Sri Lanka Institute of Information Technology

Bug Bounty - Report 06

HTTP to HTTPS Insecure Transition in Form Post

Student Name – Wanasinghe N.K

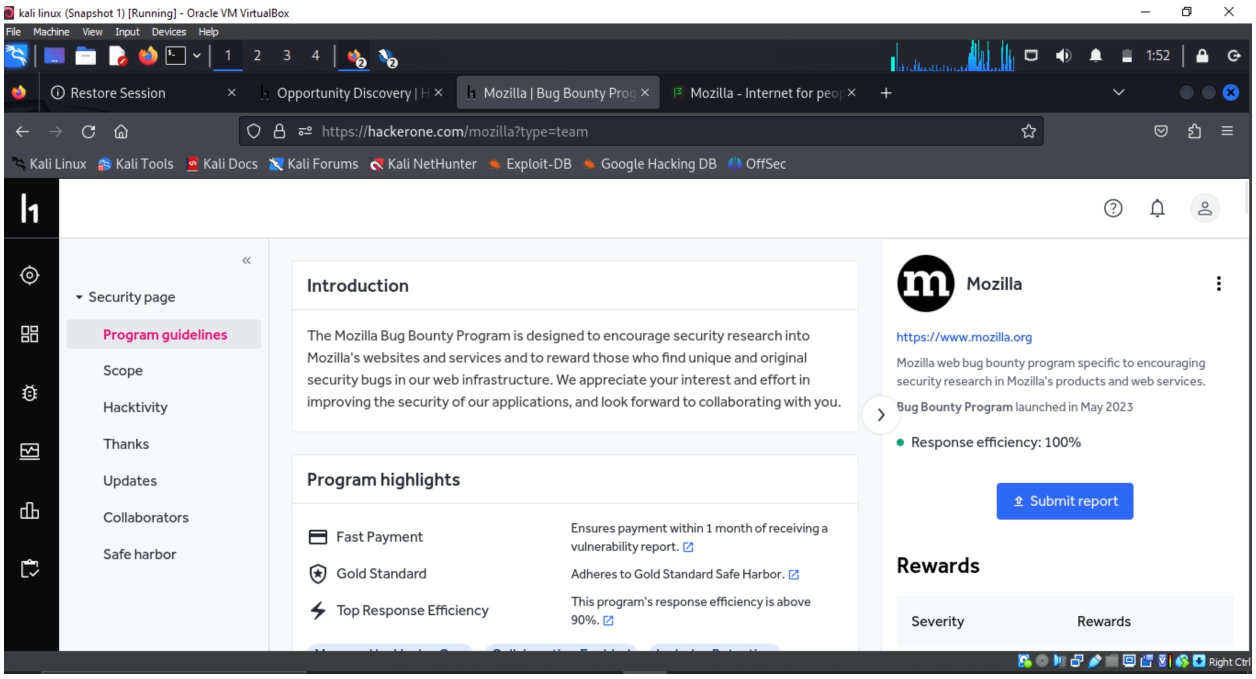
Student ID – IT23221000

IE2062 - Web Security

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

Report 06 – mozilla.org/en-US/ (Hackerone)

Main domain – www.mozilla.org/en-US/



Introduction

The Mozilla Bug Bounty Program is designed to encourage security research into Mozilla's websites and services and to reward those who find unique and original security bugs in our web infrastructure. We appreciate your interest and effort in improving the security of our applications, and look forward to collaborating with you.

Program highlights

Fast Payment	Ensures payment within 1 month of receiving a vulnerability report.
Gold Standard	Adheres to Gold Standard Safe Harbor.
Top Response Efficiency	This program's response efficiency is above 90%.

Mozilla

https://www.mozilla.org

Mozilla web bug bounty program specific to encouraging security research in Mozilla's products and web services.

Bug Bounty Program launched in May 2023

Response efficiency: 100%

Rewards

Severity	Rewards

I used OWASP ZAP tool to scan the website.

In scope domains

http://mozilla.org/en-US/

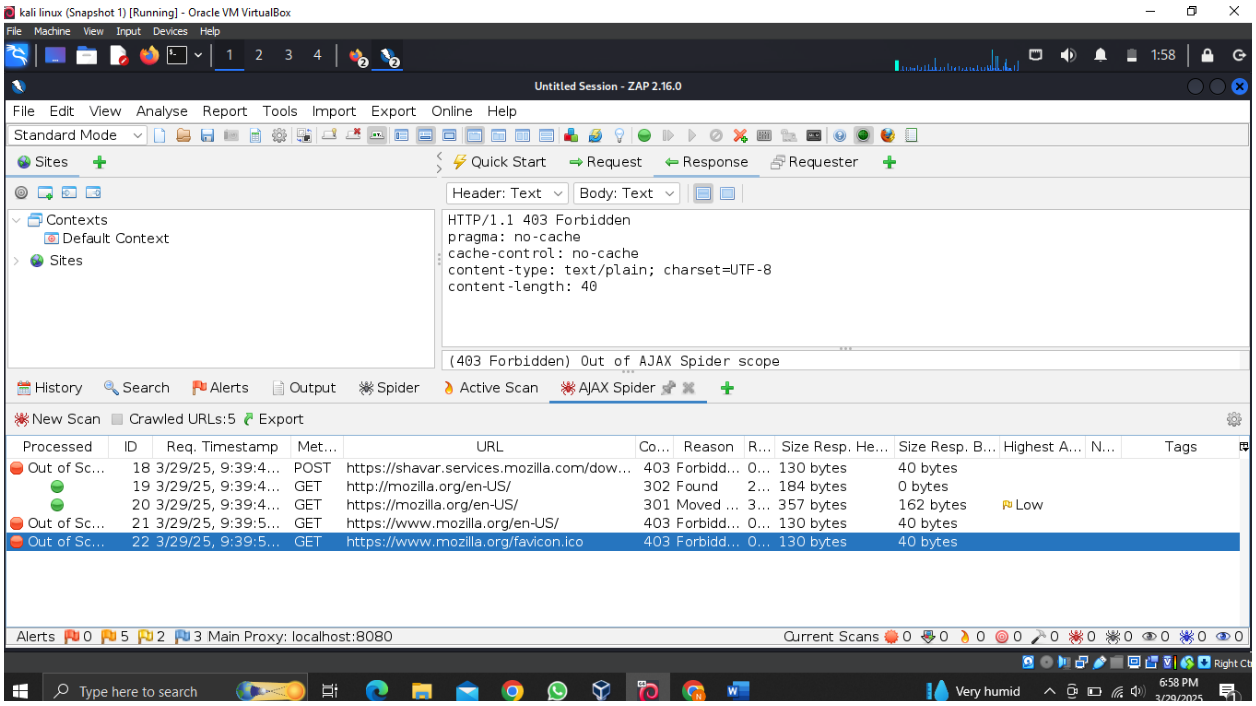
https://mozilla.org/en-US/

Out of scope domains

https://shavar.services.mozilla.com/downloads?client=navclient-auto-ffox&appver=115.13&pver=2.2

https://www.mozilla.org/en-US/

https://www.mozilla.org/favicon.ico



```
HTTP/1.1 403 Forbidden
pragma: no-cache
cache-control: no-cache
content-type: text/plain; charset=UTF-8
content-length: 40
```

```
(403 Forbidden) Out of AJAX Spider scope
```

