

Sri Lanka Institute of Information Technology



Year 2 semester 2

IT23360600

G. P. I. Perera

SLIIT KANDY UNI

BUG BOUNTY REPORT 2

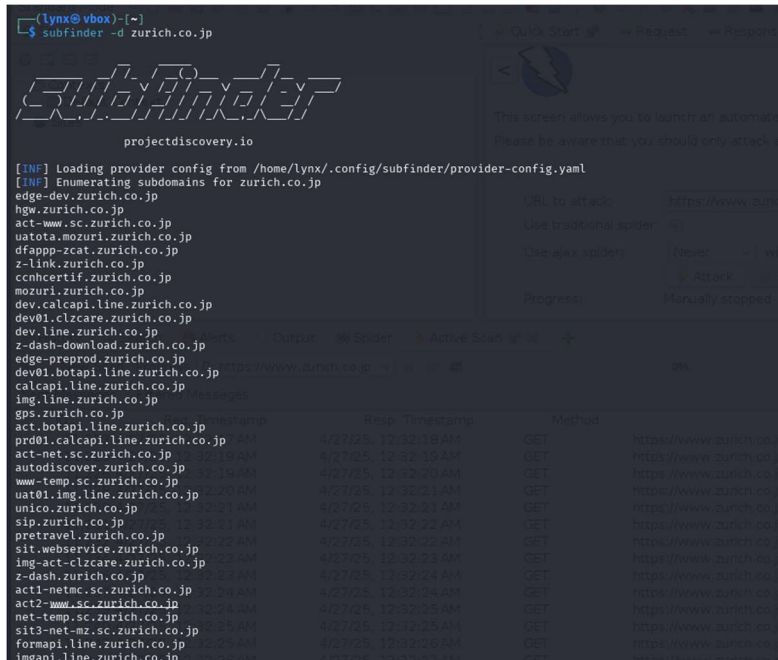
Web Security – IE2062

B.Sc. (Hons) in information Technology Specializing in Cyber Security

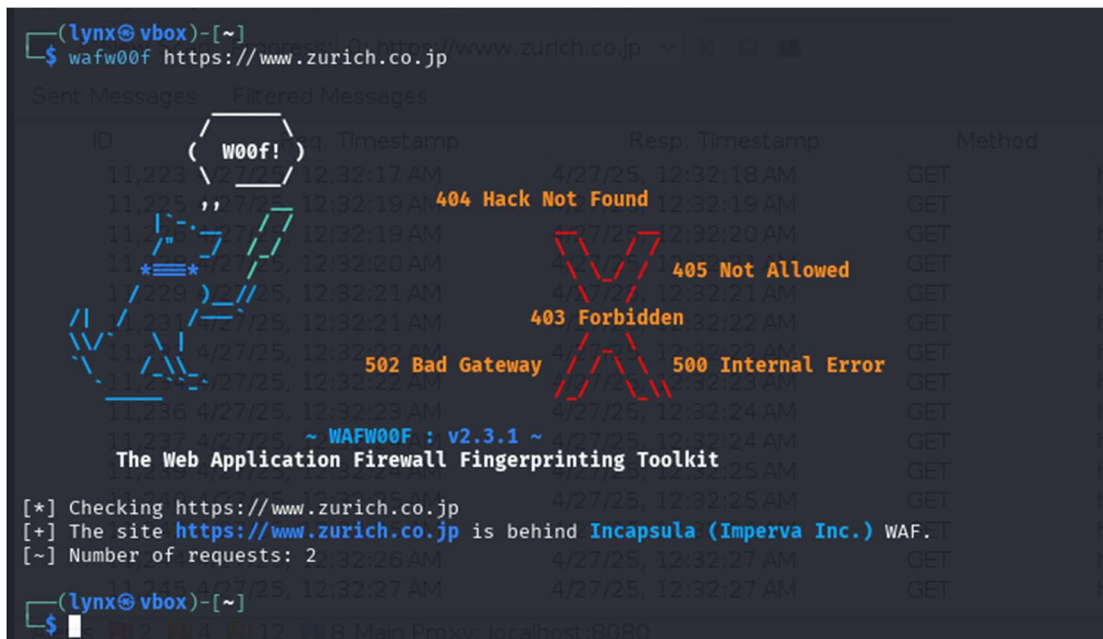
1. Requirement gathering and analysis

Selected sub domain	zurich.co.jp
Hackerone URL	https://hackerone.com/zurich-insurance
IP address	-

