

Title: Sniffing in a Controlled Environment: By Rohinish Sharma

## Objective

The goal of this demonstration is to show how **unencrypted credentials** (like usernames and passwords) can be intercepted using **packet sniffing techniques** in a controlled lab setup. This helps organizations understand the importance of **encrypting network traffic** and the **risks of insecure protocols** like HTTP and FTP.

## Key Concepts

### What is Packet Sniffing?

Packet sniffing is the process of **monitoring and capturing data packets** as they travel over a network. Tools like **Wireshark** or **tcpdump** can be used to **analyze this traffic**, making it possible to extract sensitive information — if the data is not encrypted.

### What is Network Traffic Analysis?

This refers to examining packets flowing through a network to:

- ⑩ Identify communication patterns
- ⑩ Monitor protocol use
- ⑩ Detect suspicious activity or sensitive data leaks

### Risks of Unencrypted Data Transmission

When data (especially **credentials**) is transmitted without encryption:

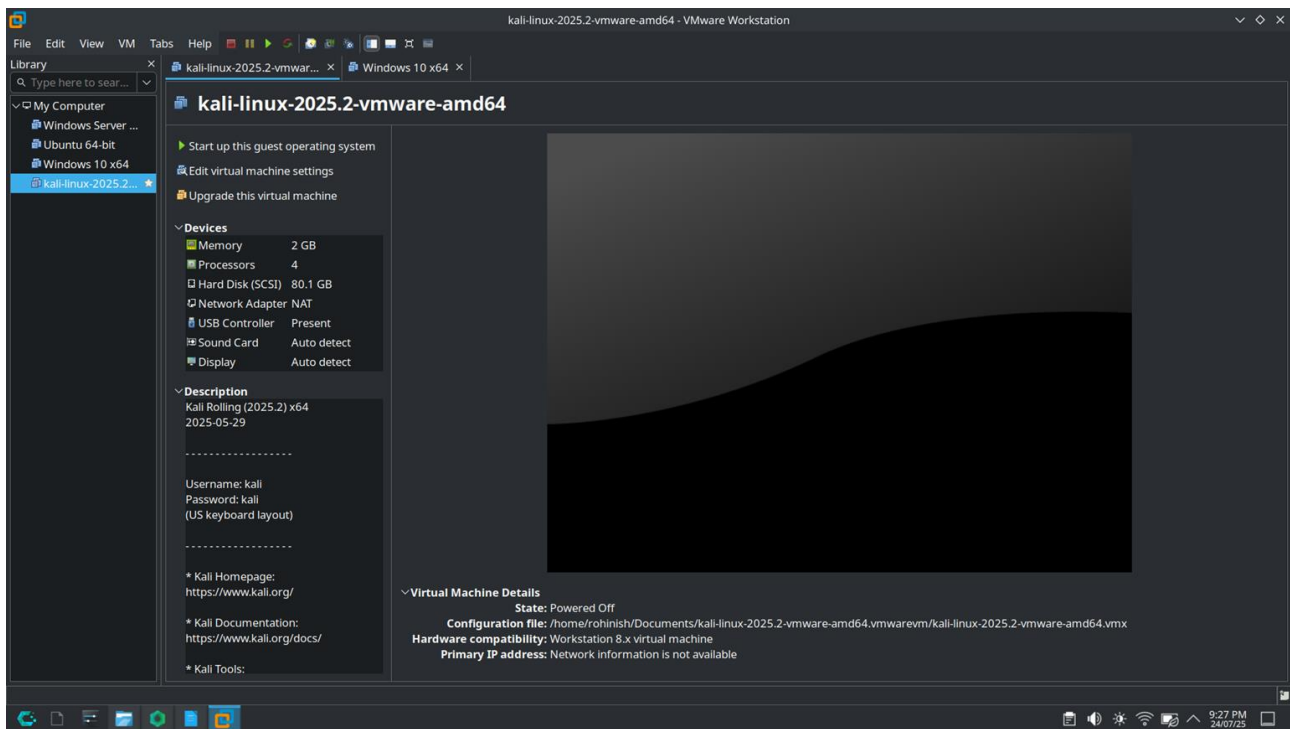
- ⑩ Anyone on the same network can **intercept** it
- ⑩ Tools like Wireshark can easily **read login forms, FTP credentials, emails, etc.**
- ⑩ Attackers can use this to **impersonate users, access systems, or steal data**

## Step 1: Setting Up the Lab Environment

**Objective:** Create a safe and isolated network to simulate real-world attacks without any legal or ethical risks.

### Actions:

- ⑩ Launch two Virtual Machines (VMs) in VMware:
  - ⑩ **Kali Linux** (Attacker)
  - ⑩ **Windows** (Victim)
- ⑩ Set both VMs to use the **Host-Only Network** adapter.