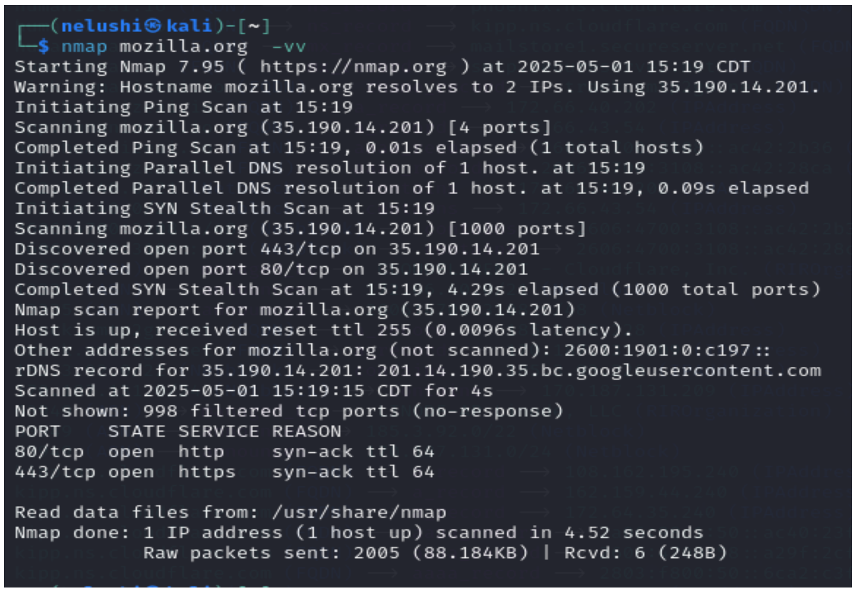
New Scan Crawled URLs:5 Export

Processed	ID	Req. Timestamp	Met…	URL	Co…	Reason	R…	Size Resp. He…	Size Resp. B…	Highest A…	N…	Tags
🔴 Out of Sc…	18	3/29/25, 9:39:4…	POST	https://shavar.services.mozilla.com/dow…	403	Forbidd…	0…	130 bytes	40 bytes			
🟢	19	3/29/25, 9:39:4…	GET	http://mozilla.org/en-US/	302	Found	2…	184 bytes	0 bytes			
🟢	20	3/29/25, 9:39:4…	GET	https://mozilla.org/en-US/	301	Moved …	3…	357 bytes	162 bytes	Low		
🔴 Out of Sc…	21	3/29/25, 9:39:5…	GET	https://www.mozilla.org/en-US/	403	Forbidd…	0…	130 bytes	40 bytes			
🔴 Out of Sc…	22	3/29/25, 9:39:5…	GET	https://www.mozilla.org/favicon.ico	403	Forbidd…	0…	130 bytes	40 bytes			

Alerts 0 5 2 3 Main Proxy: localhost:8080   Current Scans 0 0 0 0 0 0 0 0 0 0

Nmap – Network scanning and enumeration

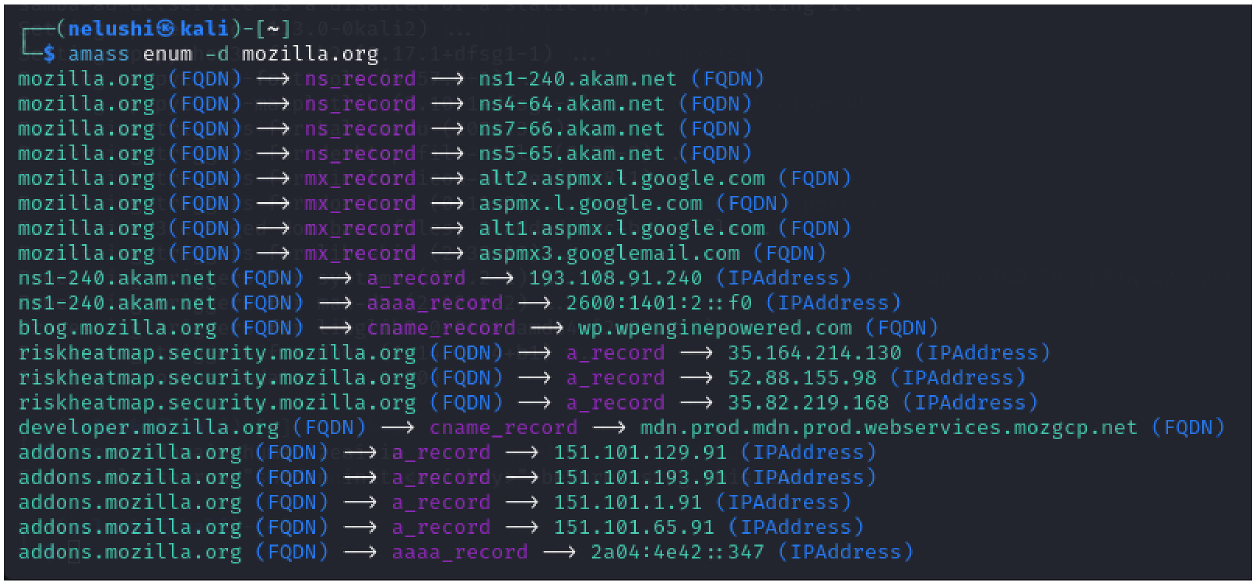
I found all the open ports and detected the running services on the target server using Nmap.



```
┌──(nelushi㉿kali)-[~]
└─$ nmap mozilla.org  -vv
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-01 15:19 CDT
Warning: Hostname mozilla.org resolves to 2 IPs. Using 35.190.14.201.
Initiating Ping Scan at 15:19
Scanning mozilla.org (35.190.14.201) [4 ports]
Completed Ping Scan at 15:19, 0.01s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 15:19
Completed Parallel DNS resolution of 1 host. at 15:19, 0.09s elapsed
Initiating SYN Stealth Scan at 15:19
Scanning mozilla.org (35.190.14.201) [1000 ports]
Discovered open port 443/tcp on 35.190.14.201
Discovered open port 80/tcp on 35.190.14.201
Completed SYN Stealth Scan at 15:19, 4.29s elapsed (1000 total ports)
Nmap scan report for mozilla.org (35.190.14.201)
Host is up, received reset ttl 255 (0.0096s latency).
Other addresses for mozilla.org (not scanned): 2600:1901:0:c197::
rDNS record for 35.190.14.201: 201.14.190.35.bc.googleusercontent.com
Scanned at 2025-05-01 15:19:15 CDT for 4s
Not shown: 998 filtered tcp ports (no-response)
PORT    STATE SERVICE REASON
80/tcp  open  http    syn-ack ttl 64
443/tcp open  https   syn-ack ttl 64

Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 4.52 seconds
           Raw packets sent: 2005 (88.184KB) | Rcvd: 6 (248B)
```