Subdomain list fom subfinder:



Firewall detection:



Nmap scan:

```
(lynx@vbox)-[~]
$ nmap www.zurich.co.jp
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-27 01:04 EDT
RTTVar has grown to over 2.3 seconds, decreasing to 2.0
RTTVar has grown to over 2.3 seconds, decreasing to 2.0
Nmap scan report for www.zurich.co.jp (45.60.16.215)
Host is up (0.22s latency).
Not shown: 600 filtered tcp ports (no-response)
PORT      STATE SERVICE
21/tcp    open  ftp
25/tcp    open  smtp
37/tcp    open  time
43/tcp    open  whois
53/tcp    open  domain
80/tcp    open  http
81/tcp    open  hosts2-ns
82/tcp    open  xfer
83/tcp    open  mit-ml-dev
84/tcp    open  ctf
85/tcp    open  mit-ml-dev
88/tcp    open  kerberos-sec
```

Active scan result from OWASP ZAP:

The screenshot shows the OWASP ZAP (Zed Attack Proxy) interface. The main window displays a list of alerts generated during an active scan. The selected alert is 'Script Served From Malicious Domain (polyfill)' (73). The detailed view of this alert is shown on the right, including the following information:

- URL:** <https://www.zurich.co.jp/car/useful/>
- Risk:** High
- Confidence:** High
- Parameter:** <https://polyfill.io/v2/polyfill.min.js?features=IntersectionObserver>
- Attack:** `<script src='https://polyfill.io/v2/polyfill.min.js?features=IntersectionObserver'></script>`
- Evidence:** `<script src='https://polyfill.io/v2/polyfill.min.js?features=IntersectionObserver'></script>`
- CWE ID:** 829
- WASC ID:** 15
- Description:** The page includes one or more script files loaded from one of the 'polyfill' domains. These are not associated with the polyfill.js library and are known to serve malicious content.
- Other Info:**
- Solution:** Change all scripts to use a known good source based on their documentation.
- Reference:** <https://sansec.io/research/polyfill-supply-chain-attack>, <https://x.com/triblondon/status/1761852117579427975>
- Alert Tags:**

Below the detailed view, another alert is visible in the list: 'Big Redirect Detected (Potential Sensitive Information Leak)' (53). The detailed view of this alert is also shown, including the following information:

- URL:** <https://www.zurich.co.jp/car/feature/other-zurich/>
- Risk:** Low
- Confidence:** Medium
- Parameter:**
- Attack:**
- Evidence:**
- CWE ID:** 201
- WASC ID:** 13
- Source:** Passive (10044 - Big Redirect Detected (Potential Sensitive Information Leak))
- Alert Reference:** 10044-1
- Input Vector:**
- 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.).
- Other Info:**
- Location header URI length:** 9 (/aboutus/)
- Predicted response size:** 309
- Response Body Length:** 123,262

Network tools in the inspect section:

The screenshot displays the Chrome DevTools interface. The top bar shows various tabs: Elements, Console, Sources, Network, Performance, Memory, Application, Privacy and security, Lighthouse, and Recorder. The Network panel is active, showing a list of requests. The 'polyfill.min.js?features=IntersectionObserver' request is highlighted in red, indicating a failed request (net:ERR_NAME_NOT_RESOLVED). The Console panel shows the corresponding error: 'Uncaught TypeError: Cannot read properties of null (reading 'previousElementSibling')' at 'following-banner.js:6' and 'Uncaught TypeError: Cannot read properties of null (reading 'add')' at 'slick.min.js:17'.

