# Web Hacking

S2Day9Web.md



by Nathan Hailu

Recall

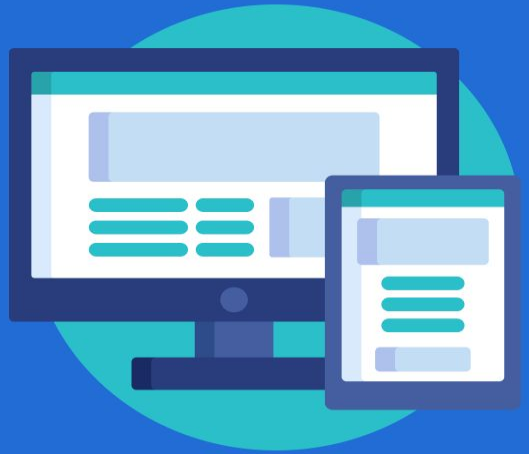
# Last Class TOPICS

# Topics

- What is Web Application
- What is Web Hacking
- URL and URI
- How does the web works?
- Web information gathering
- Burp Suite
- OWASP top 10
- Web vulns

# What is Web Application?

- A web application is a program or software that runs on a web browser to perform specific tasks.
- These web applications are made with programming languages like HTML,CSS,JS(node,react,angular),Python(Django,Flask)
- WebSites Consists 2 parts.
    - Front End / Client side /
        - Front End is the Front page/ the page which we access, the graphical part.
        - There are specific Programming languages and Frameworks to build front end of a website.
            - Example: HTML,CSS,JS(react)
        - A person who develop Front end is called “Front End Developer”
    - Back End / Server Side /
        - Back End is another side of a web which users don't have interaction with it, but their requests will be sent to the server and used to fetch the data from server and hands to the front page/front end/ of the websites.
        - Also here there are Specific Programming languages
            - Example: JS(node.js),Python(Django,Flask),PHP,SQL
        - A person who develop Back end is called “Back End Developer”
- A person who develop Both Front and Back end is called “Full Stack Developer”

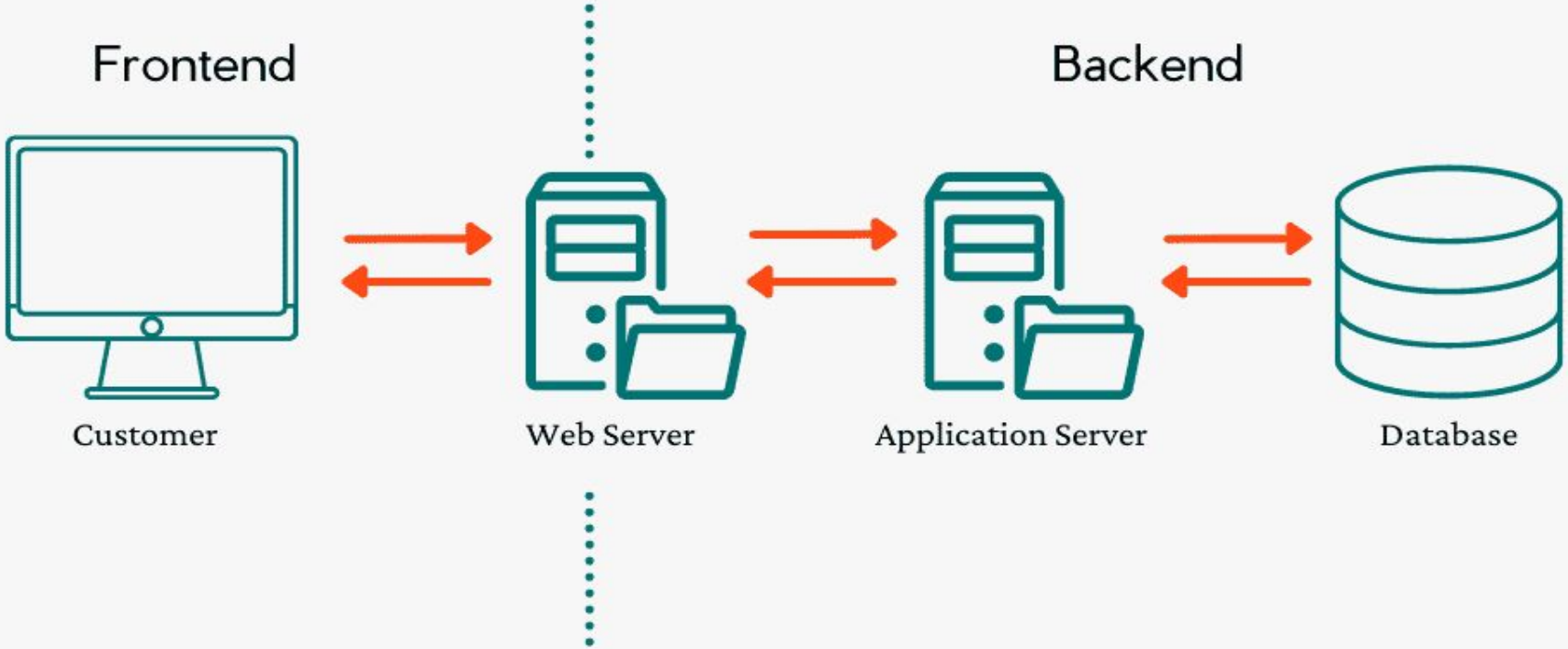


FRONT-END

BACK-END

ELEVATE X

# Frontend vs. Backend



# HTML /HyperText Markup language/

- HTML is the standard markup language for creating Web pages.
  - Markup language refers to **a text-encoding system consisting of symbols inserted in a text document to control its structure, formatting, or the relationship between its parts**.
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML have elements called Tags.
  - Tags are special words that have < > on their beginning and ending
  - Example: <a> , <html> , <body> , <head> …. `<tagname> Content goes here... </tagname>`
- HTML languages start with a HTML tag
- Those tags have opening and closing tags.
  -  … 
    - The text between opening and closing tag is called “**innerText**”
  -   …  
    - The tags inside a tag is called “**innerHTML**”
- But This doesn't mean all HTML tags need Closing tag.
  - Example: <img>  does not have closing tag.

# A Simple HTML Document

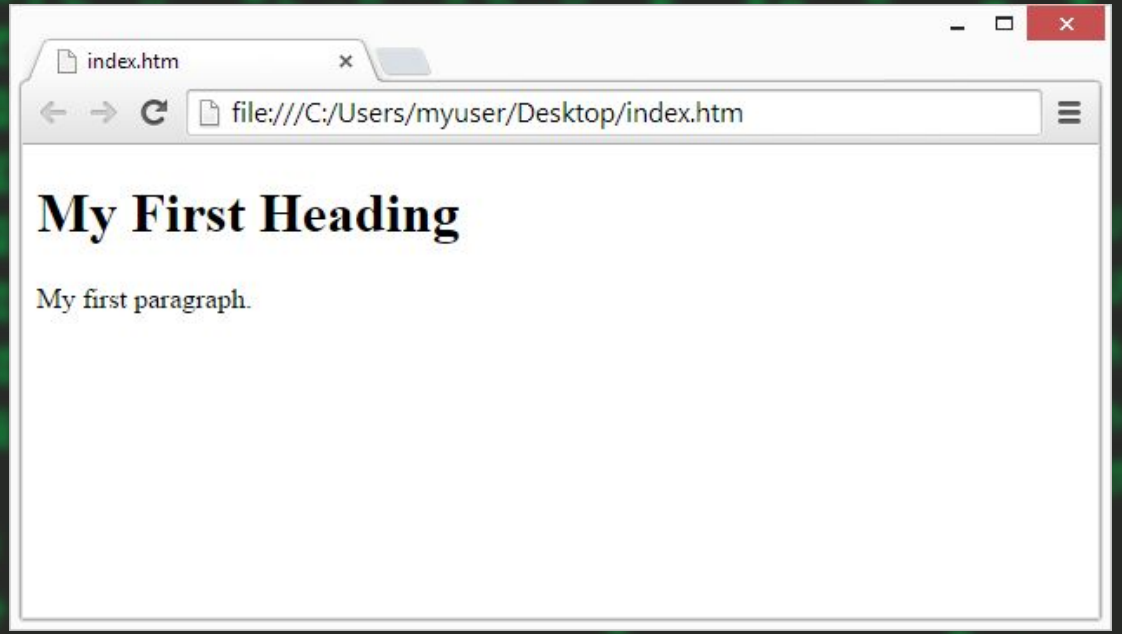
- The html tag contains 2 elements
  - Head tag
    - Contain elements those does not appear on the page.
      - Title and meta info's
  - Body tag
    - is a container for all the visible contents
      - headings, paragraphs, images, hyperlinks, tables, lists, etc.
- There are heading tags
  - <h1>,<h2>,<h3>..<h5>
- Paragraph <p>
- Anchor tags <a>  for linking
- Image tags <img>
- Break line(new line)  <br>
- Division 
  - Used to do container and hold some tags inside.

```
<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>
```

```
1  <html>
2      <head>
3          <title>Page Title</title>
4      </head>
5      <body>
6
7          <h1>My First Heading</h1>
           <p>My first paragraph.</p>

       </body>
   </html>
```

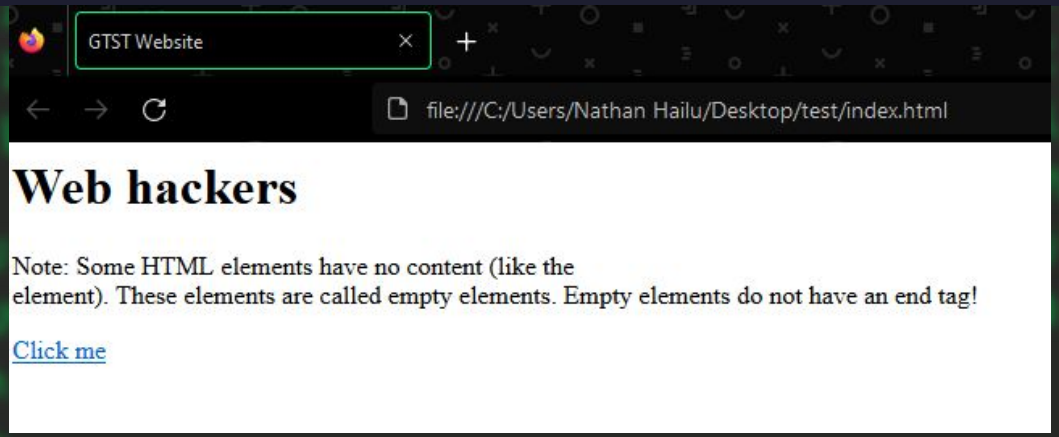
# Web browsers

- The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly.
- A browser does not display the HTML tags, but uses them to determine how to display the document:



# demo

```
<html>
    <head>
        <title>GTST Website</title>
    </head>
    <body>

        <h1>Web hackers</h1>
        <p>Note: Some HTML elements have no content (like the <br> element). These elements are called empty elements. Empty elements do not have an end tag
        <a href="https://google.com"> Click me </a>

    </body>
</html>
```



# What is web hacking

- Web hacking refers to exploitation of applications via HTTP which can be done by manipulating the application via its graphical web interface.
- It is The another Big Scope of Cyber Security
- Always Hacking Steps are same on all type of attacks.
  - Information gathering
  - Scanning
  - Exploiting
  - .
  - .
- But to hack anything you have to know how the thing works.

SO

# How do websites work?

# URL and URI

- **URI identifies:** a resource and differentiates it from others by using a name, location, or both**.**
- **URL identifies:** the web address or location of a unique resource.
- URI contains components like a scheme, authority, path, and query.
- URL has similar components to a URI, but its authority consists of a domain name and port.
- URL is part of URI
- A URI aims to identify a resource and differentiate it from other resources by using the name of the resource or location of the resource.
- A URL aims to find the location or address of a resource on the web.
    - An example of a URI can be ISBN 0-486-35557-4, https://www.javatpoint.com , www.javatpoint.com
    - An example of an URL is https://www.javatpoint.com.

# Parts of URL


A URL consists of five parts:

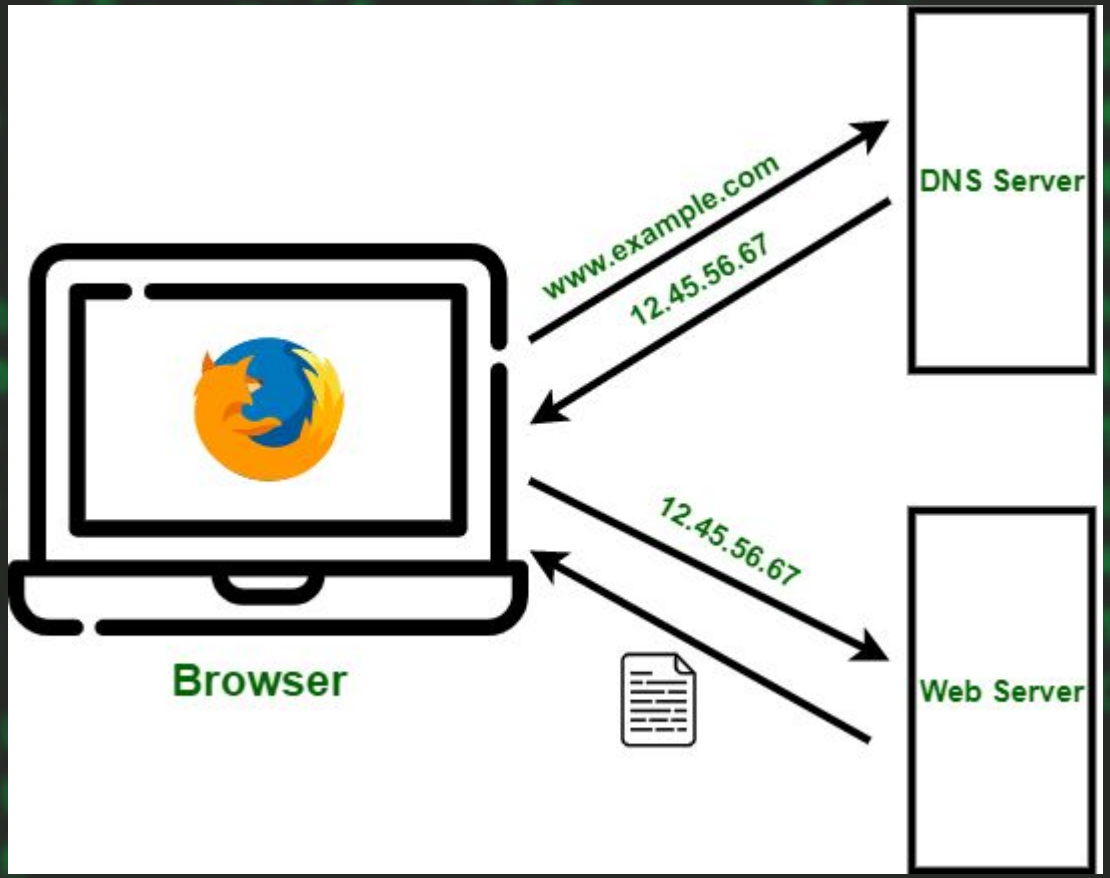
1. Scheme: tells web servers which protocol to use when it accesses a page on your website.
2. Subdomain:
   a. If your website is like a house, your subdomains are like specific rooms in that house.
   b. A subdomain in a URL indicates which particular page of your website the web browser should serve up.
   c. For instance, subdomains like “blog” or “offers” will provide your website’s blog page or offers page.
   d. Subdomains also bucket your website into its main content categories and shows Google and your visitors that there's more information on your site than just a homepage. ( meet, docs, google.com )
3. Top-level domain: specifies what type of entity your organization registers as on the internet.
   a. Generic Top level domain(gTLD): .gov .org .net
   b. Country code Top-level domain(ccTLD): .et .ru
4. Second-level domain: is the name of your website.
5. Subdirectory: also known as a subfolder,helps people understand which particular section of a webpage they’re on.

```
www.geeztech.com/technology/fields

/var/www/html/technology/fields
```