Amass – Subdomain and DNS mapping

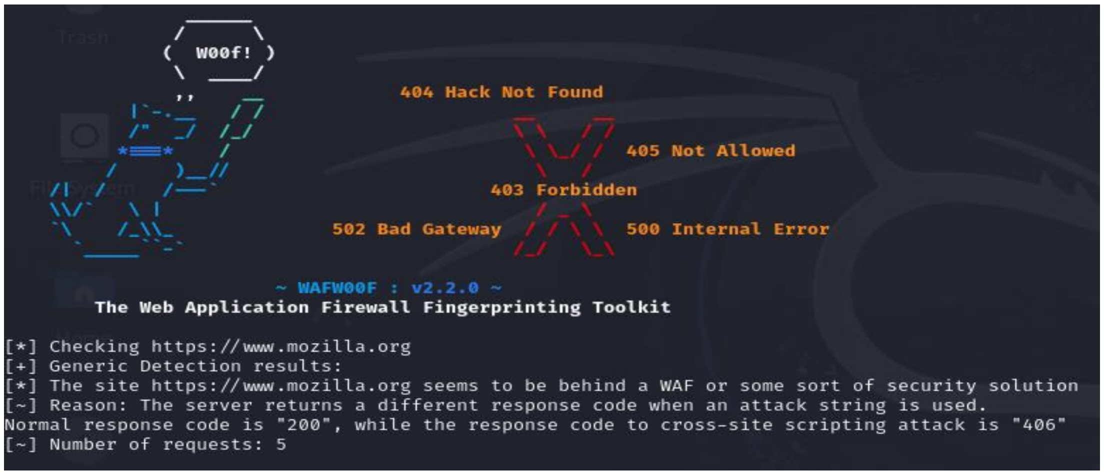
I found all the subdomains related to the target domain using Amass.



```
┌──(nelushi㉿kali)-[~]
└─$ amass enum -d mozilla.org
mozilla.org (FQDN) ⟶ ns_record ⟶ ns1-240.akam.net (FQDN)
mozilla.org (FQDN) ⟶ ns_record ⟶ ns4-64.akam.net (FQDN)
mozilla.org (FQDN) ⟶ ns_record ⟶ ns7-66.akam.net (FQDN)
mozilla.org (FQDN) ⟶ ns_record ⟶ ns5-65.akam.net (FQDN)
mozilla.org (FQDN) ⟶ mx_record ⟶ alt2.aspmx.l.google.com (FQDN)
mozilla.org (FQDN) ⟶ mx_record ⟶ aspmx.l.google.com (FQDN)
mozilla.org (FQDN) ⟶ mx_record ⟶ alt1.aspmx.l.google.com (FQDN)
mozilla.org (FQDN) ⟶ mx_record ⟶ aspmx3.googlemail.com (FQDN)
ns1-240.akam.net (FQDN) ⟶ a_record ⟶ 193.108.91.240 (IPAddress)
ns1-240.akam.net (FQDN) ⟶ aaaa_record ⟶ 2600:1401:2::f0 (IPAddress)
blog.mozilla.org (FQDN) ⟶ cname_record ⟶ wp.wpenginepowered.com (FQDN)
riskheatmap.security.mozilla.org (FQDN) ⟶ a_record ⟶ 35.164.214.130 (IPAddress)
riskheatmap.security.mozilla.org (FQDN) ⟶ a_record ⟶ 52.88.155.98 (IPAddress)
riskheatmap.security.mozilla.org (FQDN) ⟶ a_record ⟶ 35.82.219.168 (IPAddress)
developer.mozilla.org (FQDN) ⟶ cname_record ⟶ mdn.prod.mdn.prod.webservices.mozgcp.net (FQDN)
addons.mozilla.org (FQDN) ⟶ a_record ⟶ 151.101.129.91 (IPAddress)
addons.mozilla.org (FQDN) ⟶ a_record ⟶ 151.101.193.91 (IPAddress)
addons.mozilla.org (FQDN) ⟶ a_record ⟶ 151.101.1.91 (IPAddress)
addons.mozilla.org (FQDN) ⟶ a_record ⟶ 151.101.65.91 (IPAddress)
addons.mozilla.org (FQDN) ⟶ aaaa_record ⟶ 2a04:4e42::347 (IPAddress)
```

Wafw00f – Firewall Detection

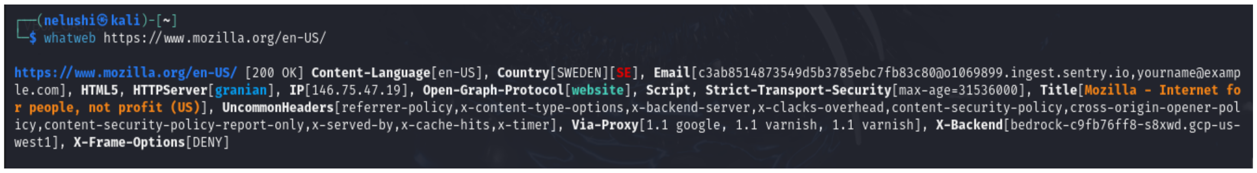
Command used – wafw00f https://www.mozilla.org/en-US/



```
                ______
               /      \
              (  W00f! )
               \  ____/
               ,,    __            404 Hack Not Found
           |`-.__   / /                      __     __
           /"  _/  /_/                       \ \   / /
          *===*    /                          \ \_/ /  405 Not Allowed
         /     )__//                           \   /
    /|  /     /---`                         403 Forbidden
    \\/`   \ |                                 / \
    `\    /_\\_              502 Bad Gateway  / / \ \  500 Internal Error
      `_____``-`                              /_/   \_\

                        ~ WAFW00F : v2.2.0 ~
        The Web Application Firewall Fingerprinting Toolkit

[*] Checking https://www.mozilla.org
[+] Generic Detection results:
[*] The site https://www.mozilla.org seems to be behind a WAF or some sort of security solution
[~] Reason: The server returns a different response code when an attack string is used.
Normal response code is "200", while the response code to cross-site scripting attack is "406"
[~] Number of requests: 5
```

Whatweb – to identify the technologies used by the site.

Commans used – whatweb https://www.mozilla.org/en-US/



```
┌──(nelushi㉿kali)-[~]
└─$ whatweb https://www.mozilla.org/en-US/

https://www.mozilla.org/en-US/ [200 OK] Content-Language[en-US], Country[SWEDEN][SE], Email[c3ab8514873549d5b3785ebc7fb83c80@o1069899.ingest.sentry.io,yourname@examp
le.com], HTML5, HTTPServer[granian], IP[146.75.47.19], Open-Graph-Protocol[website], Script, Strict-Transport-Security[max-age=31536000], Title[Mozilla - Internet fo
r people, not profit (US)], UncommonHeaders[referrer-policy,x-content-type-options,x-backend-server,x-clacks-overhead,content-security-policy,cross-origin-opener-pol
icy,content-security-policy-report-only,x-served-by,x-cache-hits,x-timer], Via-Proxy[1.1 google, 1.1 varnish, 1.1 varnish], X-Backend[bedrock-c9fb76ff8-s8xwd.gcp-us-
west1], X-Frame-Options[DENY]
```