2. Report Details

1. Vulnerability Title: Script Served from Malicious Domain (polyfill)

2. Vulnerability Description

This is **not** *itself* a standalone vulnerability like XSS or SQLi.

It's more a **security issue** or **risk indicator** flagged in a scan or assessment.

What does it mean:

- The web app/site **loads JavaScript (a script)** from an **untrusted or malicious domain**.
- In this case, it mentions **polyfill**, which usually refers to JavaScript libraries that "fill in" missing browser features (like polyfill.io).
- If the domain serving the script is **compromised or malicious, they can inject whatever JavaScript they want** — potentially stealing cookies, session tokens, user data, etc.
- And, it was discovered that CSP headers are not set on the website too.

While scanning <https://www.zurich.co.jp> the domain had over 70+ URLs that used the malicious script loaded from the polyfill domain –

```
<script  
src="https://polyfill1.io/v2/polyfill.min.js?features=IntersectionObserver"></script>
```

Observation:

When you click on network bar, refresh the page and filter it to see js (javascript) and search for polyfill you can see the site is blocked and result the error – `ERR_NAME_NOT_RESOLVED`

Reason	Explanation
Domain is dead	The real polyfill.io domain may be deleted or taken down.
Domain is blocked	ISP, DNS server, or government firewall may block access.
Security software	Antivirus / DNS firewall like NextDNS, Cloudflare Gateway, AdGuard, or even your router might block it for safety.

Conclusion:

The inclusion of script files from malicious "polyfill" domains exposes the application and its users to significant security risks. These scripts are not affiliated with the legitimate Polyfill.io service and are known to deliver malicious content such as malware, spyware, and data-stealing code. As a result, any trust in client-side integrity is compromised, leading to potential theft of sensitive user data, session hijacking, defacement, or further propagation of malware.

Immediate action must be taken to remove references to these malicious domains and audit the site's dependencies to prevent future exploitation.

In the browser Network tab and wget test, the domain polyfill.io **fails to resolve** (`ERR_NAME_NOT_RESOLVED`).

But in the network tool inspection,

The script could not be downloaded because **DNS lookup failed**. No JavaScript code could be loaded from that domain. The browser showed security warnings related to permissions policy and cookies. The request for the script is stuck as **blocked/failed** inside the DevTools Network panel. The site still references the external script in the code (even though it doesn't load successfully). This means:

- The domain polyfill.io is either **deleted, disabled, expired, or blocked** by DNS/firewall.
- **Currently**, no malicious script is loading — **because** the browser cannot even reach the server.
- **If** the domain were re-registered or revived by an attacker in the future, malicious JavaScript **could** be served without any warning to users.

Inference:

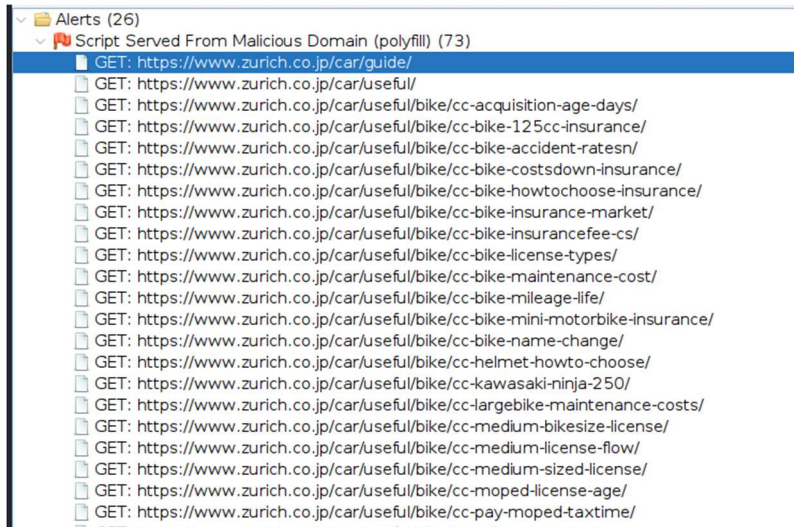
By injecting malicious scripts through trusted pages, attackers gain the ability to:

- Execute arbitrary JavaScript within the user's browser (Cross-Site Scripting-like behavior).
- Steal session cookies, authentication tokens, or personal information.
- Redirect users to phishing or malware sites.
- Perform drive-by downloads of malware.
- Damage the website's reputation and trust with users and search engines (e.g., SEO poisoning, browser warnings).