Host-Only networking ensures both machines are on the **same private subnet**, allowing direct communication while keeping them **isolated from the internet**. This simulates a **local corporate LAN** scenario for penetration testing.

## Step 2: Identifying IP Addresses & Verifying Connectivity

```
File Actions Edit View Help
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.206.132 netmask 255.255.255.0 broadcast 172.16.206.255
    inet6 fe80::b5a4:cf7f:6240:65 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:3e:06:8f txqueuelen 1000 (Ethernet)
    RX packets 5 bytes 830 (830.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 28 bytes 3464 (3.3 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 8 bytes 480 (480.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 480 (480.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali@kali)-[~]
$
```

```
Microsoft Windows [Version 10.0.19045.2965]
(c) Microsoft Corporation. All rights reserved.

C:\Users\rohinish>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

    Connection-specific DNS Suffix  . : localdomain
    Link-local IPv6 Address . . . . . : fe80::8f5a:b362:880a:f3fd%5
    IPv4 Address. . . . . : 172.16.206.135
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 172.16.206.2

C:\Users\rohinish>
```

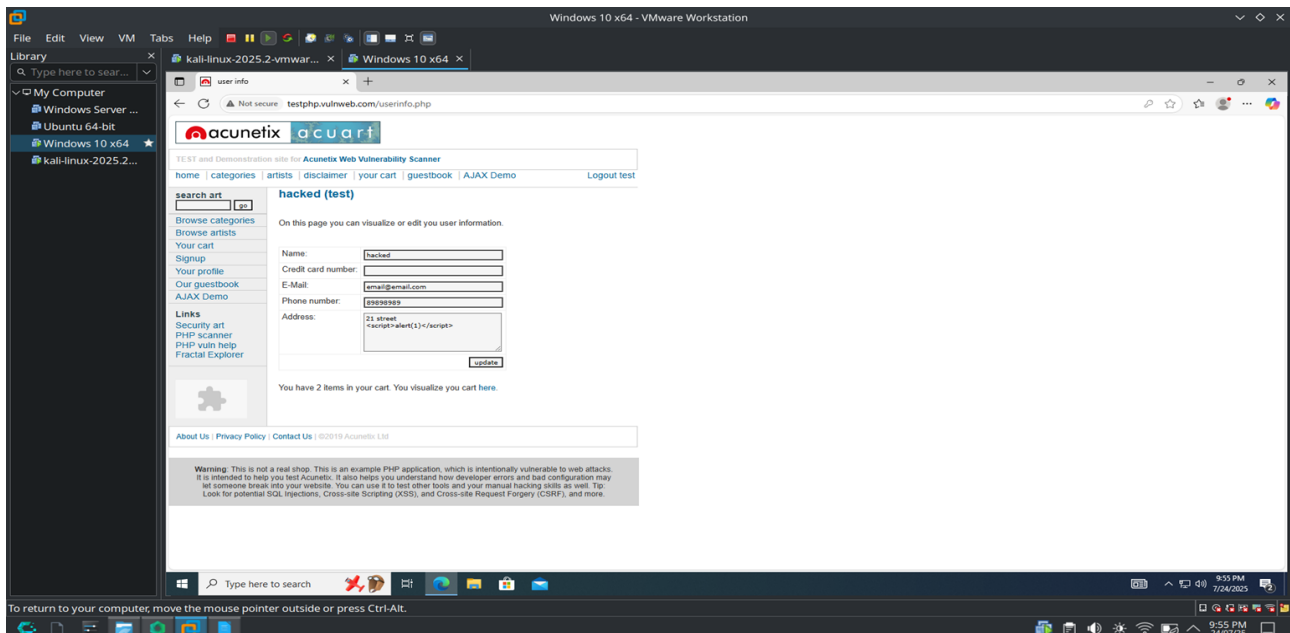
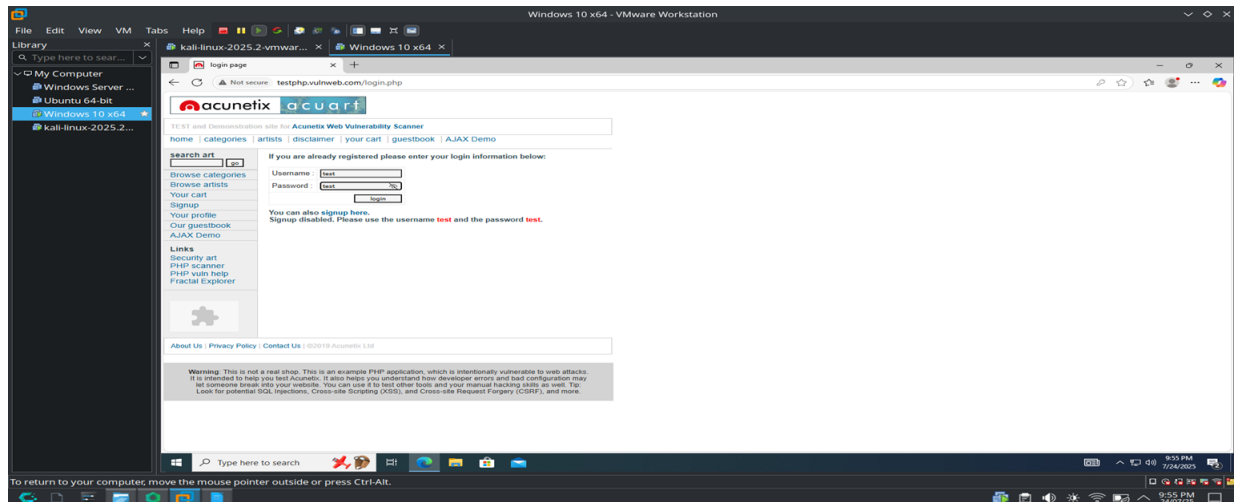
```
(kali@kali)-[~]
$ ping 172.16.206.135
PING 172.16.206.135 (172.16.206.135) 56(84) bytes of data:
64 bytes from 172.16.206.135: icmp_seq=1 ttl=128 time=1.15 ms
64 bytes from 172.16.206.135: icmp_seq=2 ttl=128 time=1.15 ms
64 bytes from 172.16.206.135: icmp_seq=3 ttl=128 time=0.700 ms
64 bytes from 172.16.206.135: icmp_seq=4 ttl=128 time=1.62 ms
64 bytes from 172.16.206.135: icmp_seq=5 ttl=128 time=1.77 ms
64 bytes from 172.16.206.135: icmp_seq=6 ttl=128 time=1.13 ms
64 bytes from 172.16.206.135: icmp_seq=7 ttl=128 time=1.80 ms
64 bytes from 172.16.206.135: icmp_seq=8 ttl=128 time=1.70 ms
64 bytes from 172.16.206.135: icmp_seq=9 ttl=128 time=1.46 ms
64 bytes from 172.16.206.135: icmp_seq=10 ttl=128 time=1.41 ms
64 bytes from 172.16.206.135: icmp_seq=11 ttl=128 time=1.24 ms