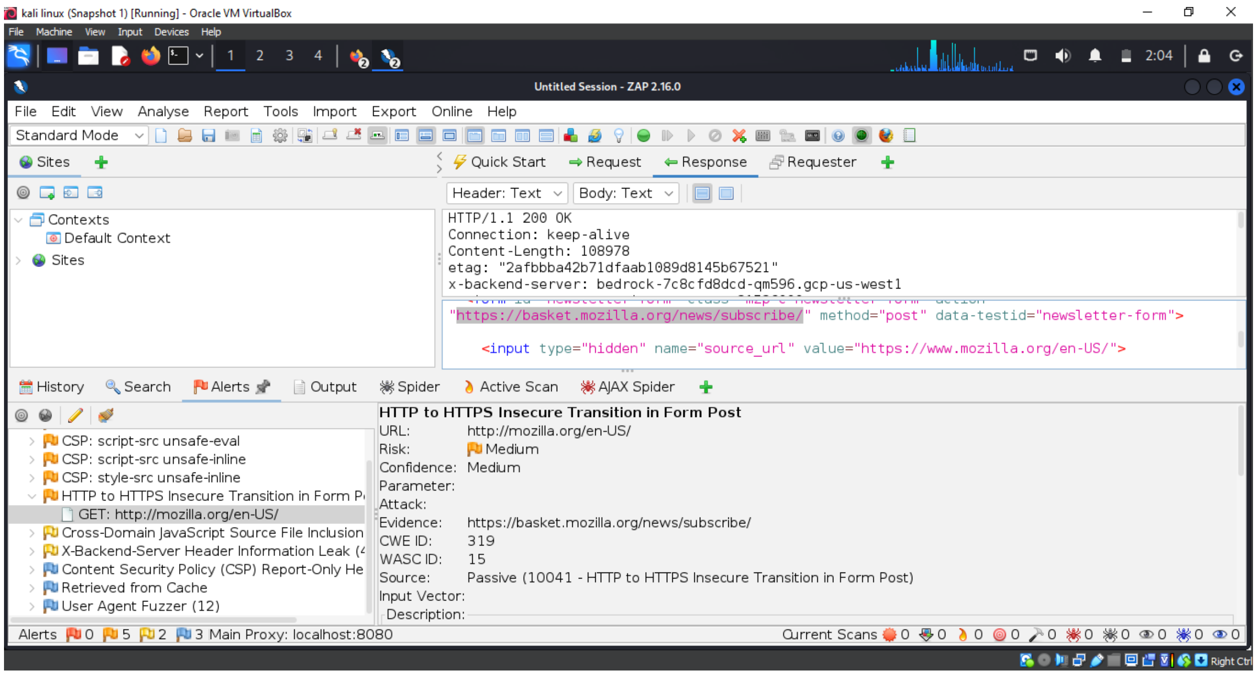
Vulnerability 01

Domain

https://basket.mozilla.org/news/subscribe/

Vulnerability title

HTTP to HTTPS Insecure Transition in Form Post



```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 108978
etag: "2afbbba42b71dfaab1089d8145b67521"
x-backend-server: bedrock-7c8cfd8dcd-qm596.gcp-us-west1
```

```
"https://basket.mozilla.org/news/subscribe/" method="post" data-testid="newsletter-form">

   <input type="hidden" name="source_url" value="https://www.mozilla.org/en-US/">
```

CSP: script-src unsafe-eval
CSP: script-src unsafe-inline
CSP: style-src unsafe-inline
HTTP to HTTPS Insecure Transition in Form P
GET: http://mozilla.org/en-US/
Cross-Domain JavaScript Source File Inclusion
X-Backend-Server Header Information Leak (4
Content Security Policy (CSP) Report-Only He
Retrieved from Cache
User Agent Fuzzer (12)

HTTP to HTTPS Insecure Transition in Form Post

URL:	http://mozilla.org/en-US/
Risk:	Medium
Confidence:	Medium
Parameter:	
Attack:	
Evidence:	https://basket.mozilla.org/news/subscribe/
CWE ID:	319
WASC ID:	15
Source:	Passive (10041 - HTTP to HTTPS Insecure Transition in Form Post)
Input Vector:	
Description:	

Alerts 0 5 2 3 Main Proxy: localhost:8080 Current Scans 0 0 0 0 0 0 0 0 0 0

Vulnerability description

The "HTTP to HTTPS Insecure Transition in Form Post" vulnerability exists when a website loads via HTTP and contains a form designed to post information to a HTTPS endpoint.

The form transition from to an HTTP to HTTPS endpoint is dangerous because the initial page is unsecure, opening the door for several types of Man-in-the-Middle (MitM) attacks.

Why is this a Security Risk?

MitM Attack Possibility: Because the page loaded over HTTP, the content of the page could be altered by an attacker. In this case, the real/secure HTTPS form could be modified or spoofed, meaning a user could be led into submitting their username, password or sensitive information to another site.

Trust Risk: Modern browsers will alert users of an insecure form submission reducing the level of trust a user has in the page.

Session Cookies Hijack: If session cookies are not securely protected, an attacker could steal and transfer them to access a user's account.

How does this work?

The security check looks for HTTP pages that contain forms submitting information to an HTTPS endpoint. Since the page is loaded via HTTP, an attacker can overwrite the original form with their own form prior to the user submitting their information.

Affected Components

- Web Pages with Forms (Frontend)

  Login Pages (login.html, index.php)

  Registration Pages (signup.html)

  Password Reset Pages (reset-password.html)

  Payment Pages (checkout.html)

If these pages load over HTTP but submit data to the HTTPS connection, an adversary might inject a malicious form or modify the request.

- Web Server Configuration (Backend)

Apache, Nginx, IIS - If the web server is misconfigured, the server might allow an HTTP page, thus allowing someone to view the page, but only secure a form submission.

Missing a Redirect from HTTP to HTTPS - If someone is able to access the website via an HTTP connection, that's an issue.

No HSTS (HTTP Strict Transport Security) - Without HSTS, attackers can downgrade the connection to HTTP using an SSL stripping connection.

- Cookies and Sessions

Session Cookies - If cookies do not have the Secure Flag or the HttpOnly flag, they can be transmitted via HTTP, allowing the attacker to steal the cookies.

CSRF Tokens - If CSRF tokens are generated from an HTTP page, then adversaries can steal and leverage those tokens in attacks.

Impact assessment

Severity – Medium

- Man-in-the-Middle Attacks

Attackers can capture the HTTP page before submitting the form.

A forgery form can replace an original form secured by HTTPS, resulting in credential information theft.

- Session Hijacking & Account Takeover

If session cookies are sent over HTTP, attackers can take the cookie and reuse it to gain unauthorized access.

- Phishing & Spoofing Attacks

Users could unknowingly submit input to a malicious version of the forms the attacker injected.

This could lead to identity theft, fraud financial transactions, and reputational damage.

- User Trust & Browser Warnings

Modern browsers warn users of insecure submissions, causing damage to a consumer's trust.

Potential loss of customers as a result of security concerns.

Overall Risk: Critical for pages related to authentication and payment.

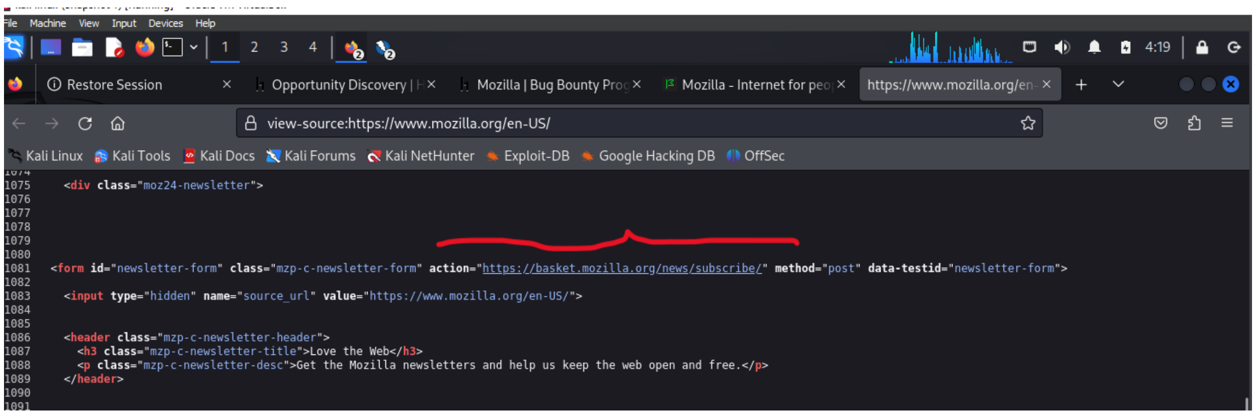
Steps to reproduce with Proof of Concept (poc)

1. First, I tried to identify an HTTP Page with a Secure Form Submission

I opened the target website in a browser using HTTP and checked the source code to find a form like,

Ex: <form action="https://example.com/login" method="POST">

I confirmed that the **page loads over HTTP** but the **form submits to HTTPS**.