This exploitation demonstrates how critical supply chain security is — even a single compromised external script can jeopardize the entire application and its users.

3. Affected Components:

According to OWASP ZAP 70+ sites from the domain contain the malicious script.



4. Impact Assessment: from OWASP ZAP

Risk level	High
Confidence	High

5. Steps to reproduce –

1. OWASP ZAP –
Open OWASP ZAP click quick scan and enter the URL on a new session and attack the URL.
2. Network inspection tool –
Search the domain, right click and select inspect. Choose the network tab and filter JS from the tab. Discover that polyfill scripts are blocked.

6. Proposed mitigation or fix

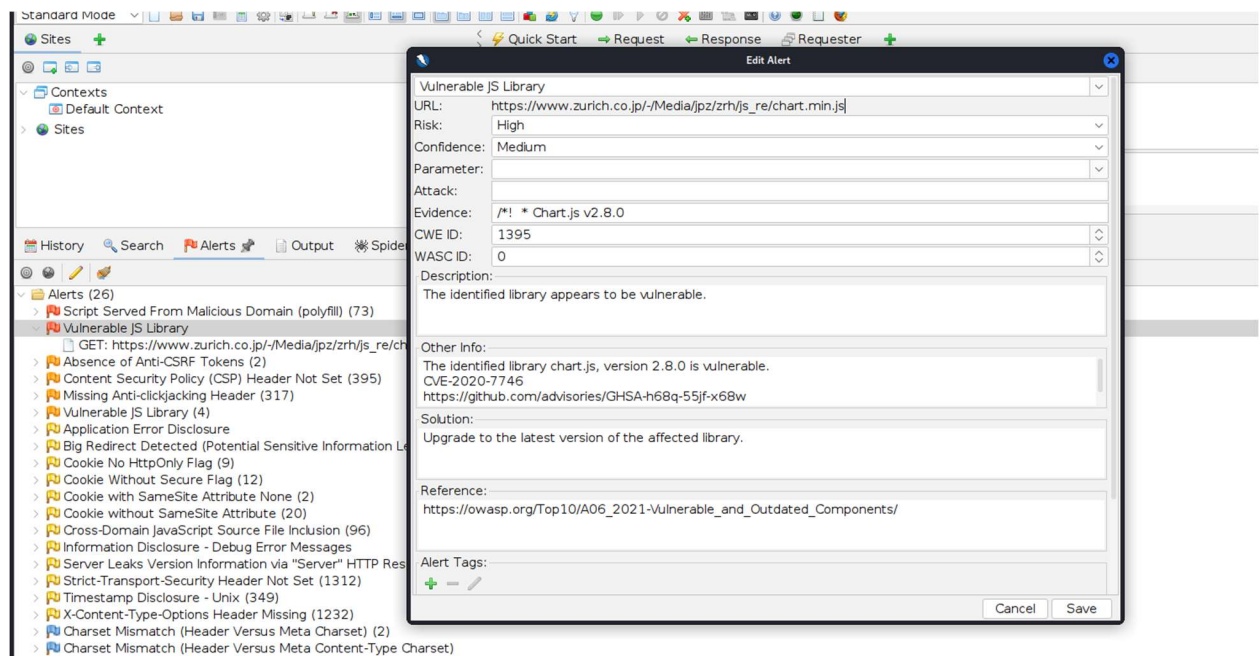
- **Immediately remove** all script references to malicious or untrusted "polyfill" domains from the application.
- **Self-host** critical JavaScript libraries, including any necessary polyfills, instead of relying on third-party CDNs to eliminate supply chain risks.
- **Implement Sub resource Integrity (SRI)** for any externally loaded scripts to ensure that the script content has not been tampered with.
- **Use a Content Security Policy (CSP)** to restrict which domains are allowed to load scripts, thereby reducing the risk of malicious script execution.