For apache servers example, but there are some applications who use Routes

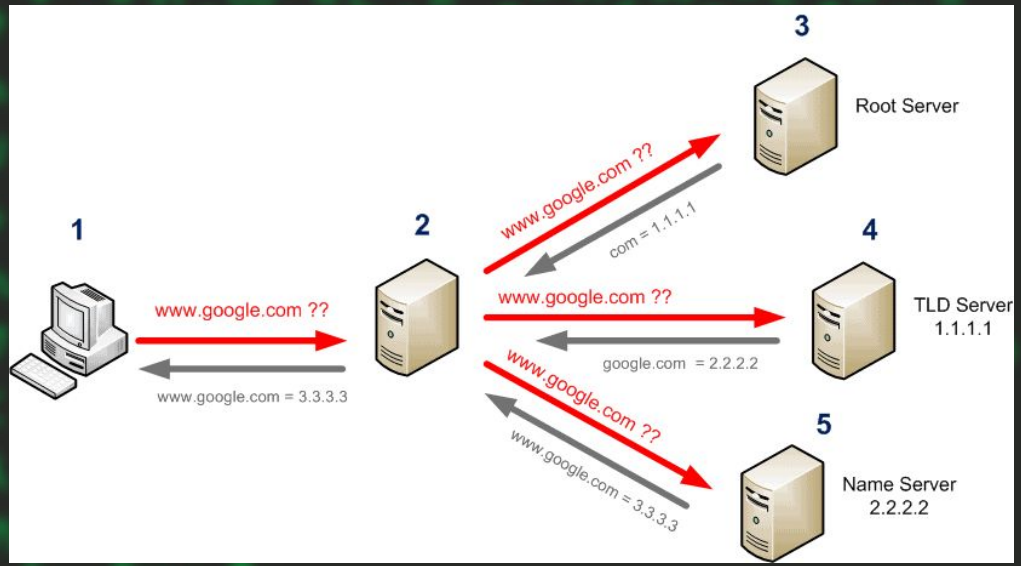
# DNS

- The Domain Name System (DNS) is the phonebook of the Internet.
- When users type domain names such as ‘google.com’ or ‘nytimes.com’ into web browsers, DNS is responsible for finding the correct IP address for those sites.
- Browsers then use those addresses to communicate with origin servers or CDN edge servers to access website information.
- This all happens thanks to DNS servers: machines dedicated to answering DNS queries.

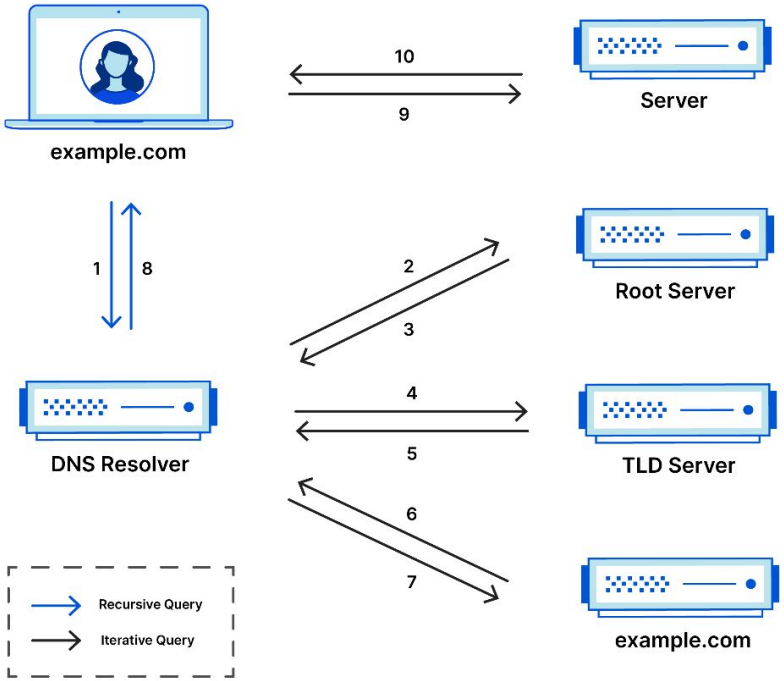


...

- The DNS request the goes out from our computer is called “DNS query”.
- there are four servers that work together to deliver an IP address to the client: recursive resolvers, root nameservers, TLD nameservers, and authoritative nameservers.



# Complete DNS Lookup and Webpage Query



# DNS Records

- DNS records (aka zone files) are instructions that live in authoritative DNS servers and provide information about a domain including what IP address is associated with that domain and how to handle requests for that domain.
- These records consist of a series of text files written in what is known as DNS syntax.
- To Access DNS Record on linux we can use tools like
  - Nslookup
  - Dig
  - Host

# Dig

```
❯ dig google.com

; <<>> DiG 9.19.17-1-Debian <<>> google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 20430
;; flags: qr rd ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             0       IN      A       142.250.181.78

;; Query time: 99 msec
;; SERVER: 172.24.64.1#53(172.24.64.1) (UDP)
;; WHEN: Sat Nov 25 19:33:59 EAT 2023
;; MSG SIZE  rcvd: 54
```

# Nslookup

```
❯ nslookup google.com
Server:         172.24.64.1
Address:        172.24.64.1#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.181.78
Name:   google.com
Address: 2a00:1450:4018:806::200e
```

# host

```
❯ host google.com
google.com has address 142.250.181.78
google.com has IPv6 address 2a00:1450:4018:806::200e
google.com mail is handled by 10 smtp.google.com.
smtp.google.com has IPv6 address 2a00:1450:400c:c02::1b
smtp.google.com has IPv6 address 2a00:1450:400c:c07::1a
smtp.google.com has IPv6 address 2a00:1450:400c:c07::1b
smtp.google.com has IPv6 address 2a00:1450:400c:c0c::1b
```

# Types of DNS records

There are Many DNS record Types but lets see some

# A Record ( Address )

- This is a Record on the server that holds IPv4 Address.

```
❯ nslookup -query=A google.com
Server:         172.24.64.1
Address:        172.24.64.1#53

Non-authoritative answer:
Name:   google.com
Address: 142.250.181.174
```

```
❯ host -t A google.com
google.com has address 142.250.181.174
```

```
❯ dig A google.com

; <<>> DiG 9.19.17-1-Debian <<>> A google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 8054
;; flags: qr rd ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             0       IN      A       142.250.181.174

;; Query time: 0 msec
;; SERVER: 172.24.64.1#53(172.24.64.1) (UDP)
;; WHEN: Sat Nov 25 19:38:37 EAT 2023
;; MSG SIZE  rcvd: 54
```

# AAAA Record (Quad Address)

- This holds the IPv6 Of the Domain

```
❯ nslookup -query=AAAA google.com
Server:         172.24.64.1
Address:        172.24.64.1#53

Non-authoritative answer:
Name:   google.com
Address: 2a00:1450:4018:806::200e
```

```
❯ host -t AAAA google.com
google.com has IPv6 address 2a00:1450:4018:806::200e
```

```
❯ dig AAAA google.com

; <<>> DiG 9.19.17-1-Debian <<>> AAAA google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39696
;; flags: qr rd ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                    IN      AAAA

;; ANSWER SECTION:
google.com.             0       IN      AAAA    2a00:1450:4018:806::200e

;; Query time: 9 msec
;; SERVER: 172.24.64.1#53(172.24.64.1) (UDP)
;; WHEN: Sat Nov 25 19:39:55 EAT 2023
;; MSG SIZE  rcvd: 66
```

# MX Record ( Mail Exchange )

- Directs mail to an email server.

```
❯ dig MX google.com

; <<>> DiG 9.19.17-1-Debian <<>> MX google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46855
;; flags: qr rd ad; QUERY: 1, ANSWER: 10, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                    IN      MX

;; ANSWER SECTION:
google.com.             0       IN      MX      10 smtp.google.com.
smtp.google.com.        0       IN      A       74.125.71.26
smtp.google.com.        0       IN      A       74.125.133.26
smtp.google.com.        0       IN      A       74.125.133.27
smtp.google.com.        0       IN      A       108.177.15.26
smtp.google.com.        0       IN      A       108.177.15.27
smtp.google.com.        0       IN      AAAA    2a00:1450:400c:c02::1a
smtp.google.com.        0       IN      AAAA    2a00:1450:400c:c07::1a
smtp.google.com.        0       IN      AAAA    2a00:1450:400c:c07::1b
smtp.google.com.        0       IN      AAAA    2a00:1450:400c:c0c::1a

;; Query time: 69 msec
;; SERVER: 172.24.64.1#53(172.24.64.1) (UDP)
;; WHEN: Sat Nov 25 19:44:42 EAT 2023
;; MSG SIZE  rcvd: 276
```

```
❯ host -t MX insa.gov.et
insa.gov.et mail is handled by 10 smtp2.insa.gov.et.
insa.gov.et mail is handled by 10 smtp.insa.gov.et.
```

# NS Record ( Name Server )

- Returns the DNS servers (nameservers) of the domain. Server Where all the DNS records are stored! 🦌

```
❯ host -t NS insa.gov.et
insa.gov.et name server ns2.telecom.net.et.
insa.gov.et name server ns1.telecom.net.et.
```

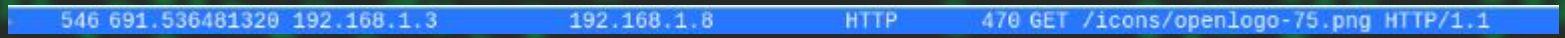
```
❯ dig NS google.com

; <<>> DiG 9.19.17-1-Debian <<>> NS google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21626
;; flags: qr rd ad; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;google.com.                    IN      NS

;; ANSWER SECTION:
google.com.             0       IN      NS      ns2.google.com.
google.com.             0       IN      NS      ns3.google.com.
google.com.             0       IN      NS      ns4.google.com.
google.com.             0       IN      NS      ns1.google.com.

;; Query time: 19 msec
;; SERVER: 172.24.64.1#53(172.24.64.1) (UDP)
;; WHEN: Sat Nov 25 19:46:43 EAT 2023
;; MSG SIZE  rcvd: 150
```

# And more...

- There are more Records like
  - TXT
  - CNAME
  - SOA
  - SRV
  - PTR
  - ...
- If you wanna Have Advanced knowledge on this Try to learn about “Zone Transfer”

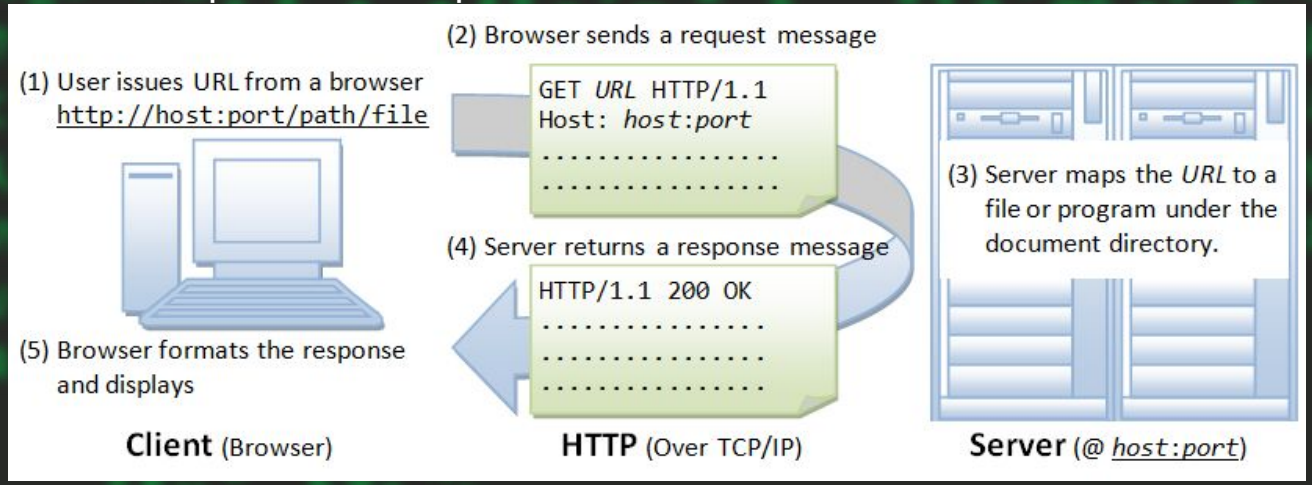
# How do websites work.

- When you access a website or click a link, it will send a HTTP request to the server and get the copy of the website files to the client this is called HTTP Response.