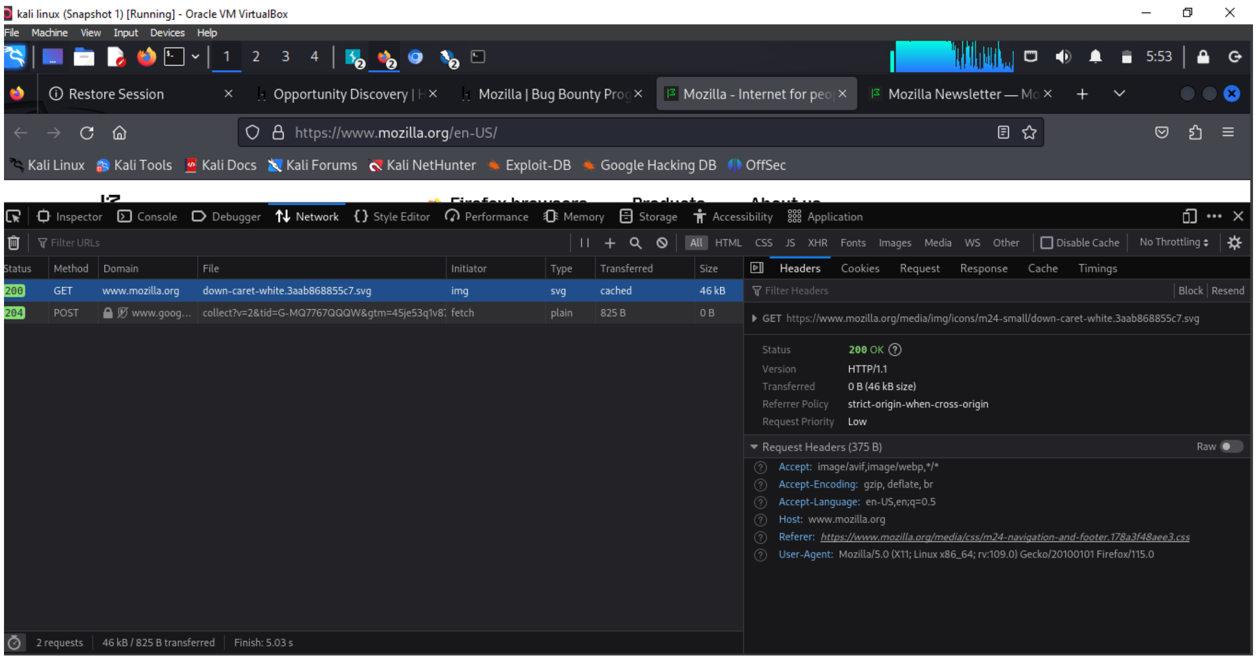


view-source:https://www.mozilla.org/en-US/

```
  <div class="moz24-newsletter">




<form id="newsletter-form" class="mzp-c-newsletter-form" action="https://basket.mozilla.org/news/subscribe/" method="post" data-testid="newsletter-form">

  <input type="hidden" name="source_url" value="https://www.mozilla.org/en-US/">


  <header class="mzp-c-newsletter-header">
    <h3 class="mzp-c-newsletter-title">Love the Web</h3>
    <p class="mzp-c-newsletter-desc">Get the Mozilla newsletters and help us keep the web open and free.</p>
  </header>
```

2. Then, I checked if there are any security headers like HSTS by opening the developer tools, network tab.

The header is not set.



kali linux (Snapshot 1) [Running] - Oracle VM VirtualBox

https://www.mozilla.org/en-US/

```
Status  Method  Domain          File                                       Initiator  Type   Transferred  Size
200     GET     www.mozilla.org down-caret-white.3aab868855c7.svg          img        svg    cached       46 kB
204     POST    www.goog...     collect?v=2&tid=G-MQ7767QQQW&gtm=45je53q1v8: fetch    plain  825 B        0 B

2 requests | 46 kB / 825 B transferred | Finish: 5.03 s
```

```
GET https://www.mozilla.org/media/img/icons/m24-small/down-caret-white.3aab868855c7.svg

Status            200 OK
Version           HTTP/1.1
Transferred       0 B (46 kB size)
Referrer Policy   strict-origin-when-cross-origin
Request Priority  Low

Request Headers (375 B)
Accept: image/avif,image/webp,*/*
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5
Host: www.mozilla.org
Referer: https://www.mozilla.org/media/css/m24-navigation-and-footer.178a3f48aee3.css
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
```

The site is vulnerable to downgrading attacks, in which an attacker will force the user to send data over HTTP instead of HTTPS.

Sensitive HTTP data can be stolen, tampered with, or intercepted by attackers within the same network.

What is the importance of HSTS?

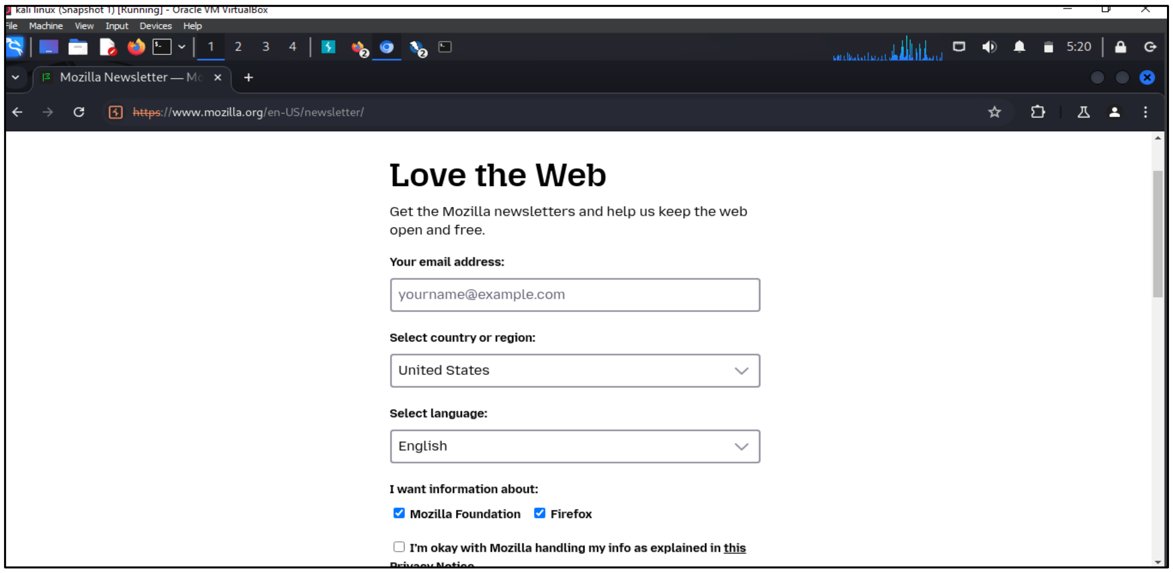
In HSTS, the browser ensures that ALL HTTP requests are automatically converted into HTTPS requests to mitigate man-in-the-middle attacks.

Without HSTS, it is possible for an attacker to intercept the user's original unencrypted HTTP request, where they can either hijack the session or inject unwanted malicious content into the session.

3. Finally, I tried to intercept the HTTP request using Burp Suite.

I opened burp suite and clicked “interception on” and filled the form.

After submitting the form, I checked the request in burp and sent it to burp repeater,


kali linux (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
5:20
Mozilla Newsletter — Mc
~~https~~://www.mozilla.org/en-US/newsletter/

Love the Web

Get the Mozilla newsletters and help us keep the web open and free.

