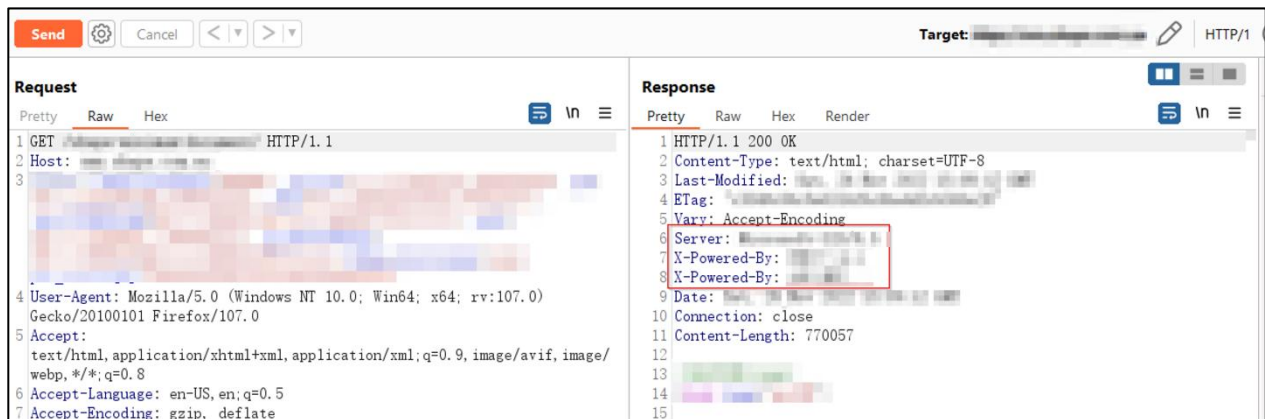
```
1 GET / HTTP/1.1
2 Host: 
3 Accept-Encoding: gzip, deflate
4 Accept: */*
5 Accept-Language: en-US;q=0.9,en;q=0.8
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
7 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107
8 Safari/537.36
9 Connection: close
10 Cache-Control: max-age=0
11 Origin: https://hack.com
```

Response

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=UTF-8
3 Server: secret-ABC/00.0
4 X-Powered-By: Secret.Tech & Secret/0.0.0
5 Link: 
6 Link: 
7 Link: 
8 X-Powered-By: 
9 Date: 
10 Connection: close
11 Content-Length: 127325
```

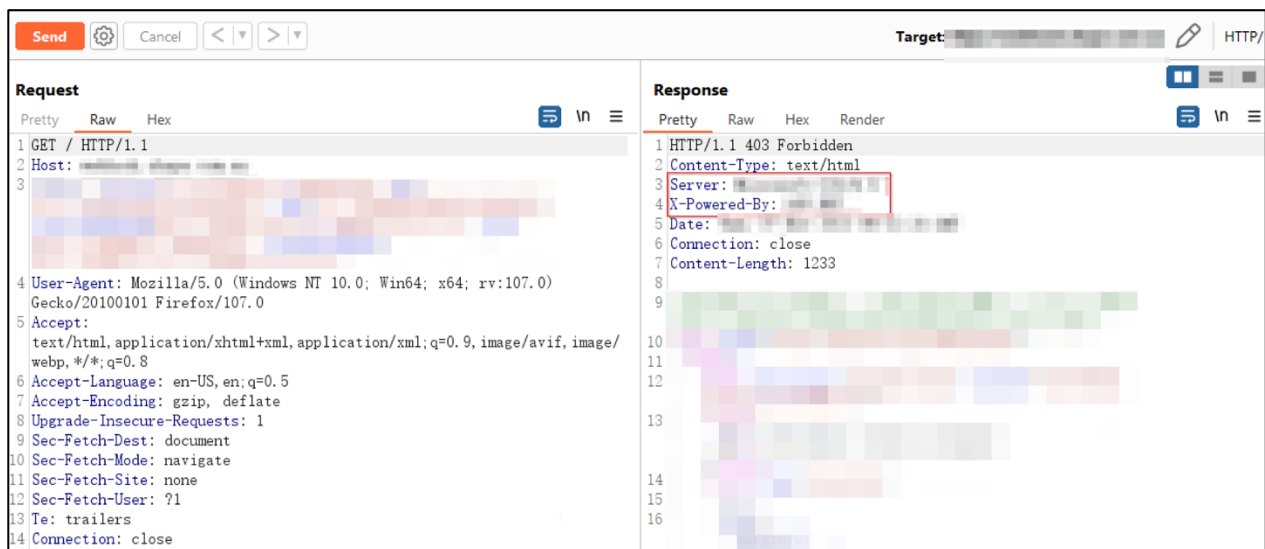
In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech & Secret/0.0.0.

Instance 6:



In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech & Secret/0.0.0.

Instance 7:



In the response above, the server header reveals that the web server is secret-ABC/00.0 and X-Powered-By Secret.Tech.

Recommendation: Remove software and versions from the web server's response headers. Information should be disclosed on a "Need to Know" basis. Information pertaining to software version numbers should be removed wherever possible. To remove the HTTP

Server header, the "removeServerHeader" attribute should be set to true in the web.config file.

For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <system.webServer>
    <security>
      <requestFiltering removeServerHeader ="true" />
    </security>
  </system.webServer>
</configuration>
```