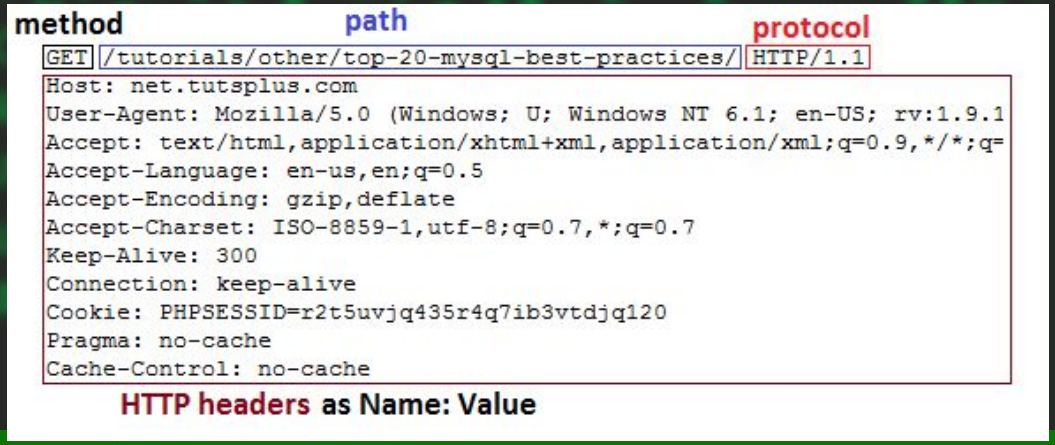
# HTTP request and response

- As the name suggests HTTP requests are a request which the browser sends to the server
- HTTP responses are a response from the server to the browser.
- The requests and Response are sent and received with a Header.



# HTTP Headers

- The **HTTP headers** are used to pass information between the clients and the server through the **request** and **response** header.
- All the headers are case-insensitive, headers fields are separated by colon, key-value pairs in clear-text string format.
- The end of the header section denoted by an empty field header(New line).



# Types of Headers

- **General Header:** This type of headers applied on Request and Response headers both but without affecting the database body.
- **Request Header:** This type of headers contains information about the fetched request by the client.
- **Response Header:** This type of headers contains the location of the source that has been requested by the client.
- **Entity Header:** This type of headers contains the information about the body of the resources like Content-length.

# Request Headers

```
GET /videos HTTP/1.1
Host: www.geeztech.com
Cookie: PHPSESSID=GA1.2.399324548.1669189742
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/109.0
Accept: text/html,application/xhtml+xml
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
```

- This is A header sent to the server.
- In Request Header There are different kind of headers
  - Example: GET, Host, Cookie,...

...

The 1st line Contains

- Request Method
- Path: The path where the file/folder is located
- Protocol Type: which HTTP protocol ( HTTP 1 , HTTP/1.1 , HTTP/2

The 2nd line

- Host: the website link

The 3rd line

- Cookie: used to check a user

The 4th line

- User-Agent: used to place the browser information

```
GET /videos HTTP/1.1
Host: www.geeztech.com
Cookie: PHPSESSID=GA1.2.399324548.1669189742
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win
Accept: text/html,application/xhtml+xml
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
```

# Response Header

- This is response from The server to the browser
- 1st line
  - HTTP: Tells the server Protocol
  - Status Code
- 2nd line
  - Date: Date of the response sent
- 3rd line
  - Content-Type: What type of content the server sent Encode type
- 4th line
  - Content-length: The number of the alphanumeric and spaces
- 5th line
  - Server: type of the webserver
- 6th line
  - This line is empty used to show that the headers ending. And begining of the body
- 7th…. Line
  - The html content

```
HTTP/2 200 OK
Date: Fri, 27 Jan 2023 12:15:36 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 329
Server: nginx

<!DOCTYPE html>
<html lang="en-US">
    <head>
        <meta charset="UTF-8"/>
        <title>Website Hacking</title>
    </head>
    <body>
        <p>hello there</p>
    </body>
</html>
```

# HTTP request methods

- The method **designates the type of request being made to the web server**.
- The most common types of request methods are GET and POST but there are many others, including HEAD, PUT, DELETE, CONNECT, and OPTIONS.
- GET and POST are widely supported while support for other methods is sometimes limited but expanding.

| Method | Description |
|---|---|
| GET | Request for resource from server |
| POST | Submit data to the server |
| HEAD | Same as GET but does not return the body |
| PUT | The data within the request must be stored at the URL supplied, replacing any existing data. |
| DELETE | Delete a resource |
| OPTIONS | Return the HTTP methods supported by the server |
| CONNECT | Client requests the HTTP proxy to forward a TCP connection to some destination. Used to create a TCP/IP tunnel for secure connections using HTTP proxies. |

# GET vs POST on url

GET Requests -

```
https://example.com/login?username=nathan&password=n@than2323
```

```
https://
example.com
/login
?
username=nathan
&
password=n@than2323
```

The Words “Username” and “Password”

Are called **Parameters**.

...

POST Request -

```
https://example.com/login
```

```
POST /login HTTP/1.1
Host: example.com
..
..
..

username=nathan&password=n@than2323
```

# HTTP Status Code

- The Status-Code element in a server response
- is a 3-digit integer where the first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role.
- There are 5 values for the first digit:
  - **1xx: Informational**
    - It means the request has been received and the process is continuing.
  - **2xx: Success**
    - It means the action was successfully received, understood, and accepted.
  - **3xx: Redirection**
    - It means further action must be taken in order to complete the request.
  - **4xx: Client Error**
    - It means the request contains incorrect syntax or cannot be fulfilled.
  - **5xx: Server Error**
    - It means the server failed to fulfill an apparently valid request.

# Some common codes

- 200 = request is Successful( OK )
- 301 = The requested page has moved to a new url . ( Moved Permanently )
- 302 = The requested page has moved temporarily to a new url .( Found ) => when there is redirection
- 400 = The server did not understand the request. (Bad Request)
- 401 = The requested page needs a username and a password. (Unauthorized)
- 403 = Access is forbidden to the requested page.(Forbidden)
- 404 = The server can not find the requested page.(Not Found)
- 405 = The method specified in the request is not allowed.(Method Not Allowed)
- 500 = The request was not completed. The server met an unexpected condition.(Internal Server Error)

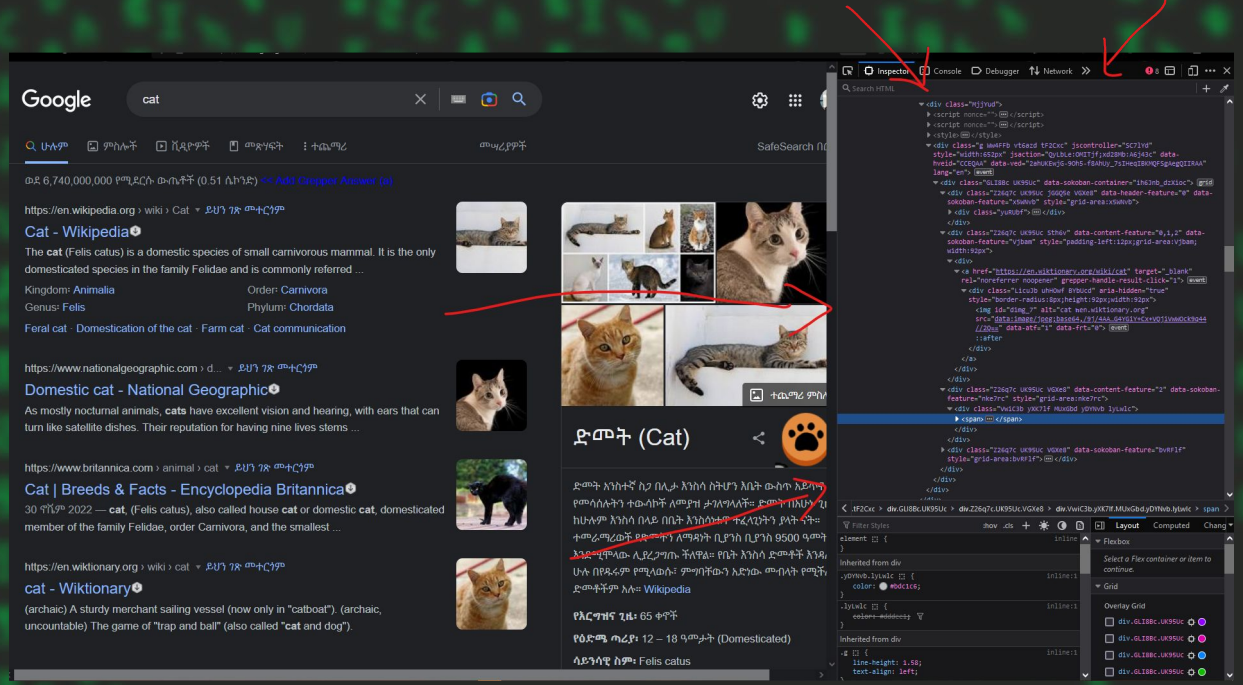
# Where do you see the headers?

The headers are shown on some methods.

1. Developers tool(on browser)
2. Curl
3. Burp suite

# Developers tool

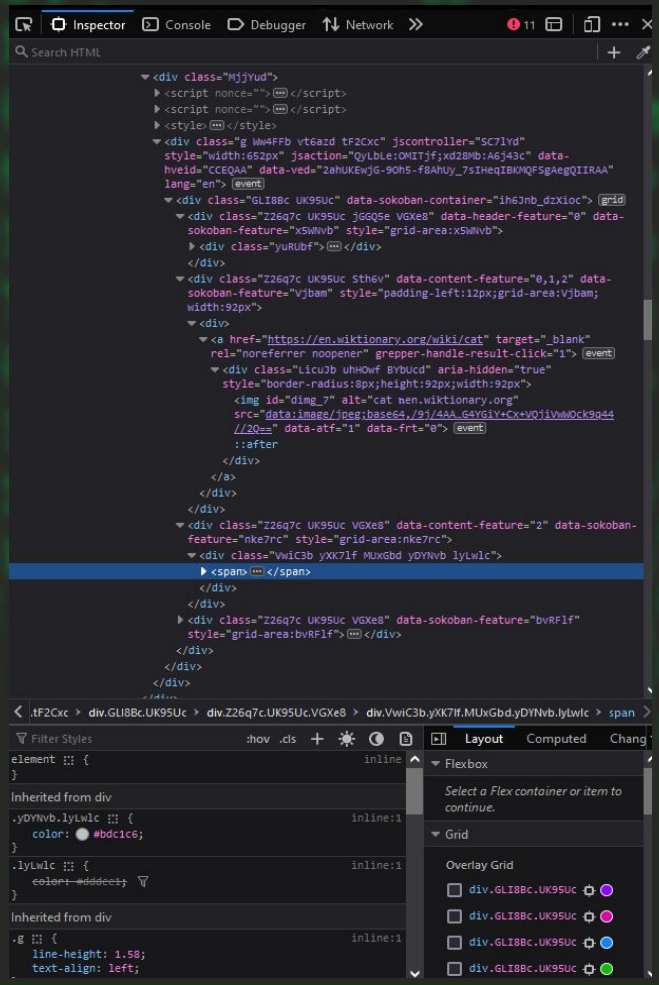
- To open it on browser
  - Press Ctrl+shift+C

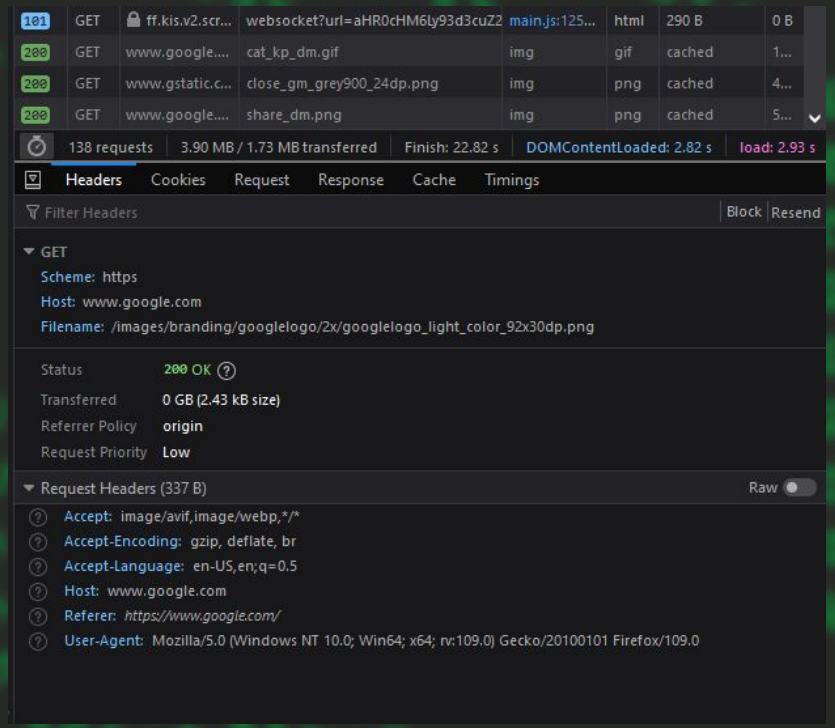


This tools contains lot of things

1. Inspector: to see and edit the HTML and CSS
2. Console: to run some Javascript codes
3. Debugger: used to do debug in runtime
4. Network: to see the requests and responses
5. Storage: to store cache and cookies
6. …

To get the requests we go to the Network tab






Click on on of the requests and see below on the header(drag it up)

# Curl on linux


- cURL, which stands for client URL, is a command line tool that developers use to transfer data to and from a server.
- At the most fundamental, cURL lets you talk to a server by specifying the location (in the form of a URL) and the data you want to send.
- cURL supports several different protocols, including HTTP and HTTPS, and runs on almost every platform.
- This makes cURL ideal for testing communication from almost any device (as long as it has a command line and network connectivity) from a local server to most edge devices.
- Syntax:
  - curl [options] [URL]

```
rexder@HunterMachine ~> curl https://www.google.com
<!doctype html><html itemscope="" itemtype="http://sche
=UTF-8" http-equiv="Content-Type"><meta content="/image
"image"><title>Google</title><script nonce="F7Ff7Vn5la-
mAI',kEXPI:'0,1356780,2629,6059,206,4804,2316,383,246,5
3228,3847,10622,22743,5079,1593,1279,2742,149,1103,840,
95,1851,15756,3,346,230,6459,149,13975,4,1528,2304,7039
3105,2,39761,5679,1020,31122,4569,6258,23418,1246,5841,
4,13463,24,6146,7,1922,5784,3995,20639,752,388,9543,483
182,1125,8051,939,702,5199,1182,559,2328,281,3002,564,5
,135,2125,425,4720,947,352,204,4,6,291,74,29,3186,383,1
,8,20,335,446,3,171,232,1169,593,97,743,574,3129,203,6,
466,321,9,854,1899,125,161,51,2,375,176,1608,114,1199,6
```