Your email address:

yourname@example.com

Select country or region:

United States

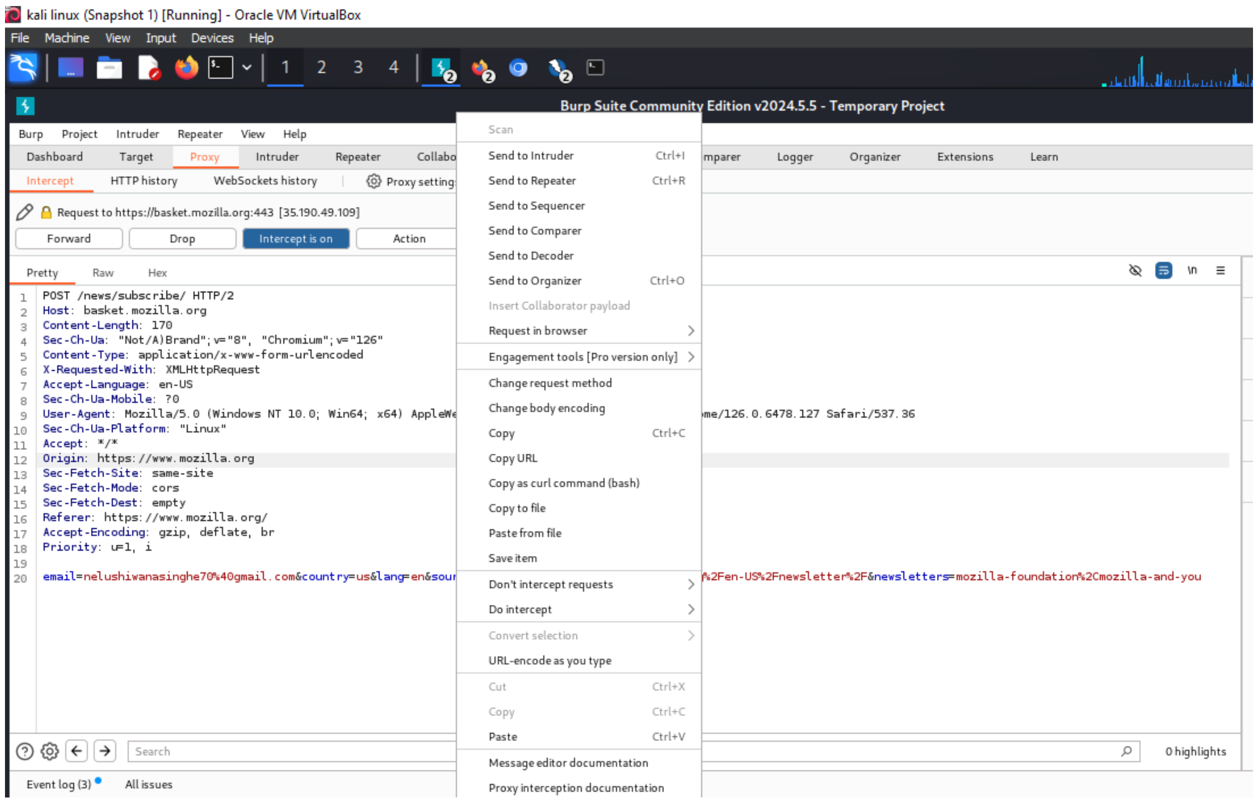
Select language:

English

I want information about:

[✓ Mozilla Foundation] [✓ Firefox]

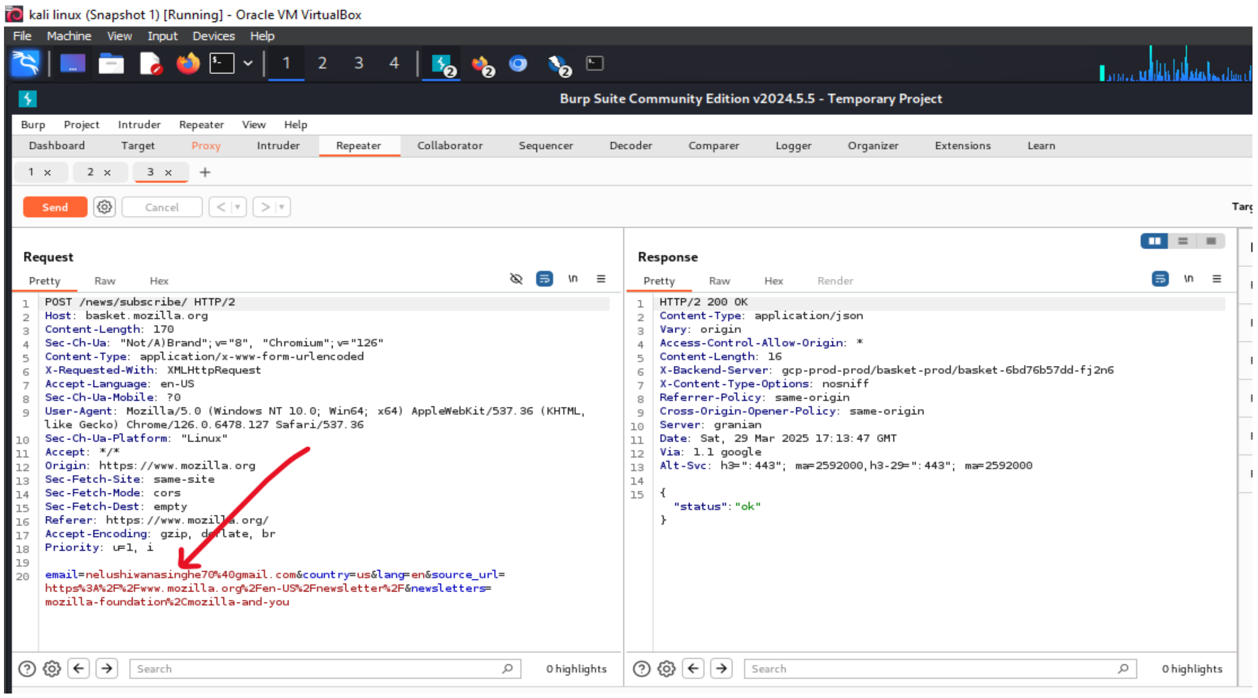
☐ I'm okay with Mozilla handling my info as explained in this Privacy Notice


kali linux (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2024.5.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collabo… …mparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy setting…
Forward Drop Intercept is on Action
Pretty Raw Hex
Scan
Send to Intruder Ctrl+I
Send to Repeater Ctrl+R
Send to Sequencer
Send to Comparer
Send to Decoder
Send to Organizer Ctrl+O
Insert Collaborator payload
Request in browser
Engagement tools [Pro version only]
Change request method
Change body encoding
Copy Ctrl+C
Copy URL
Copy as curl command (bash)
Copy to file
Paste from file
Save item
Don't intercept requests
Do intercept
Convert selection
URL-encode as you type
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Message editor documentation
Proxy interception documentation
Search
0 highlights
Event log (3) All issues

Request to https://basket.mozilla.org:443 [35.190.49.109]

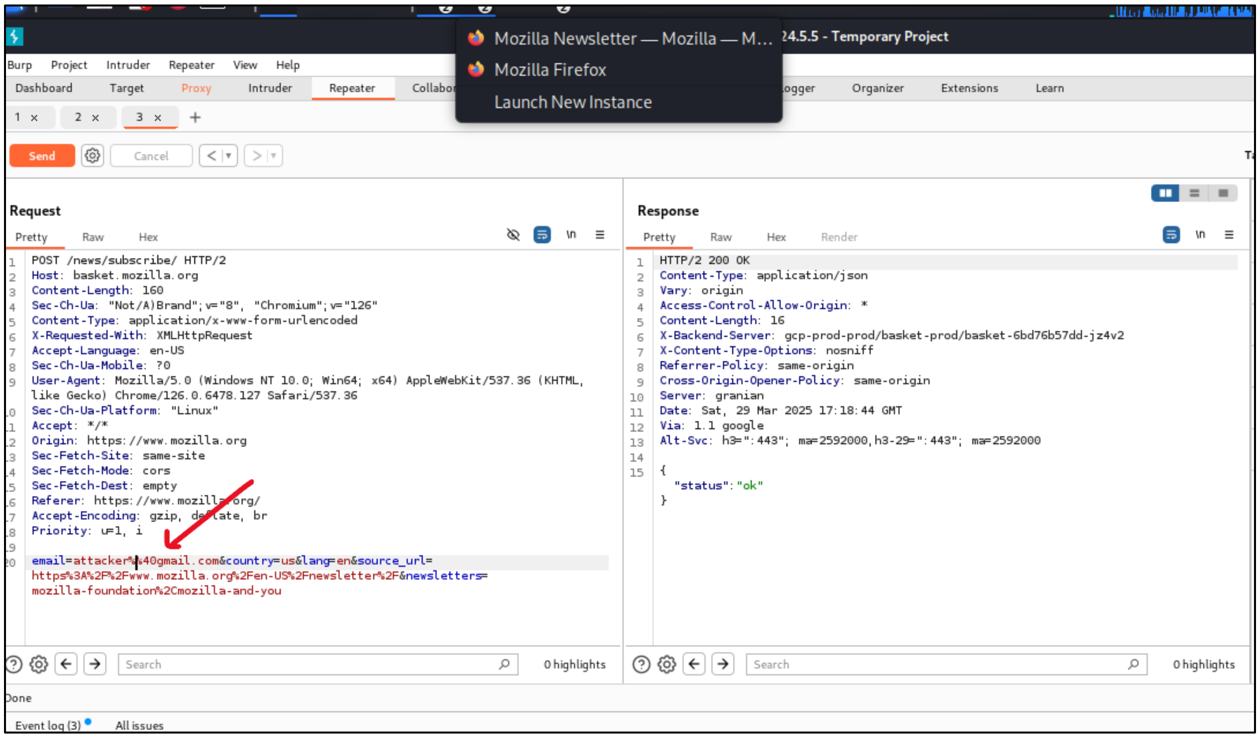
```
POST /news/subscribe/ HTTP/2
Host: basket.mozilla.org
Content-Length: 170
Sec-Ch-Ua: "Not/A)Brand";v="8", "Chromium";v="126"
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Accept-Language: en-US
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWe        me/126.0.6478.127 Safari/537.36
Sec-Ch-Ua-Platform: "Linux"
Accept: */*
Origin: https://www.mozilla.org
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.mozilla.org/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i

email=nelushiwanasinghe70%40gmail.com&country=us&lang=en&sour        %2Fen-US%2Fnewsletter%2F&newsletters=mozilla-foundation%2Cmozilla-and-you
```