- **Conduct a full dependency audit** to verify the authenticity and integrity of all third-party resources and libraries currently used.
- **Monitor** and **regularly review** external resources to ensure they remain safe and up to date.

7. Additional findings -

Vulnerable js library –

OWASP ZAP:



Retire.js –

Retire.js

☒ Enabled ☐ Show unknown

jquery	1.12.4.min	Found in https://www.zurich.co.jp/-/Media/jpz/zrh/js_re/jquery-1.12.4.min.js - Vulnerability info:	
		Low jQuery 1.x and 2.x are End-of-Life and no longer receiving security updates 73 162	[1]
		Medium 3rd party CORS request may execute 2432 CVE-2015-9251 GHSA-rmxg-73gg-4p98	[1] [2] [3] [4] [5] [6]
		Medium jQuery before 3.4.0, as used in Drupal, Backdrop CMS, and other products, mishandles jQuery.extend(true, {}, ...) because of Object.prototype pollution CVE-2019-11358 4333 GHSA-6c3j-c64m-qhgq	[1] [2] [3]
		Medium passing HTML containing <option> elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. CVE-2020-11023 4647 GHSA-jpcq-cgw6-v4j6	[1]
		Medium Regex in its jQuery.htmlPrefilter sometimes may introduce XSS CVE-2020-11022 4642 GHSA-gxr4-xjj5-5px2	[1]


8. Submission:

#3114161

Script Served From Malicious Domain (polyfill)

[ADD HACKER SUMMARY](#)

TIMELINE · EXPORT

 lynx_jr2002 submitted a report to Zurich Insurance.

2 hours ago

This is not itself a standalone vulnerability like XSS or SQLi. It's more a security issue or risk indicator flagged in a scan or assessment.

- Your web app/site loads JavaScript (a script) from an untrusted or malicious domain.
- In this case, it mentions polyfill, which usually refers to JavaScript libraries that "fill in" missing browser features (like polyfill.io).
- If the domain serving the script is compromised or malicious, they can inject whatever JavaScript they want — potentially stealing cookies, session tokens, user data, etc.
- And also it was discovered that CSP headers are not set on the website too.

Impact

The inclusion of script files from malicious "polyfill" domains exposes the application and its users to significant security risks. These scripts are not affiliated with the legitimate Polyfill.io service and are known to deliver malicious content such as malware, spyware, and data-stealing code. As a result, any trust in client-side integrity is compromised, leading to potential theft of sensitive user data, session hijacking, defacement, or further propagation of malware.


Immediate action must be taken to remove references to these malicious domains and audit the site's dependencies to prevent future exploitation.

3 attachments

F4290456: Screenshot_2025-04-27_055004.png

F4290457: Screenshot_2025-04-27_053729.png

F4290458: Screenshot_2025-04-27_062336.png

 Add comment

Request Mediation

Leave a comment to all participants

Write

Preview

Parsed with Markdown

9. Reply:



[h1_analyst_magnus](#) **HackerOne triage** closed the report and changed the status to ● Informative.

7 days ago

Hi [@lynx_jr2002](#),

Thank you for all the efforts you put into writing this report, however, please note that automated vulnerability scanners commonly have low priority issues and/or false positives. Before submitting the results from a scanner, please take a moment to confirm that the reported issues are valid and exploitable with business impact.

For any scenario to be accepted as a practical security vulnerability you need to demonstrate the security issue along with a working proof-of-concept, if you are able to leverage this behavior, then please provide a working POC that can be used to reproduce the issue and demonstrate a security impact upon other users along with sufficient evidence and we will review this report again.

Regards,

[@h1_analyst_magnus](#)