# ...Options

- To get web content of site

```
rexder@HunterMachine ~> curl https://www.google.com
<!doctype html><html itemscope="" itemtype="http://schem
=UTF-8" http-equiv="Content-Type"><meta content="/images
"image"><title>Google</title><script nonce="F7Ff7Vn5la-3
mAI',kEXPI:'0,1356780,2629,6059,206,4804,2316,383,246,5,
```

- Change request method

```
rexder@HunterMachine ~> curl -X GET https://www.google.com
<!doctype html><html itemscope="" itemtype="http://schema.or
=UTF-8" http-equiv="Content-Type"><meta content="/images/bra
"image"><title>Google</title><script nonce="Ce1CrqAEFoTuuVK3
MA',kEXPI:'0,18167,1341242,1710,4348,207,4804,2316,383,246,5
```

- To see the response headers

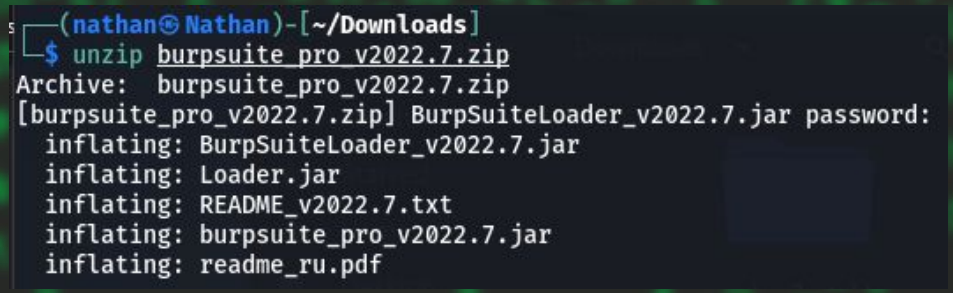
```
rexder@HunterMachine ~ [2]> curl -I https://www.google.com
HTTP/2 200
content-type: text/html; charset=ISO-8859-1
p3p: CP="This is not a P3P policy! See g.co/p3phelp for more in
date: Fri, 27 Jan 2023 13:17:26 GMT
server: gws
x-xss-protection: 0
x-frame-options: SAMEORIGIN
expires: Fri, 27 Jan 2023 13:17:26 GMT
cache-control: private
set-cookie: 1P_JAR=2023-01-27-13; expires=Sun, 26-Feb-2023 13:1
set-cookie: AEC=ARSKqsKyW18JzCv2aWdk0dEg1c0Rn38mDQZScLyTAK7t6ha
=/; domain=.google.com; Secure; HttpOnly; SameSite=lax
set-cookie: NID=511=ANYUlA0RA4Ms0sUveILdfLOtDoNriwZ3TJKsosuZ2zP
lEIPeO_MOymEPOZEGbYi9Wsdx0T7Sa2Qzfgakk5lHe45I1iYh6SpTGLeKM-Mxoa
domain=.google.com; HttpOnly
alt-svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000
```

# Burp suite

- Burp or Burp Suite is a set of tools used for penetration testing of web applications.
- It is developed by the company named Portswigger, which is also the alias of its founder Dafydd Stuttard.
- BurpSuite aims to be an all in one set of tools and its capabilities can be enhanced by installing add-ons that are called BApps.
- Burp Suite is **an integrated platform and graphical tool for performing security testing of web applications**
- it supports the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities.
- It have a lot of tools all together.
- Simply Help Hackers To Act like A Proxy, it will intercept A request,
    - Used to change it or just watch it

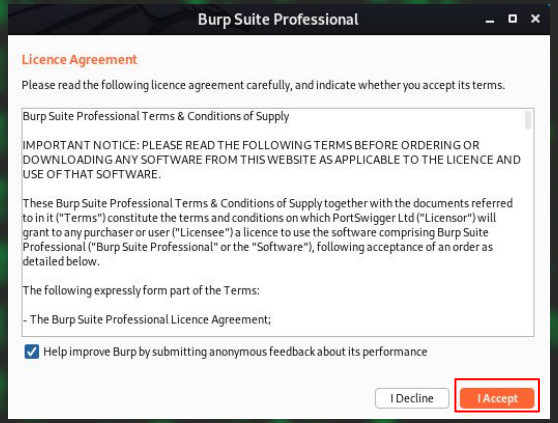
# Installing

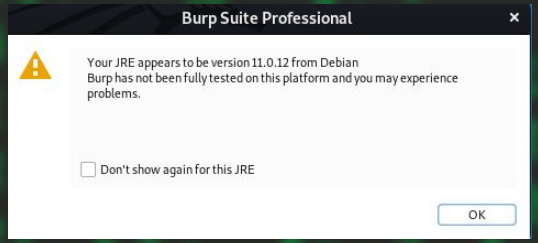
- You can Install the Pro Version(cracked) with the following Steps, but u can use the free too
- Burpsuite have 2 versions
  - Community = free
  - Enterprise - paid
  - Professional - paid
- We will install the professional burp.
  - After Downloading it
  - Extract it
  - On the 1st terminal
    - java -javaagent:BurpSuiteLoader_v2022.7.jar -noverify -jar burpsuite_pro_v2022.7.jar
  - On another terminal
    - java -jar Loader.jar

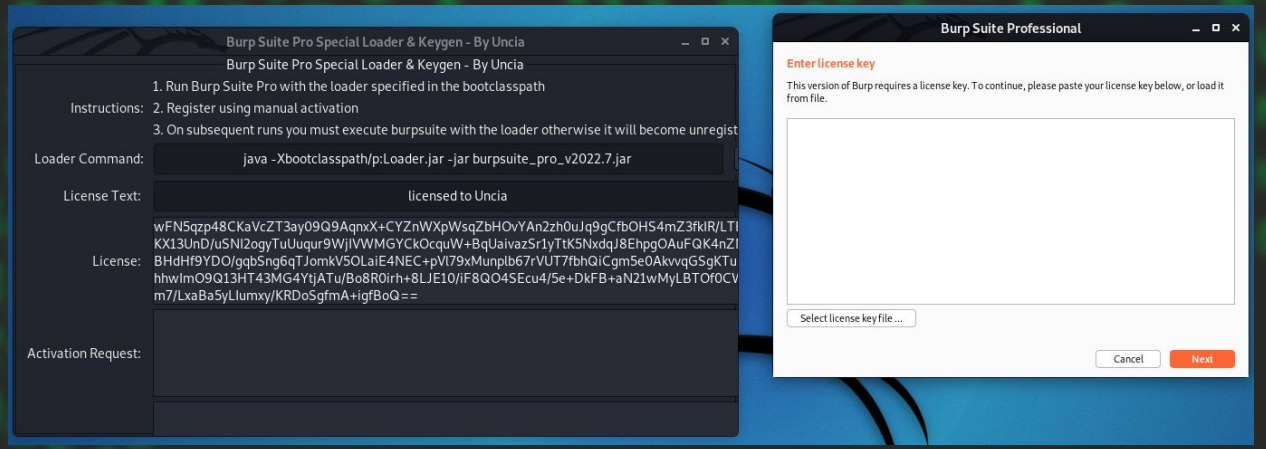


...

- There will 2 screens as you see.
- ON the 1st window Change the following:
  - License Text: Unicia to your name.
- Then Copy the License code
  - Use Ctrl-C / Ctrl-V
- Paste it to the Burp “Enter License Key”
- Next…

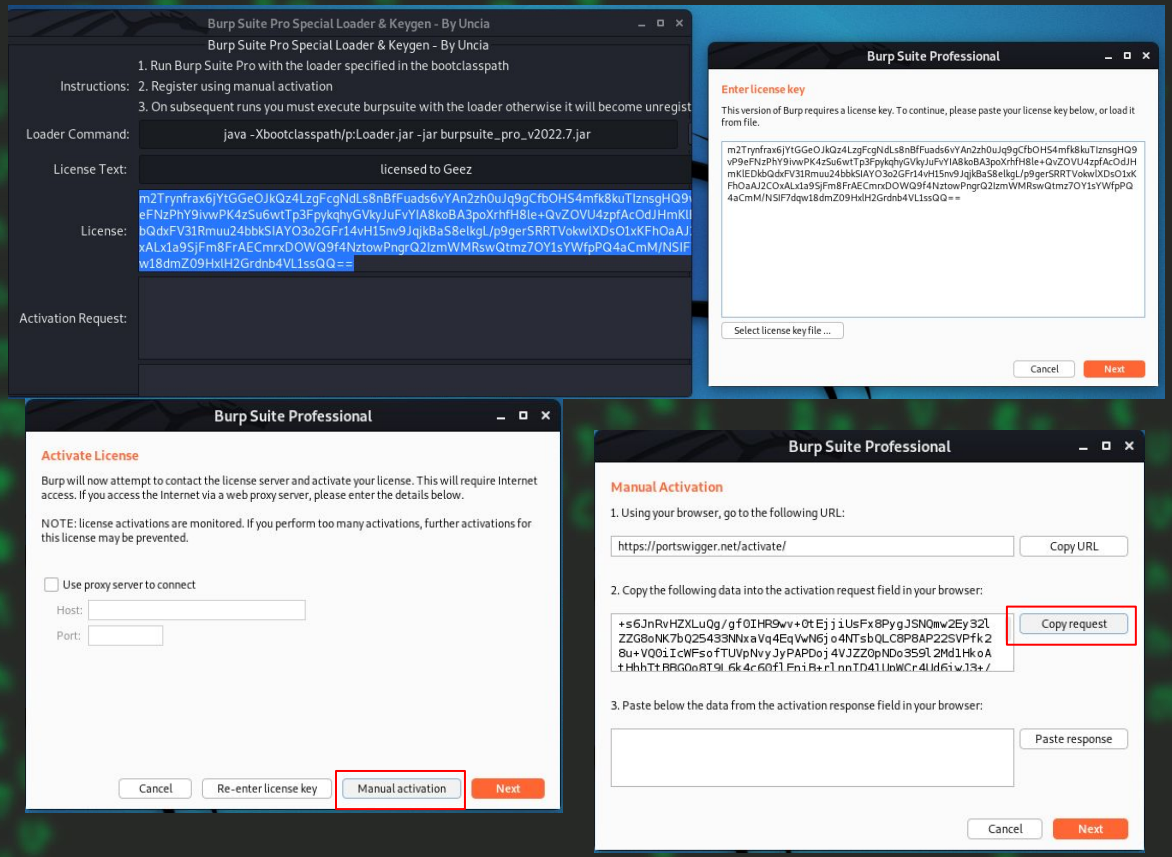


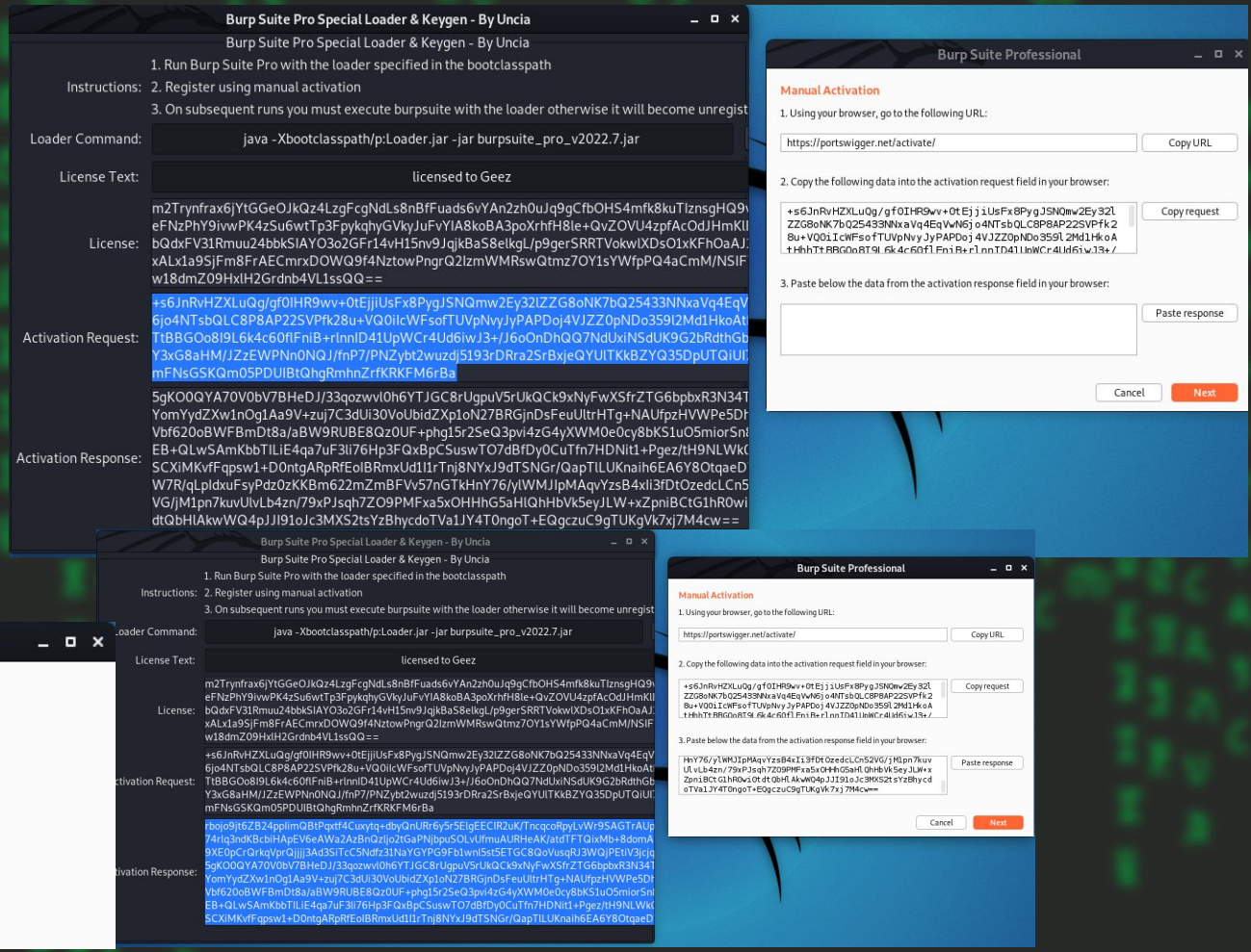
...