64 bytes from 172.16.206.135: icmp_seq=12 ttl=128 time=1.80 ms
64 bytes from 172.16.206.135: icmp_seq=13 ttl=128 time=1.16 ms
64 bytes from 172.16.206.135: icmp_seq=14 ttl=128 time=1.70 ms
64 bytes from 172.16.206.135: icmp_seq=15 ttl=128 time=1.59 ms
64 bytes from 172.16.206.135: icmp_seq=16 ttl=128 time=0.666 ms
64 bytes from 172.16.206.135: icmp_seq=17 ttl=128 time=1.69 ms
64 bytes from 172.16.206.135: icmp_seq=18 ttl=128 time=1.78 ms
64 bytes from 172.16.206.135: icmp_seq=19 ttl=128 time=1.69 ms
64 bytes from 172.16.206.135: icmp_seq=20 ttl=128 time=1.70 ms
64 bytes from 172.16.206.135: icmp_seq=21 ttl=128 time=1.31 ms
64 bytes from 172.16.206.135: icmp_seq=22 ttl=128 time=0.821 ms
64 bytes from 172.16.206.135: icmp_seq=23 ttl=128 time=1.99 ms
64 bytes from 172.16.206.135: icmp_seq=24 ttl=128 time=1.80 ms
64 bytes from 172.16.206.135: icmp_seq=25 ttl=128 time=1.06 ms
64 bytes from 172.16.206.135: icmp_seq=26 ttl=128 time=1.95 ms
64 bytes from 172.16.206.135: icmp_seq=27 ttl=128 time=1.66 ms
64 bytes from 172.16.206.135: icmp_seq=28 ttl=128 time=1.97 ms
64 bytes from 172.16.206.135: icmp_seq=29 ttl=128 time=1.69 ms
64 bytes from 172.16.206.135: icmp_seq=30 ttl=128 time=2.07 ms
64 bytes from 172.16.206.135: icmp_seq=31 ttl=128 time=2.22 ms
64 bytes from 172.16.206.135: icmp_seq=32 ttl=128 time=2.01 ms
64 bytes from 172.16.206.135: icmp_seq=33 ttl=128 time=1.45 ms
64 bytes from 172.16.206.135: icmp_seq=34 ttl=128 time=1.65 ms
64 bytes from 172.16.206.135: icmp_seq=35 ttl=128 time=1.83 ms
64 bytes from 172.16.206.135: icmp_seq=36 ttl=128 time=1.82 ms
64 bytes from 172.16.206.135: icmp_seq=37 ttl=128 time=1.68 ms
64 bytes from 172.16.206.135: icmp_seq=38 ttl=128 time=1.62 ms
64 bytes from 172.16.206.135: icmp_seq=39 ttl=128 time=1.73 ms
64 bytes from 172.16.206.135: icmp_seq=40 ttl=128 time=1.73 ms
64 bytes from 172.16.206.135: icmp_seq=41 ttl=128 time=1.73 ms
64 bytes from 172.16.206.135: icmp_seq=42 ttl=128 time=1.73 ms
64 bytes from 172.16.206.135: icmp_seq=43 ttl=128 time=1.34 ms
64 bytes from 172.16.206.135: icmp_seq=44 ttl=128 time=1.74 ms
64 bytes from 172.16.206.135: icmp_seq=45 ttl=128 time=1.48 ms
64 bytes from 172.16.206.135: icmp_seq=46 ttl=128 time=2.11 ms
```