Reference(s):

https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/01-Information_Gathering/02-Fingerprint_Web_Server

6.3 Unencrypted Communication

Risk: Informational

CVSS Score: 0.0

CVSS Vector: N/A

Description: The application is configured to be accessed over plaintext HTTP communication protocol in addition to being accessible over HTTPS.

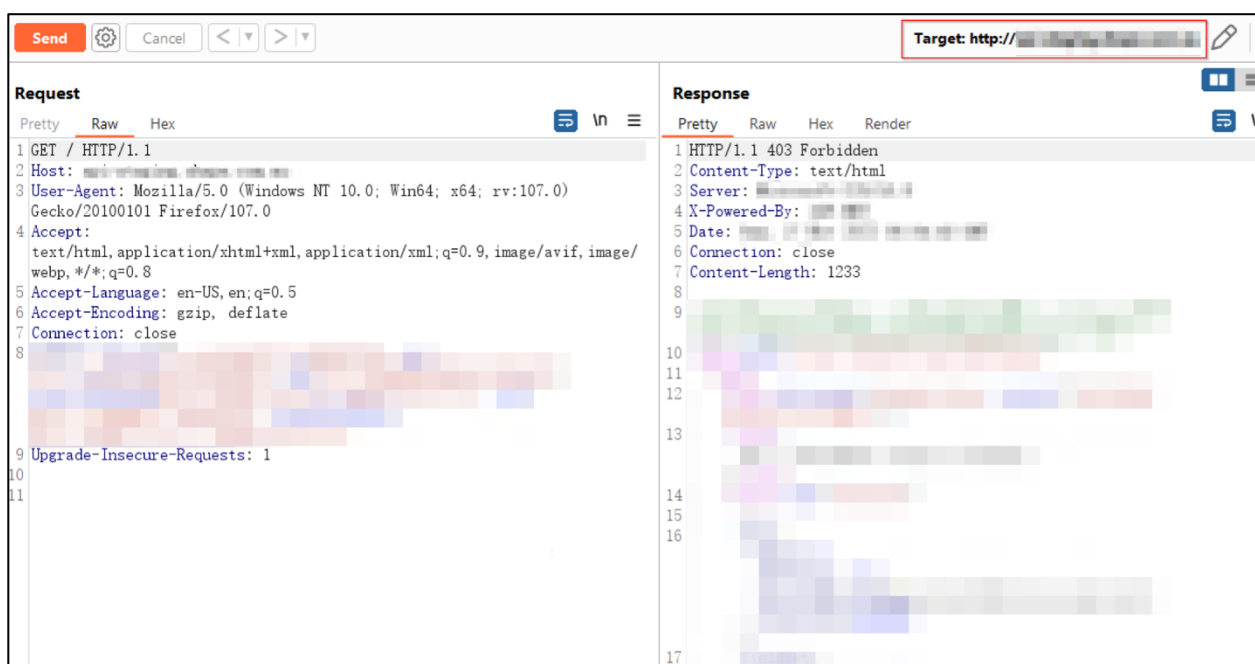
Impact: Any data transmitted over unencrypted HTTP protocol, including credentials, personally identifiable information (PII) such as name, email, phone and mobile number and address, and financial information would be in plain text. A malicious individual could use network monitoring software to obtain the unencrypted contents of the communications, if the user makes a request over unencrypted communication protocol. Access of the system over wireless networks would be a targeted environment.