- After the Next button
- Click Manual Activation
- Copy request
- Paste it on The Loader “Activation Request”



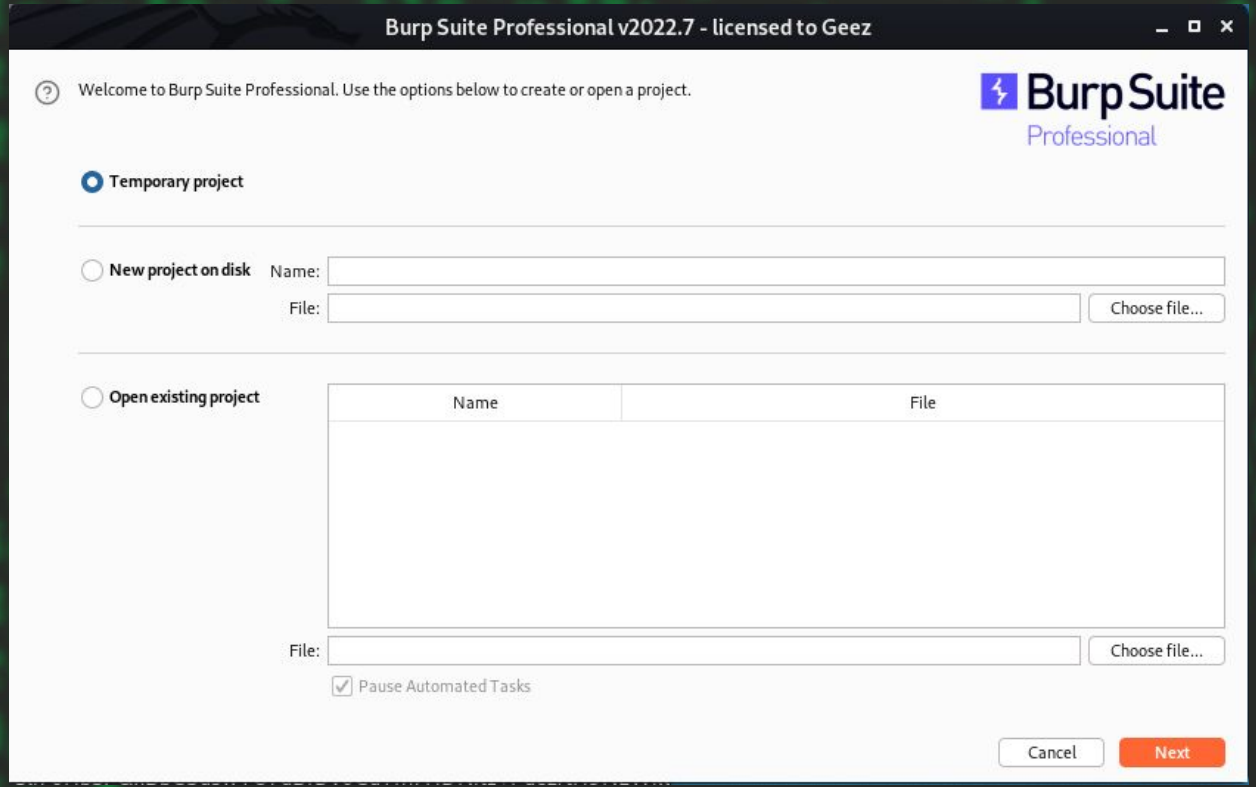
...

- Then Copy The “Activation Response”
- Click paste respose on Burpsuite.
- Finally🔥, Activated Burp PRO!
- You can close the loader



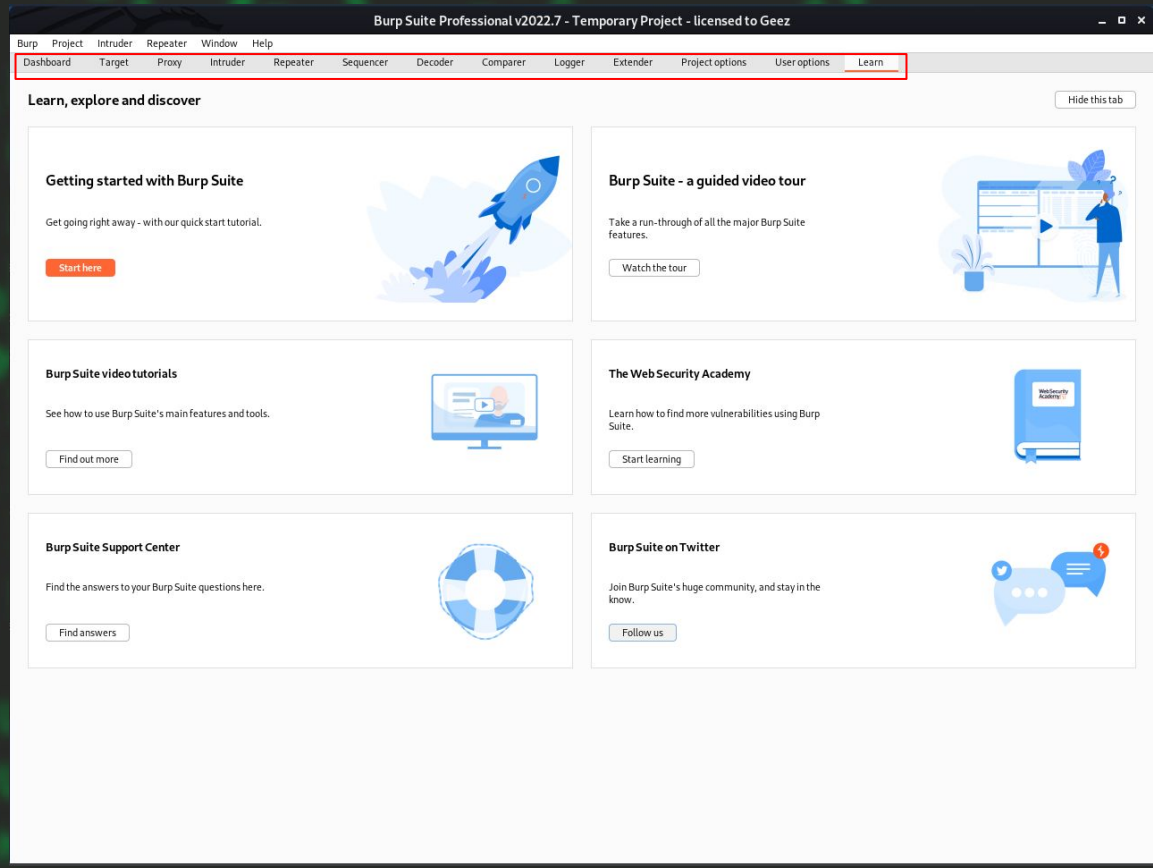
# Opening Burp

- This is The 1st page of Burp.
- Helps you:
  - To open project
  - Create new project
  - Use Temporary Project
- Then just click “Start BUrp”



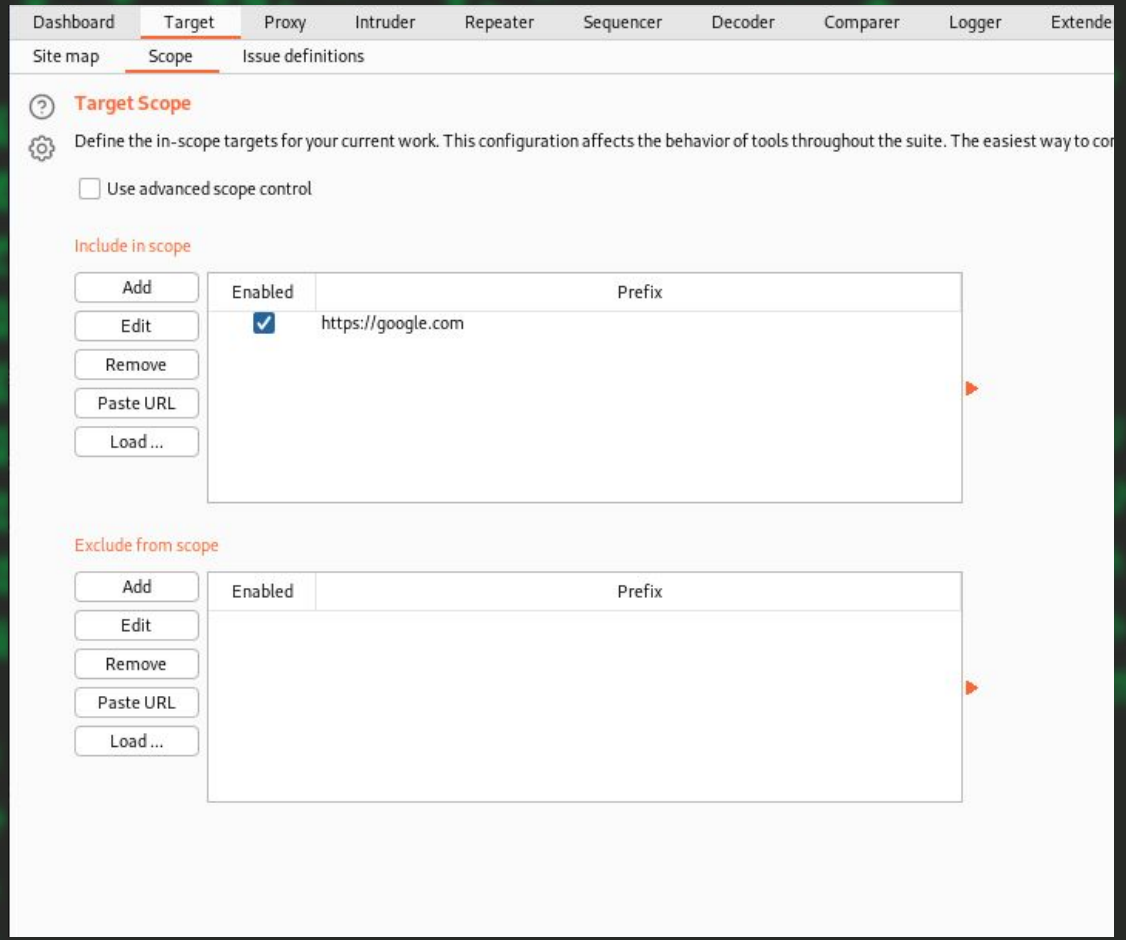
# Starting Burp

- This is the Home page of Burp.
- There are many tabs as you see on the above.
  - Target: to add targets inscope & see progress
  - Proxy: to setup proxy IP's also to intercept and watch requests and responses
  - Intruder: to Do bruteforce attacks
  - Repeater: To do manual checks
  - Comparer: to compare 2 requests/responses
  - ....
- When you Try to open burp next time goto the folder and execute this command
- java -javaagent:BurpSuiteLoader_v2022.7.jar -noverify -jar burpsuite_pro_v2022.7.jar



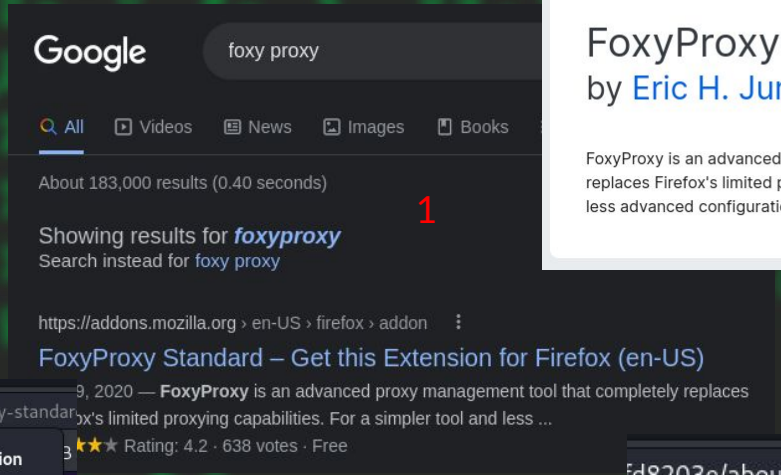
...

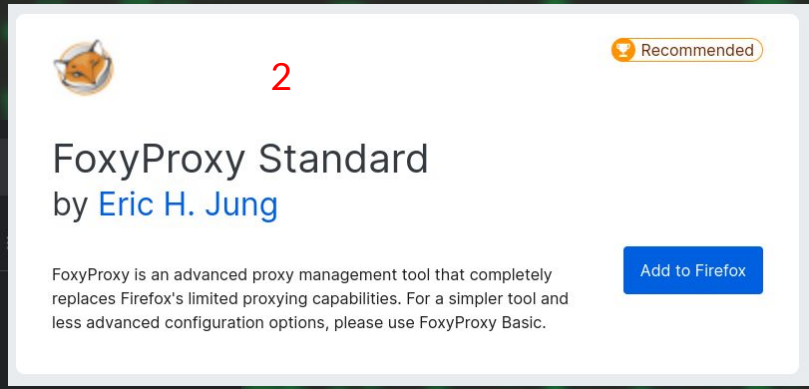
- Now Burp will only intercept or see Requests and reponses from google.com.

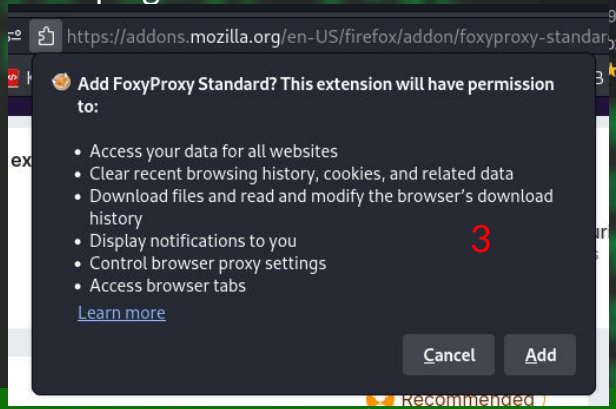


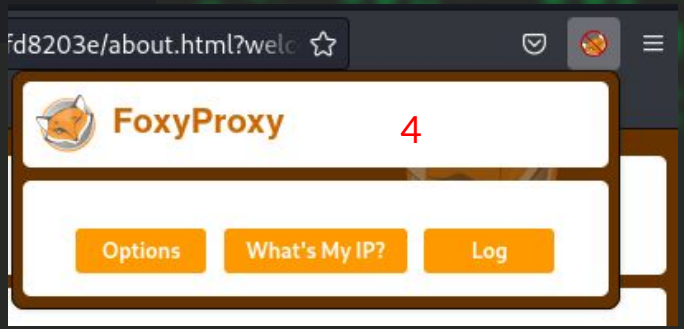
# 1 step to see interception

- Now, we need to connect our browser with Burp.
- To Do this we can setup our proxy setting on our browser or install a proxy plugin.