In there, I started to modify the email address of mine to attackers@gmail.com to see the response of the server.


kali linux (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2024.5.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
1 × 2 × 3 × +
Send Cancel
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
Search
0 highlights

```
POST /news/subscribe/ HTTP/2
Host: basket.mozilla.org
Content-Length: 170
Sec-Ch-Ua: "Not/A)Brand";v="8", "Chromium";v="126"
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Accept-Language: en-US
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/126.0.6478.127 Safari/537.36
Sec-Ch-Ua-Platform: "Linux"
Accept: */*
Origin: https://www.mozilla.org
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.mozilla.org/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i

email=nelushiwanasinghe70%40gmail.com&country=us&lang=en&source_url=
https%3A%2F%2Fwww.mozilla.org%2Fen-US%2Fnewsletter%2F&newsletters=
mozilla-foundation%2Cmozilla-and-you
```

```
HTTP/2 200 OK
Content-Type: application/json
Vary: origin
Access-Control-Allow-Origin: *
Content-Length: 16
X-Backend-Server: gcp-prod-prod/basket-prod/basket-6bd76b57dd-fj2n6
X-Content-Type-Options: nosniff
Referrer-Policy: same-origin
Cross-Origin-Opener-Policy: same-origin
Server: granian
Date: Sat, 29 Mar 2025 17:13:47 GMT
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{
  "status":"ok"
}
```


Mozilla Newsletter — Mozilla — M...
Mozilla Firefox
Launch New Instance
24.5.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collabo Logger Organizer Extensions Learn
Send Cancel
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
0 highlights
Done
Event log (3) All issues

```
POST /news/subscribe/ HTTP/2
Host: basket.mozilla.org
Content-Length: 160
Sec-Ch-Ua: "Not/A)Brand";v="8", "Chromium";v="126"
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Accept-Language: en-US
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/126.0.6478.127 Safari/537.36
Sec-Ch-Ua-Platform: "Linux"
Accept: */*
Origin: https://www.mozilla.org
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://www.mozilla.org/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i

email=attacker%40gmail.com&country=us&lang=en&source_url=
https%3A%2F%2Fwww.mozilla.org%2Fen-US%2Fnewsletter%2F&newsletters=
mozilla-foundation%2Cmozilla-and-you
```

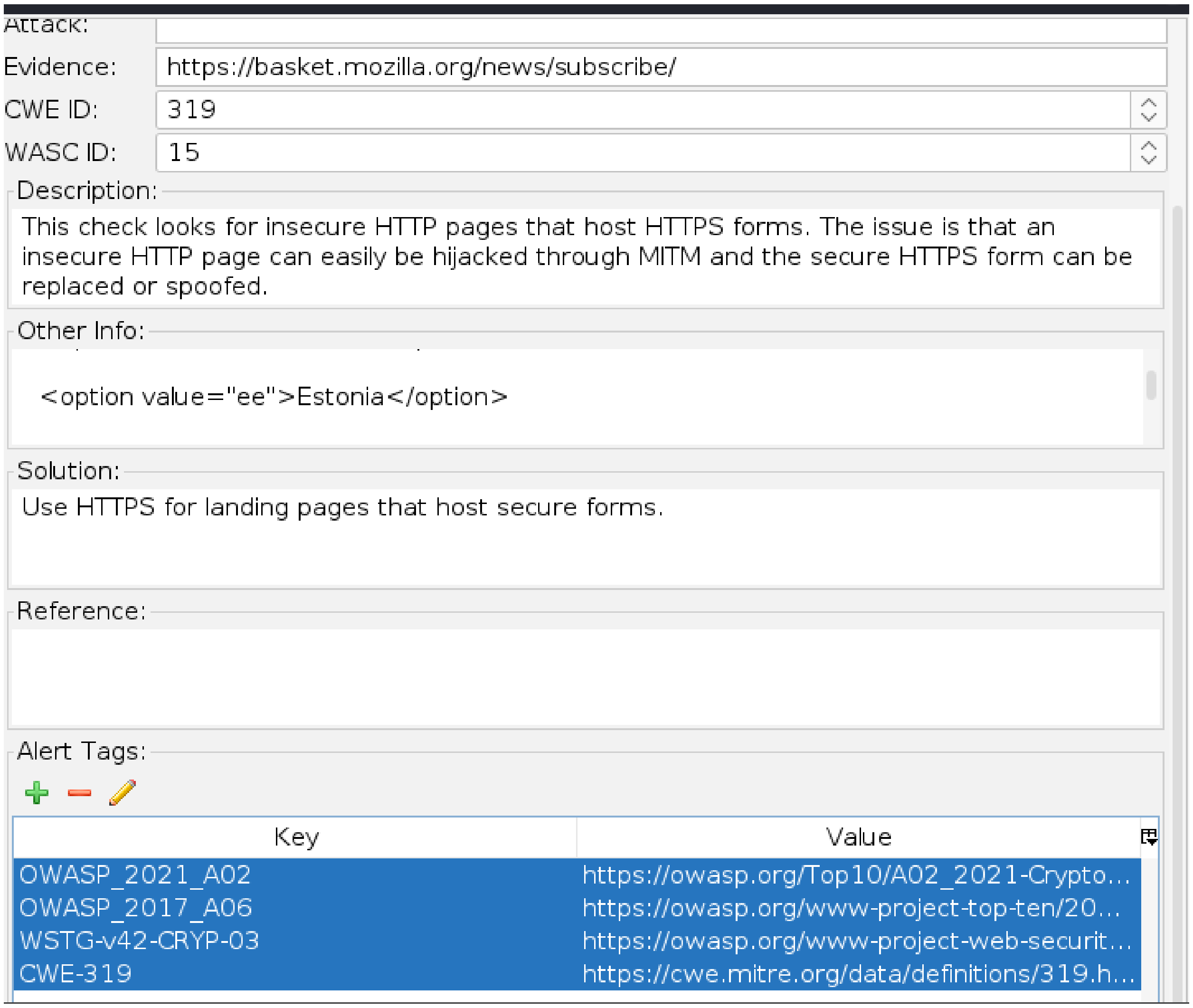
```
HTTP/2 200 OK
Content-Type: application/json
Vary: origin
Access-Control-Allow-Origin: *
Content-Length: 16
X-Backend-Server: gcp-prod-prod/basket-prod/basket-6bd76b57dd-jz4v2
X-Content-Type-Options: nosniff
Referrer-Policy: same-origin
Cross-Origin-Opener-Policy: same-origin
Server: granian
Date: Sat, 29 Mar 2025 17:18:44 GMT
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{
  "status":"ok"
}
```

The response returns a **successful status code** (**200 OK**) in the response. This means that, I can intercept user details and still get a successful status.