Step 3: Accessing the Site from Windows VM

### Actions:

- ⑩ Open browser in Windows VM.

- ⑩ Fill the form with credentials (admin / password) and submit.



## Step 4: Starting Wireshark on Kali Linux

### Actions:

- ⑩ Launch Wireshark:
- ⑩ Select the active network interface (e.g., eth0, ens33)
- ⑩ Apply filter:

http

| No. | Time         | Source         | Destination    | Protocol | Length | Info  |
|-----|--------------|----------------|----------------|----------|--------|---|
| 77  | 1.126516253  | 172.16.206.132 | 142.250.77.227 | OCSP     | 488    | Request   |
| 79  | 1.247964589  | 142.250.77.227 | 172.16.206.132 | OCSP     | 1157   | Response  |
| 177 | 2.789055827  | 172.16.206.132 | 34.107.221.82  | HTTP     | 364    | GET /success.txt?ipv4 HTTP/1.1                                  |
| 179 | 2.845042834  | 34.107.221.82  | 172.16.206.132 | HTTP     | 270    | HTTP/1.1 200 OK (text/plain)                                    |
| 207 | 15.432626731 | 172.16.206.132 | 44.228.249.3   | HTTP     | 393    | GET / HTTP/1.1  |
| 209 | 15.992102052 | 44.228.249.3   | 172.16.206.132 | HTTP     | 2613   | HTTP/1.1 200 OK (text/html)                                     |
| 211 | 18.612876847 | 172.16.206.132 | 44.228.249.3   | HTTP     | 440    | GET /login.php HTTP/1.1   |
| 213 | 19.271180048 | 44.228.249.3   | 172.16.206.132 | HTTP     | 2802   | HTTP/1.1 200 OK (text/html)                                     |
| 232 | 33.762047781 | 172.16.206.132 | 44.228.249.3   | HTTP     | 578    | POST /userinfo.php HTTP/1.1 (application/x-www-form-urlencoded) |
| 234 | 34.436170182 | 44.228.249.3   | 172.16.206.132 | HTTP     | 2959   | HTTP/1.1 200 OK (text/html)                                     |

## Step 5: Capturing and Analyzing HTTP Credentials

### Actions:

1. In Wireshark, locate packet with POST /login HTTP/1.1
2. Right-click → Follow → **HTTP Stream**
3. Extract:

```

Frame 448: 578 bytes on wire (4624 bits), 578 bytes captured (4624 bits) on interface eth0
Ethernet II, Src: VMware 3e:06:8f (00:0c:29:3e:06:8f), Dst: VMware fb:b8:c8 (00:50:56:fb:b8:c8)
Internet Protocol Version 4, Src: 172.16.206.132, Dst: 44.228.249.3
Transmission Control Protocol, Src Port: 50366, Dst Port: 80, Seq: 726, Ack: 5308, Len: 578
Hypertext Transfer Protocol
  HTML Form URL Encoded: application/x-www-form-urlencoded
    Form item: "uname" = "test"
    Form item: "pass" = "test"
  
```

## Conclusion

This project shows that:

- ⑩ Unencrypted traffic is easily intercepted
- ⑩ Sensitive data must be protected using encryption
- ⑩ Organizations must move away from insecure legacy protocols

## Recommendations

- ⑩ Enforce use of **HTTPS**, **SFTP**, **SSH**
- ⑩ Deploy **TLS/SSL certificates** across all web services
- ⑩ Monitor and audit network activity regularly
- ⑩ Educate users about secure communication