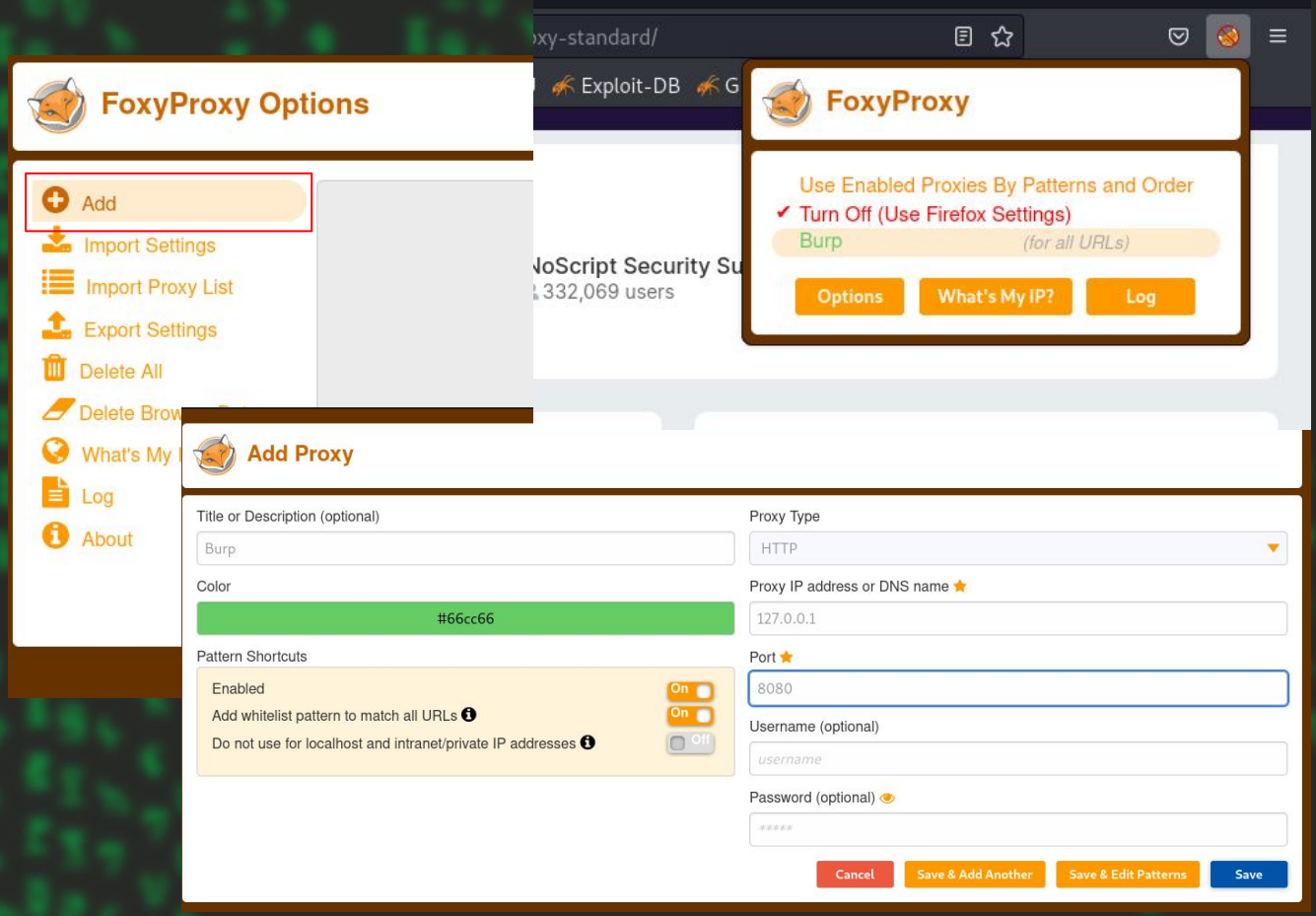



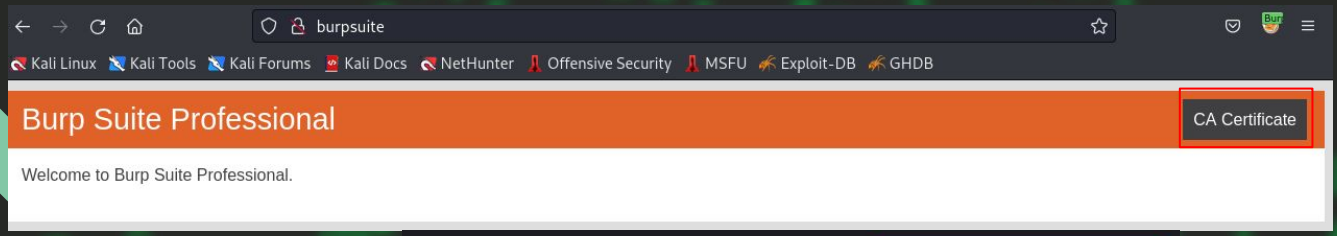
…

- Now Add The Burp Proxy IP here.
- Add
- Title: anything u need “Burp”..
- Proxy IP: 127.0.0.1
- Port: 8080
- Save

Now to Intercept u just click the foxy icon and click burp

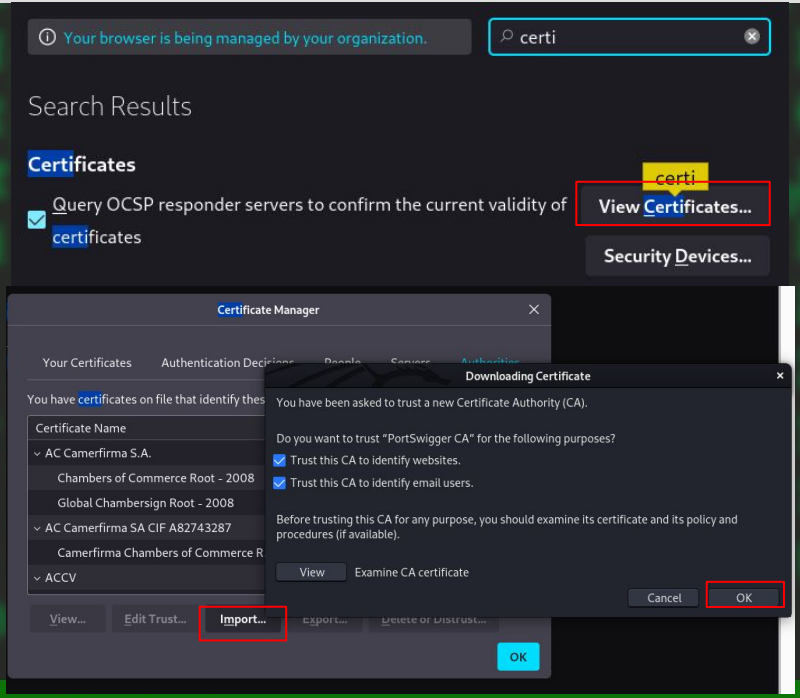
For HTTPS sites

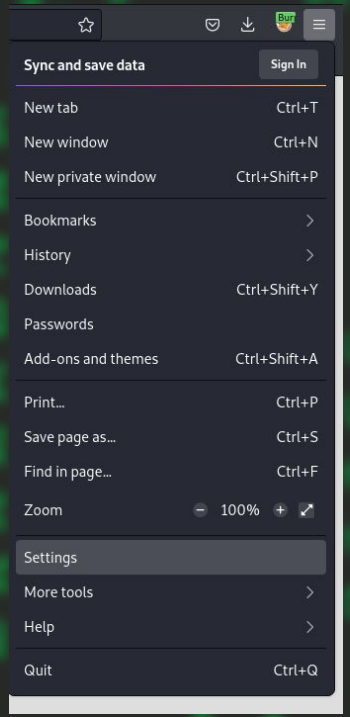





1. Goto burpsuite/
2. Download the CA Certificate
3. Goto Setting
4. Search fro Certificates
5. View Cerit.
6. Import
7. Add the cacert
8. Tick the 2 boxs
9. OK

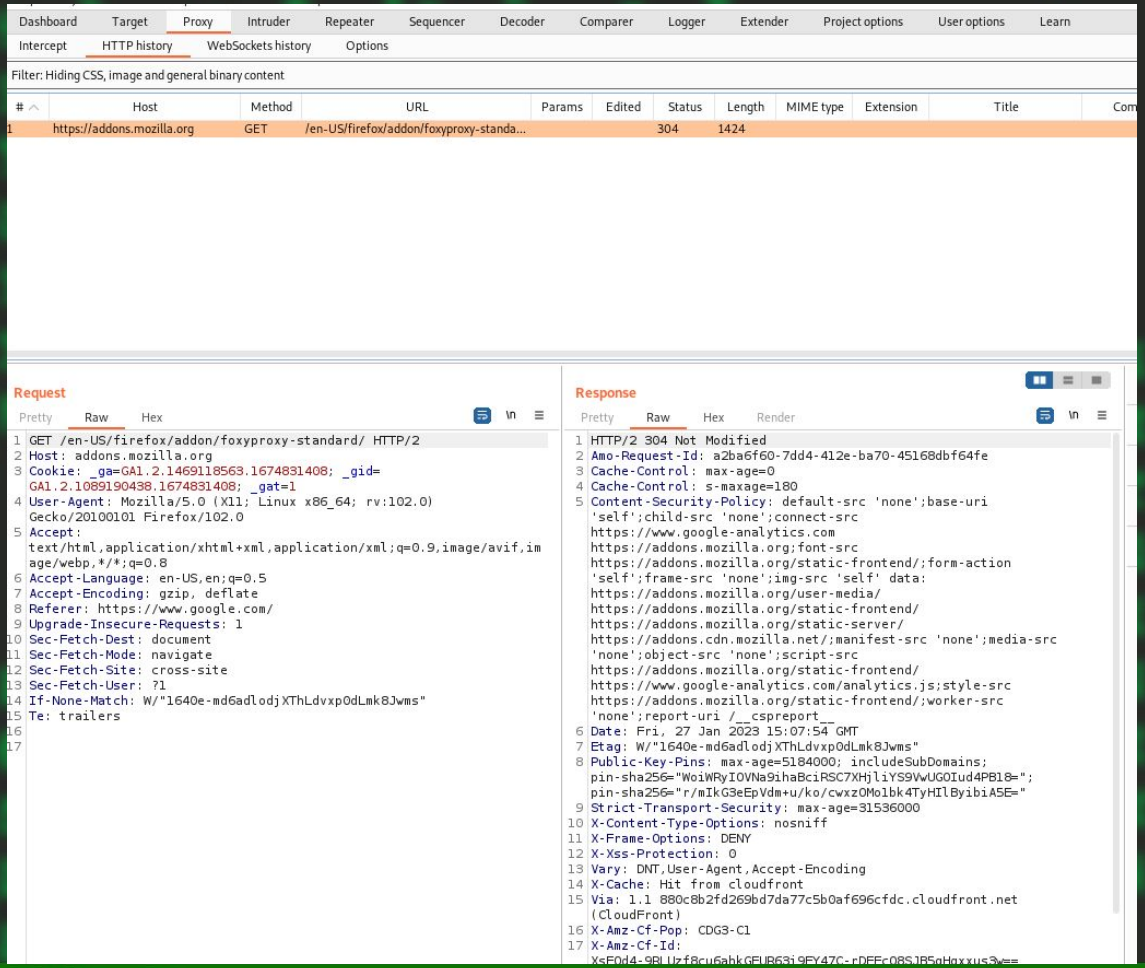
DONE



# Demo

- Now search 1 site and
- goto burp Proxy
- HTTP history
- and check it

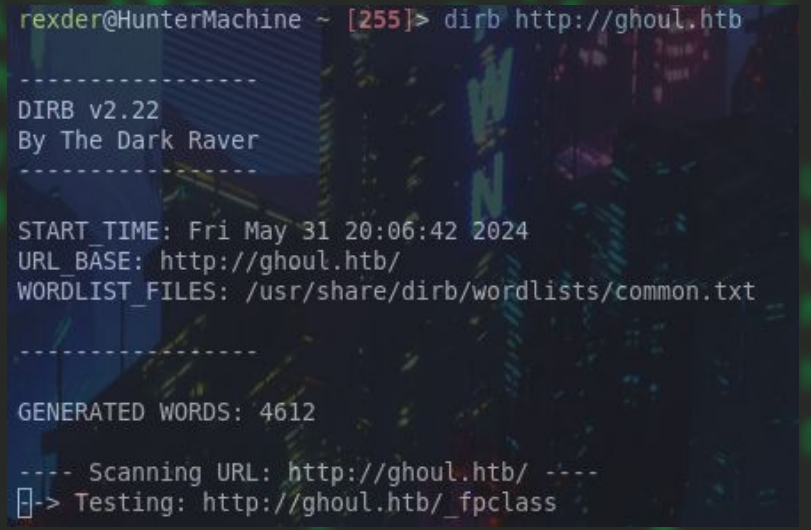


# Exercise 1

1. Install and Configure Burpsuite.
2. See the request from example.com
3. Copy and Send the request of example.com to the GTST Group

# Web Enumeration

- Web Enumeration is Gathering Informations about a websites, Web Apps, APIs.
- Things We gather:
  - Development Framework
  - Directories
  - Sensitive Files
  - VHOSTS
  - Subdomains
- On web Enumeration there is a term called “Fuzzing”
- Fuzzing Reference to Sending random requests to a server and determining a bug based on what the server responds.
  - But we Sometimes Use it when we try to mention an attack called “BruteForce”(they are almost same)



# OWASP Top 10


- OWASP Stands for Open Web Application Security Project.
- The OWASP Top 10 is **a standard awareness document for developers and web application security**.
- It represents about the most critical security risks to web applications.
- Globally recognized by developers as the first step towards more secure coding.
- This Project Releases Top 10 risky vulns every 4 years.
- Detail: https://owasp.org/Top10/
- OWASP also releases API vulnerabilities

...