In the case that the form is using HTTP rather than HTTPS, the attacker may intercept the request and alter it while in transit. This could lead to potential man-in-the-middle attacks or modification of the data.

Keys and values of the alert

OWASP_2021_A02	https://owasp.org/Top10/A02_2021-Cryptographic_Failures/
OWASP_2017_A06	https://owasp.org/www-project-top-ten/2017/A6_2017-Security_Misconfiguration.html



Attack:

Evidence: https://basket.mozilla.org/news/subscribe/

CWE ID: 319

WASC ID: 15

Description:
This check looks for insecure HTTP pages that host HTTPS forms. The issue is that an insecure HTTP page can easily be hijacked through MITM and the secure HTTPS form can be replaced or spoofed.

Other Info:
```
  <option value="ee">Estonia</option>
```

Solution:
Use HTTPS for landing pages that host secure forms.

Reference:

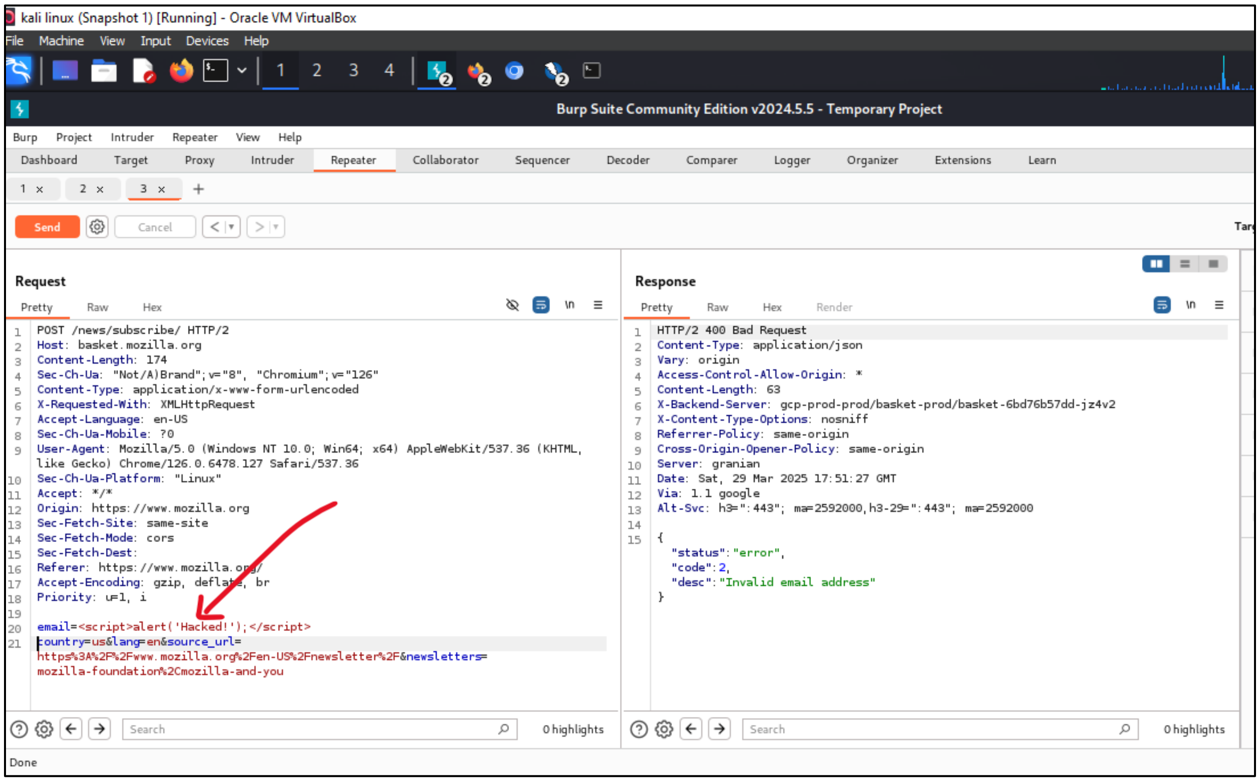
Alert Tags:

Key	Value
OWASP_2021_A02	https://owasp.org/Top10/A02_2021-Crypto…
OWASP_2017_A06	https://owasp.org/www-project-top-ten/20…
WSTG-v42-CRYP-03	https://owasp.org/www-project-web-securit…
CWE-319	https://cwe.mitre.org/data/definitions/319.h…

Therefore, this vulnerability is rated as **OWASP 2021 A02: Cryptographic Failure** because it is transmitting sensitive information (username, password, email, etc.) over HTTP rather than HTTPS, which is vulnerable to Man-in-the-Middle attacks, an attacker can intercept the request and modify it. It is also under **OWASP 2017 A06: Security Misconfiguration** as requiring HTTPS, missing HSTS headers, and not serving secure form submission, all indicate improper security configurations, with the application still being exposed to intercept data tampering.

4. Additionally, I tried to inject a malicious javascript code to the email field to see if the site is vulnerable to XSS attack, but it generated an error, meaning that site is not vulnerable to basic XSS in the email field, because it's blocking the malicious input at this stage.

It provided a 400 bad request status code.


kali linux (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Burp Suite Community Edition v2024.5.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send
Cancel
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
Search
0 highlights
Done

```
POST /news/subscribe/ HTTP/2
Host: basket.mozilla.org
Content-Length: 174
Sec-Ch-Ua: "Not/A)Brand";v="8", "Chromium";v="126"
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
Accept-Language: en-US
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/126.0.6478.127 Safari/537.36
Sec-Ch-Ua-Platform: "Linux"
Accept: */*
Origin: https://www.mozilla.org
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest:
Referer: https://www.mozilla.org/
Accept-Encoding: gzip, deflate, br
Priority: u=1, i

email=<script>alert('Hacked!');</script>
country=us&lang=en&source_url=
https%3A%2F%2Fwww.mozilla.org%2Fen-US%2Fnewsletter%2F&newsletters=
mozilla-foundation%2Cmozilla-and-you
```

```
HTTP/2 400 Bad Request
Content-Type: application/json
Vary: origin
Access-Control-Allow-Origin: *
Content-Length: 63
X-Backend-Server: gcp-prod-prod/basket-prod/basket-6bd76b57dd-jz4v2
X-Content-Type-Options: nosniff
Referrer-Policy: same-origin
Cross-Origin-Opener-Policy: same-origin
Server: granian
Date: Sat, 29 Mar 2025 17:51:27 GMT
Via: 1.1 google
Alt-Svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000

{
  "status":"error",
  "code":2,
  "desc":"Invalid email address"
}
```

Proposed mitigation or fix

1. Use HTTPS to serve all pages (use SSL/TLS).

2. Enforce HTTP-to-HTTPS redirection using **.htaccess, nginx.conf,** or **IIS settings.**

3. Enable HSTS (HTTP Strict Transport Security) to block access via HTTP
   Strict-Transport-Security: max-age=31536000; includeSubDomains; preload

4. Use Secure Cookies (**Secure, HttpOnly, SameSite=Strict**).

5. Monitor logs to look for mixed-content warnings and HTTP submissions.

Addressing this vulnerability mitigates possible credential theft via MITM.