Instance(s): There were 2 instances of this issue identified at the following location:

URL(s)
https://api-secret.acme.corp.com
https://subdomain.acme.corp.com

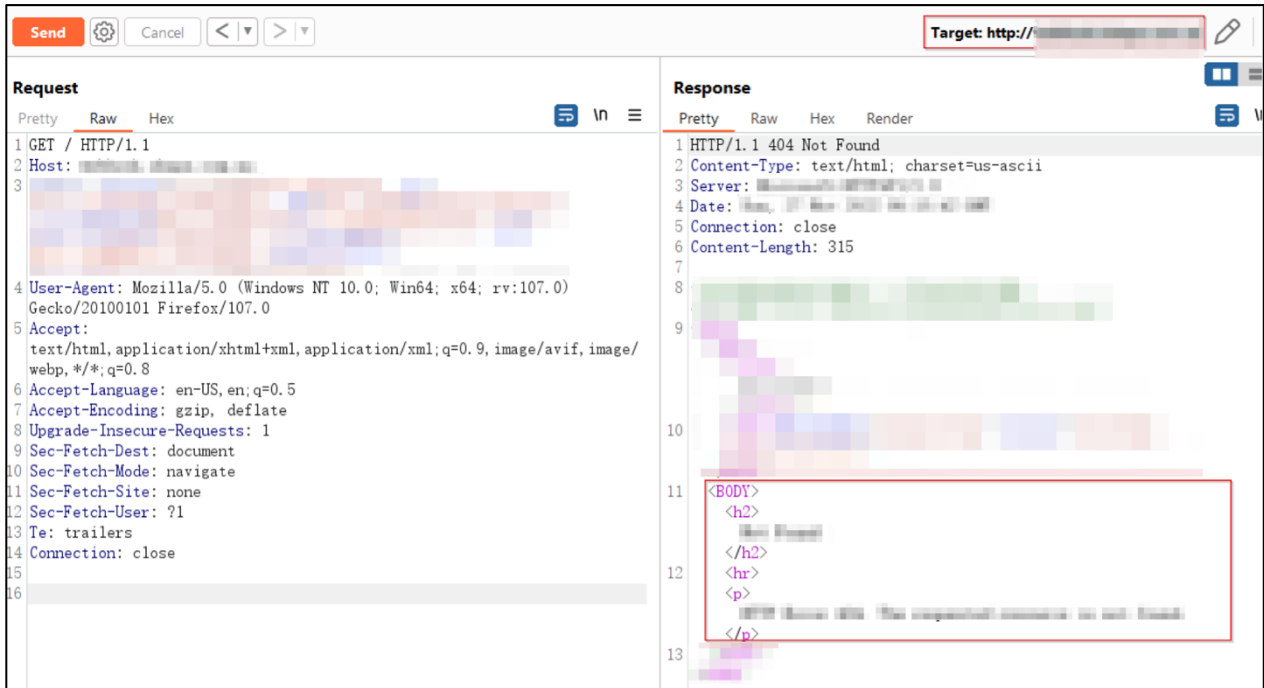
Proof of Concept:

Instance 1: The server responded with 403 Forbidden when an HTTP request was sent.



The screenshot shows a web browser's developer tools interface. At the top, there's a 'Send' button and a 'Target' field containing 'http://api-secret.acme.corp.com'. Below this, the 'Request' tab is active, showing a GET request to 'http://api-secret.acme.corp.com'. The request headers include 'Host: api-secret.acme.corp.com', 'User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:107.0) Gecko/20100101 Firefox/107.0', 'Accept: text/html, application/xhtml+xml, application/xml;q=0.9, image/avif, image/webp, */*;q=0.8', 'Accept-Language: en-US,en;q=0.5', 'Accept-Encoding: gzip, deflate', 'Connection: close', and 'Upgrade-Insecure-Requests: 1'. The 'Response' tab is also active, showing a 403 Forbidden status with 'Content-Type: text/html' and 'Content-Length: 1233'. The response body is partially visible, showing a 403 Forbidden message.

Instance 2: The server responded with 404 Not Found when an HTTP request was sent.



The screenshot displays the Network tab of a web browser's developer tools. The top bar shows a 'Send' button and a 'Target' field with a red box around the URL. The 'Request' pane on the left shows a GET request to a host. The 'Response' pane on the right shows an HTTP/1.1 404 Not Found response. The response headers include Content-Type: text/html; charset=us-ascii, Server: Apache/2.4.18 (Ubuntu), Date: Mon, 17 Mar 2020 10:10:10 GMT, Connection: close, and Content-Length: 315. The response body is highlighted with a red box and contains the following HTML:

```
<BODY>
<h2>
  404 Not Found
</h2>
<hr>
<p>
  HTTP Error 404: The requested resource is not found.
</p>
```

Recommendation: We recommend reconfiguring the web-server to disable access of the application over unencrypted HTTP protocol and implement secure HTTPS protocol to prevent risk of interception and subsequent disclosure of sensitive and confidential information.

Reference(s): https://portswigger.net/kb/issues/01000200_unencrypted-communications

7 Document Control

Name	Date	Version	Comment
[Tester 1]	XX/XX/202X	0.1	Initial Draft
[Tester 2]	XX/XX/202X	0.2	Technical QA (Final)
[Tester 1]	XX/XX/202X	1.0	Final Release