- This is the OWASP's Top 10 Vulns reported in 2021
- Each Vulnerabilities are Detailed.
- Let us see some Common web Vulnerabilities

**Top 10:2021 List**

A01 Broken Access Control

A02 Cryptographic Failures

A03 Injection

A04 Insecure Design

A05 Security Misconfiguration

A06 Vulnerable and Outdated Components

A07 Identification and Authentication Failures

A08 Software and Data Integrity Failures

A09 Security Logging and Monitoring Failures

A10 Server Side Request Forgery (SSRF)

# Brute-Force and Dictionary Attack

- It is included in Broken Authentication/Access Control Bug
- This is a kind of attack that is usually done to a login pages.
- It uses Wordlist and try to check a lot of words in the place of the username and password.
- It is a guessing game but the guess is done with computer.

```
[80][http-get-form] host: 192.168.100.155   login: admin   password: password
[80][http-get-form] host: 192.168.100.155   login: admin   password: p@ssword
[80][http-get-form] host: 192.168.100.155   login: admin   password: 12345
[80][http-get-form] host: 192.168.100.155   login: admin   password: 1234567890
[80][http-get-form] host: 192.168.100.155   login: admin   password: Password
[80][http-get-form] host: 192.168.100.155   login: admin   password: 123456
[80][http-get-form] host: 192.168.100.155   login: admin   password: 1234567
[80][http-get-form] host: 192.168.100.155   login: admin   password: 12345678
[80][http-get-form] host: 192.168.100.155   login: admin   password: 1q2w3e4r
[80][http-get-form] host: 192.168.100.155   login: admin   password: 123
[80][http-get-form] host: 192.168.100.155   login: admin   password: 1
[80][http-get-form] host: 192.168.100.155   login: admin   password: 12
1 of 1 target successfully completed, 12 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-07-27 15:28:24
```

# Foothold

There are some tools, used to Crack Passwords of Files, Hashes

- Hashcat
- John The Ripper
- More..

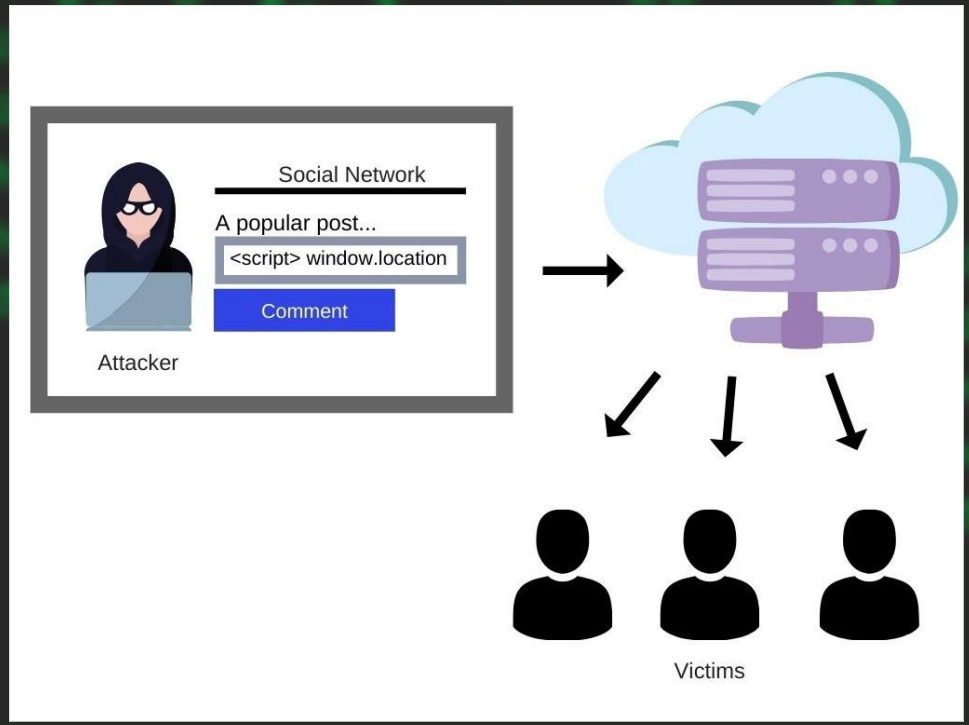
Try to Learn These tools for further System and web hacking Potential

https://www.youtube.com/watch?v=5fy6Lq1vgZk&pp=ygUHSGFzaGNhdA%3D%3D

https://www.youtube.com/watch?v=XjVYI1Ts6XI&pp=ygUQam9obiB0aGUgcmlwcGVyIA%3D%3D

# XSS

- It is included in Injection Bug
- XSS/ Cross site scripting/ is a vulnerability that leads to a lot of huge attacks.
- This Bug is exploited. As the following
  - If there is a search place and the search place expects a text to search and displays below.
  - But if we add some html/JS codes on that place, this means it will add the code to the innerHTML
  - SO our code will be executed!



# Cross-Site Scripting

allows attackers to inject malicious scripts into web pages viewed by other users.

haxxed.com?cookie=DFjkhlaa80

breddit

How much do you folks like bread?

**roll_with_it**
I dream of baking tins.

**I_knead_you_right_now**
I love it so much, I think I might be part duck.

I love it so much, I think I might be part duck.

breddit.com/r/comments/how_mu

breddit

How much do you folks like bread?

**roll_with_it**
I dream of baking tins.

**I_knead_you_right_now**
I love it so much, I think I might be part duck.

```
<script>alert('Your croissants are limp and sad.')</script>
```

haxxed.com?cookie=DFjkhlaa80

Your croissants are limp and sad.

OK

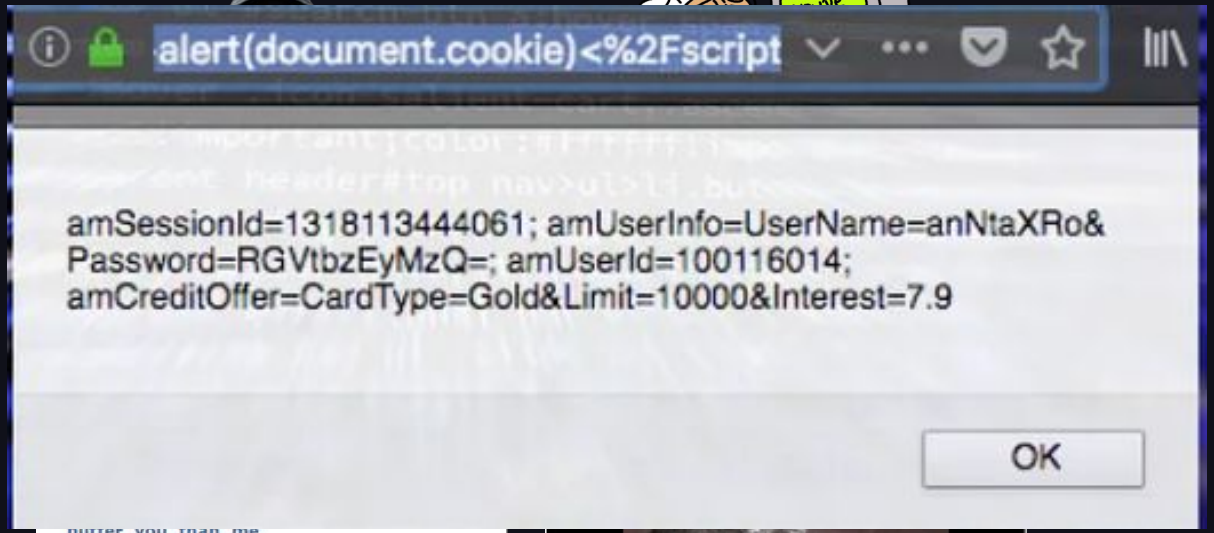
I dream of baking tins.

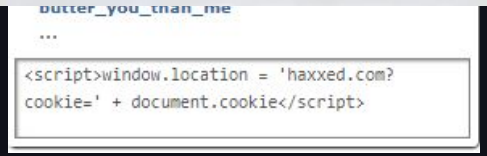
**I_knead_you_right_now**
I love it so much, I think I might be part duck.

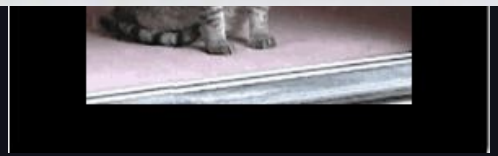
**butter_you_than_me**
...

SO???

What is the matter?



This is Simple Demo,
But Attacker can Use
this injection attack
to access your
authentication tokens,
Session IDs
They can Use this and
Hook your browser with
tool called “BeefXSS”

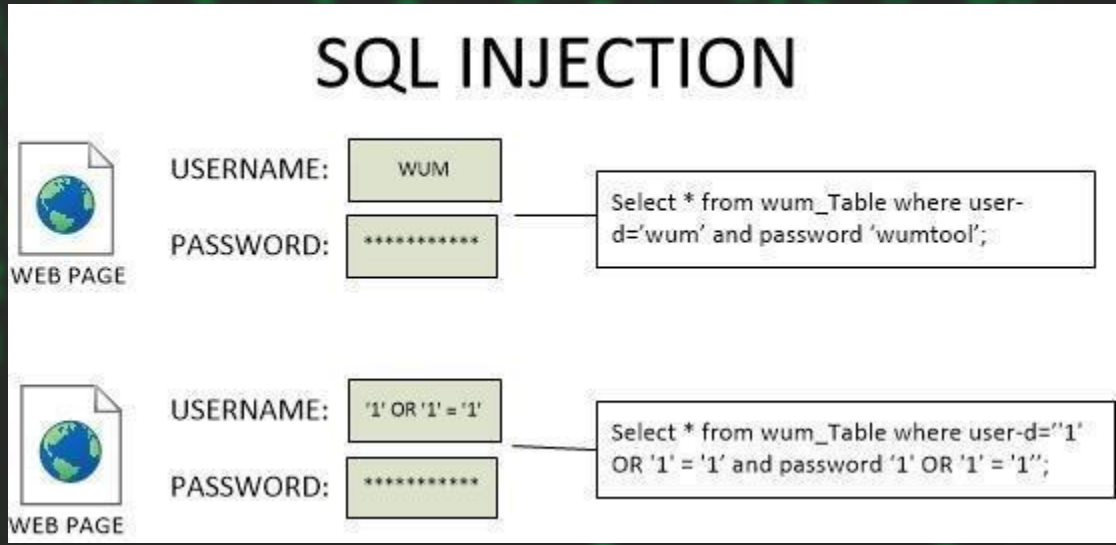
# REMEMBER

All the Injection Attacks are done on some Parameter.

```
https://example.com/comment=<script>alert('XSS')</script>   -- correct
https://example.com/<script>alert('XSS')</script>  -- Not
```

# SQL injection

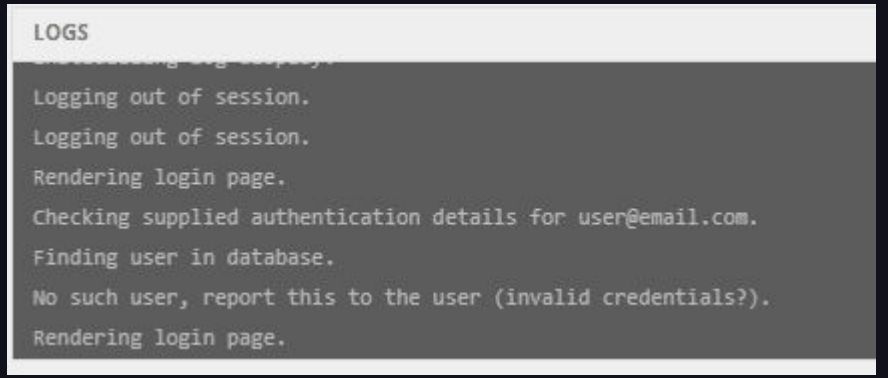
- It is included in Injection Bug
- It is same with xss, but here we will add a sql code to the search place.
- SQL is a query language used in Back end to retrieve data from database.
- Most of the time used to bypass login pages.



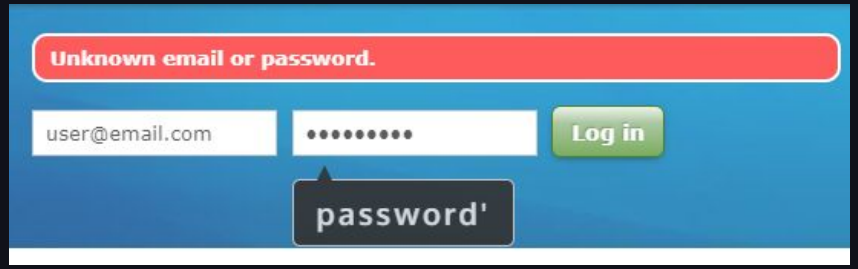
# SQL Injection

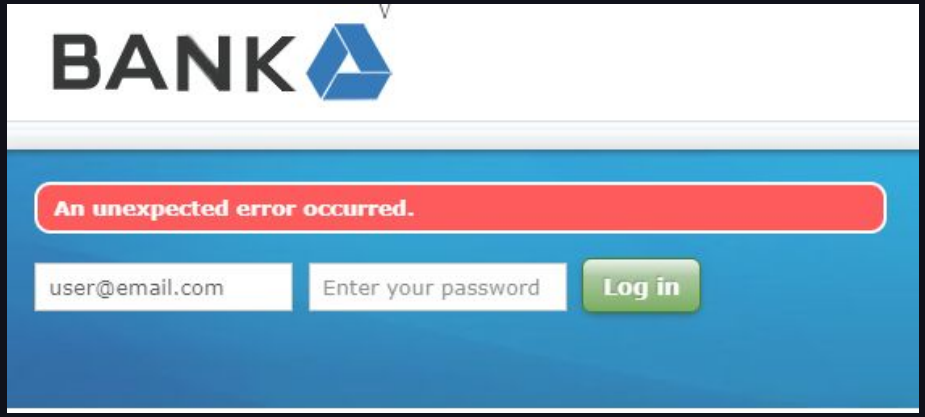
Attackers Use this Vuln to manipulate or gain unauthorized access to a ***database***.

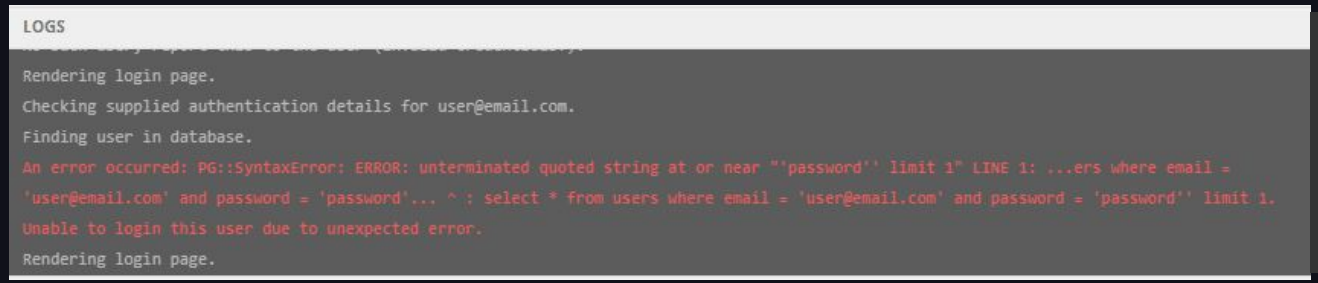
DEMO:

# …Nathan Bank😌




```
SELECT *
  FROM users
 WHERE email = 'user@email.com'
   AND pass  = 'password'' LIMIT 1
```


```
SELECT *
  FROM users
 WHERE email = 'user@email.com'
   AND pass  = '' or 1=1--' LIMIT 1
```


- This is Just Demo, But this is how the Hack happens,
- On the payload Attackers can Craft a SQL query that can cause RCE, Full Database Detail listing and much moreeeee.

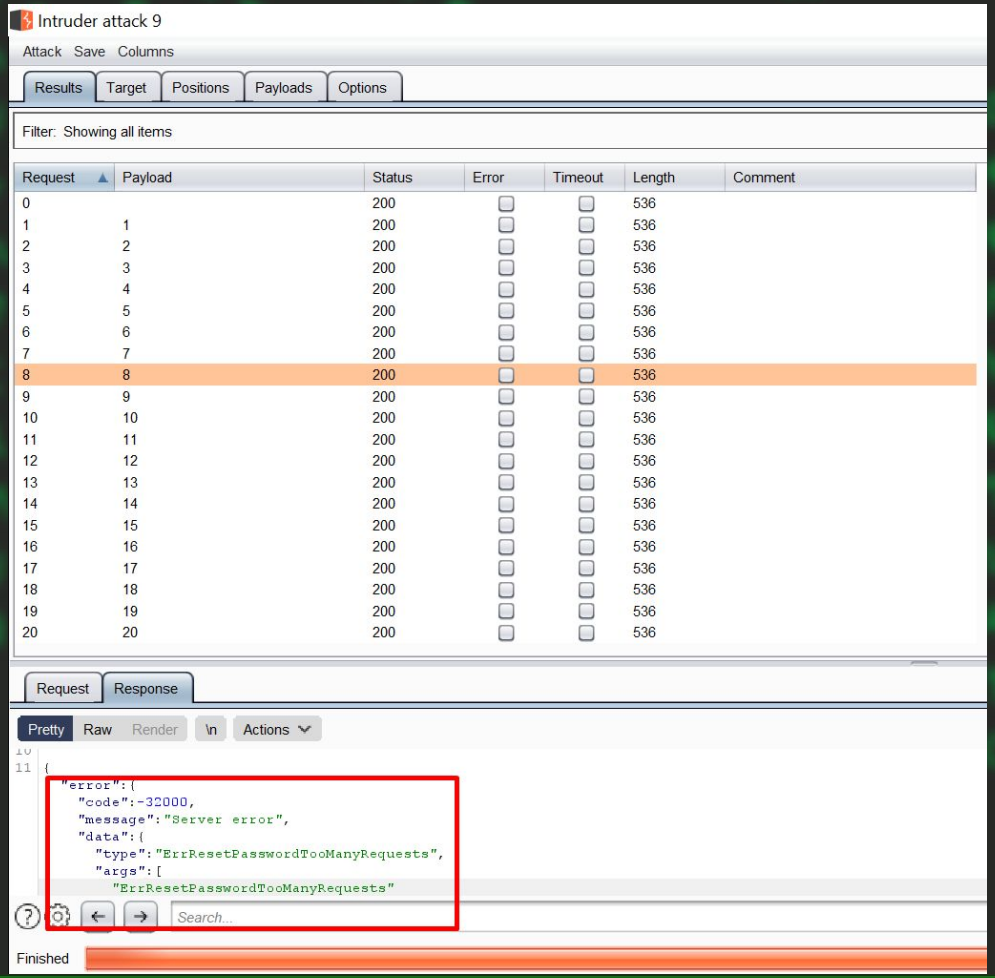
# Further

Learn about Tool called SQLMap.

```
$ python sqlmap.py -u "http://debiandev/sqlmap/mysql/get_int.php?id=1" --batch
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.3.4.44#dev}
|_ -| . [']     | .'| . |
|___|_  [,]_|_|_|__,|  _|
      |_|V...       |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent i
s illegal. It is the end user's responsibility to obey all applicable local, state and fed
eral laws. Developers assume no liability and are not responsible for any misuse or damage
 caused by this program

[*] starting @ 10:44:53 /2019-04-30/

[10:44:54] [INFO] testing connection to the target URL
[10:44:54] [INFO] heuristics detected web page charset 'ascii'
[10:44:54] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:44:54] [INFO] testing if the target URL content is stable
[10:44:55] [INFO] target URL content is stable
[10:44:55] [INFO] testing if GET parameter 'id' is dynamic
[10:44:55] [INFO] GET parameter 'id' appears to be dynamic
[10:44:55] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
 (possible DBMS: 'MySQL')
```

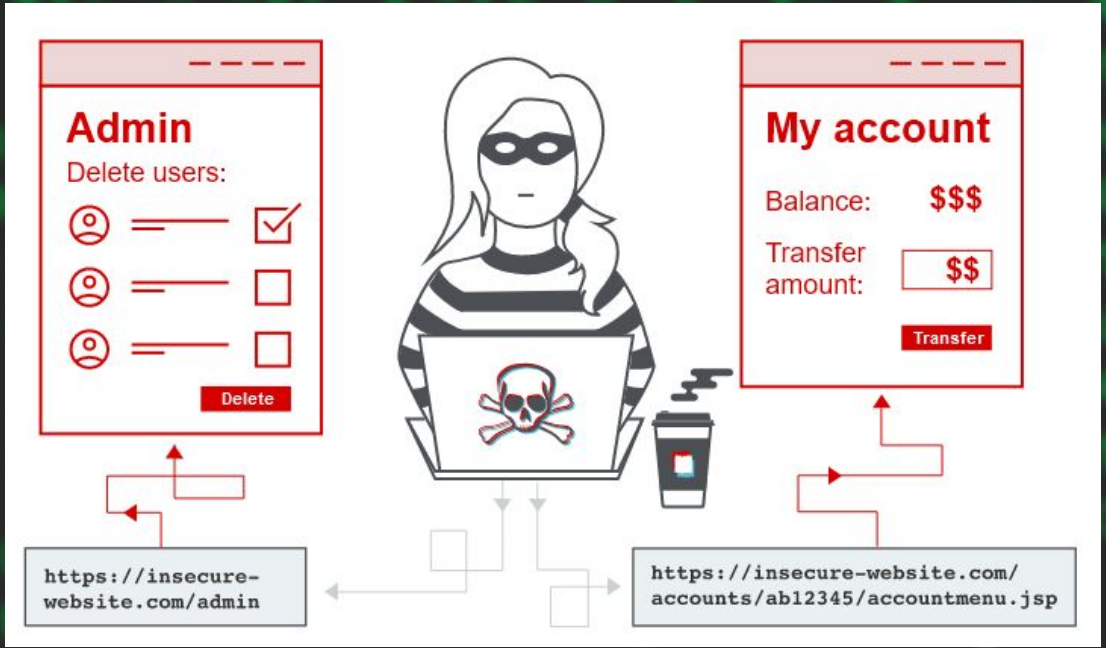
# Rate limit

- This is a limiting problem.
- Think like if the developers did made a limit to some task.
- Example:
  - If there is website , that send an OTP/verification code
  - And if it doesn't limit to some code sends only.
  - I can make the site to send 100000 OTPs to ma phone this means
  - We will make the site to lose a lot of money



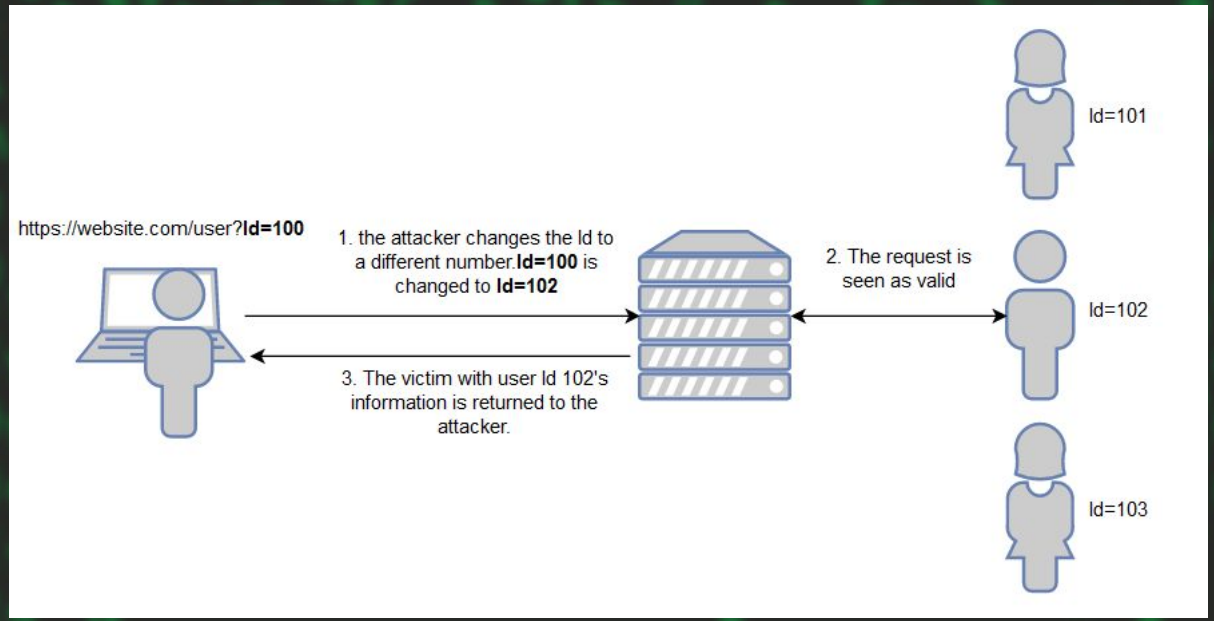
# Access Control

- This is a problem occurred on how a websites control an access of a user.
- If there is a problem on the access control.
- Then think that normal user can get access of admins or root user.
- This is Good Bug to find because u dont have that much automation tools.



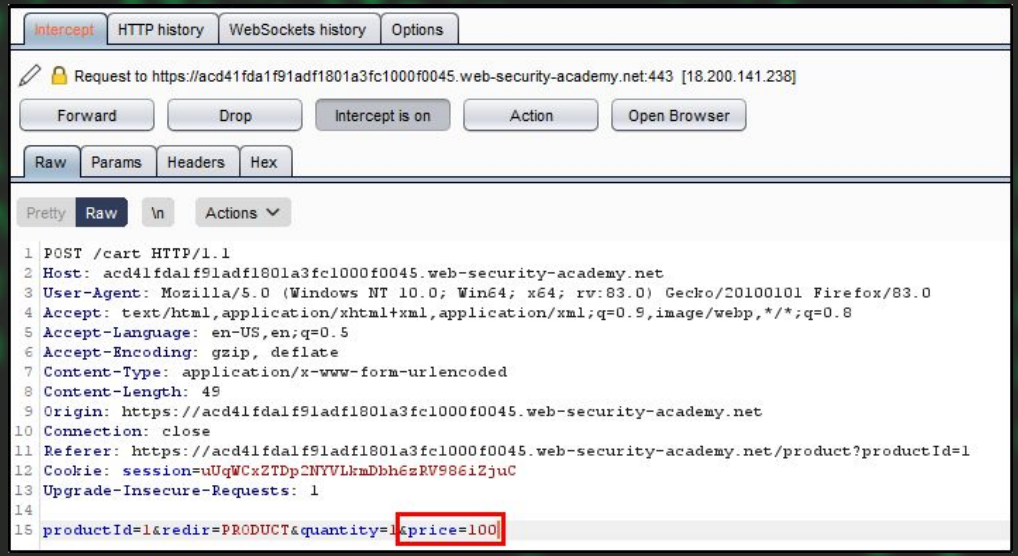
# IDOR

- IDOR/ **Insecure direct object references** /: is a bug included in Access Control.
- This is a bug that happens when you have an id number 1
- And is abebe is id 1 then if i changed that number to 0 and got another users information



# Business Logic

- This Bug is a logic Flow.
- It occurs by the way how the programmer thinks and hackers thinking.
- Think like if You have bank website and it have a purpose of send money.
- Then if i can change add to negative numbers
- And if the site responded by giving me more money and minimizing the amount of the user i planned to send then this is business logic
- Also this is a Good Bug to learn.



# HOW TO ADVANCE ON THIS FIELD

You can Follow this Road map, TO be Good Web Penetration tester:

1. Learn Web Development( Learn Basics Not advanced Development)
    a. frontend(HTML,CSS,JS)
    b. backend(Django or nodeJS)
        i. use Youtube resources they will teach u the basics and Quickly
2. Learn Web Attacks
    a. You can use PortSwigger WebAcademy
3. Learn Web Enumeration techniques
    a. Tools like(VIM,FFUF,SQLMAP,...)
4. Do Web challenges ( CTFs )

HERE YOU ARE A GOOD SKILLED PENTESTER!

5. Try to test Ethiopian Websites
6. Try to test Foreign Websites.

# Reading Assignment

Read about tools called John The Ripper, Hashcat and Hydra, FUFF and Do the following questions.

1. What is the Tool?
2. For which kinda Password Attack is made.
3. How to Install
4. What are the options ?(Explain some)
    a. Explain the How we can do VHOST,subdomain and Directory enumeration using FUFF

Submitted in form of docs/pdf

Resources will be attached on the google classroom, that you can use them.

# Class is over

1. DO notes
2. Practice
3. Ask question

- Be ready for the assessment exam!