Experiment No. 9

Title: Network Sniffing - Wireshark

Batch: B-2 **Roll No.: 16010422234** **Experiment No.: 9**

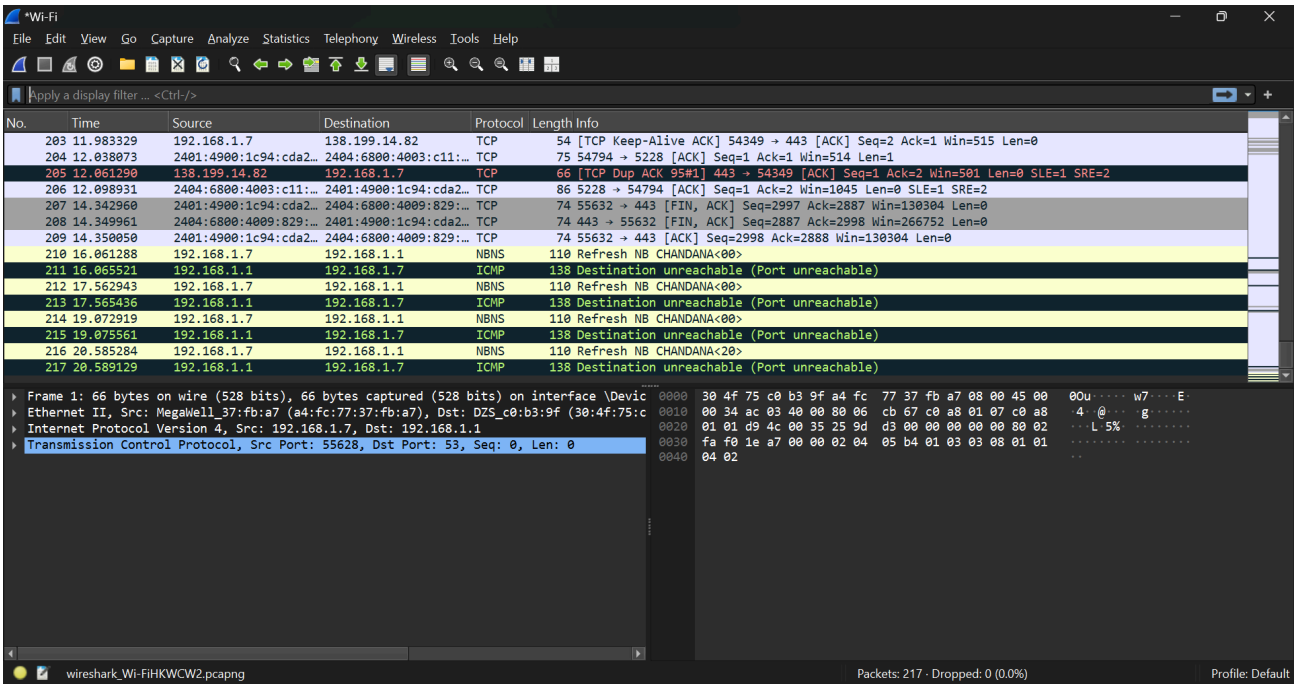
Aim: To perform network sniffing using wire shark tool

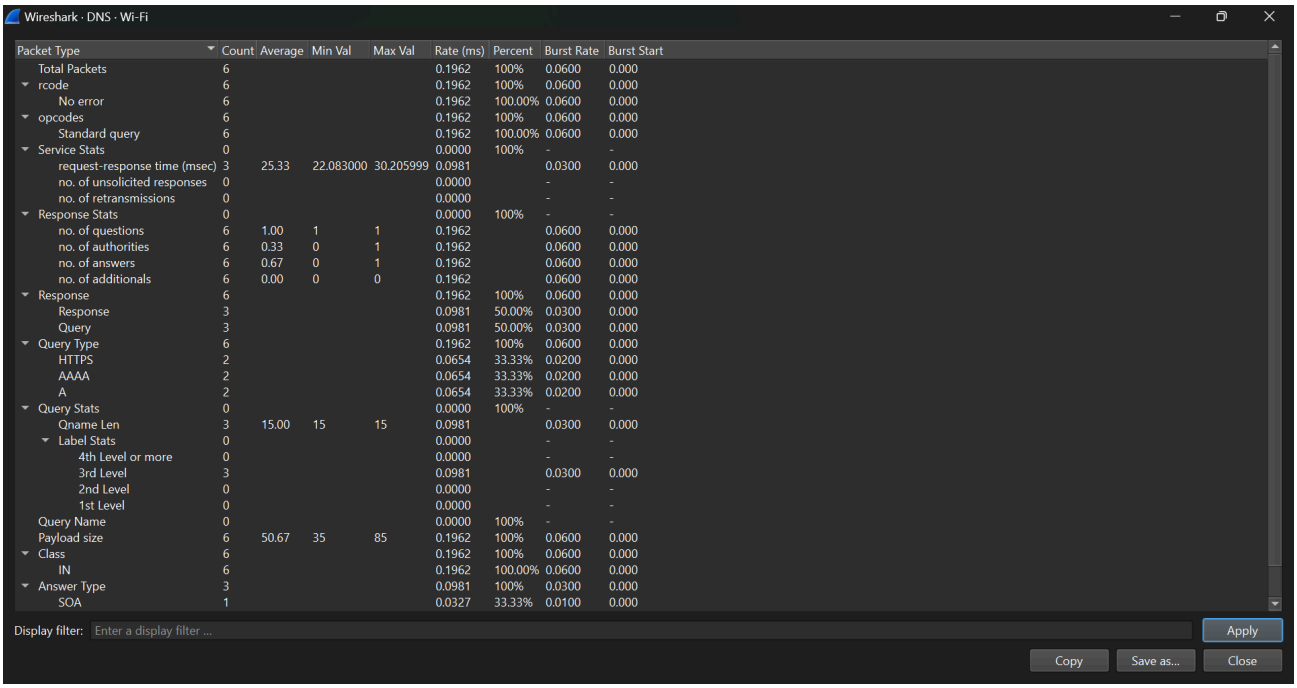
Questions:

1. **What is the difference between Burp suite and Wire shark tools?**
   - ➔ Wireshark is primarily a network protocol analyzer used for network troubleshooting, analysis, software and protocol development, and education. It captures and displays packets in real-time and can interpret the data flowing in and out of a network.
   - ➔ Burp Suite, on the other hand, is more focused on web application security testing. It functions as an intercepting proxy, allowing a user to see and modify the traffic between a browser and the server. It is used for security testing of web applications and can manipulate web traffic to test the security parameters of the application.

2. **Suggest the methods and/or security mechanisms to protect the password being leaked using tools like wireshark.**
   - ➔ Use Encryption: Employ TLS/SSL for all sensitive communications. Encrypted traffic cannot be easily read by packet sniffers if intercepted.
   - ➔ Secure Authentication Protocols: Implement secure authentication mechanisms such as OAuth, which do not transmit passwords directly.
   - ➔ HTTPS: Ensure that all login pages and data transmissions use HTTPS to secure web traffic.
   - ➔ Strong Password Policies: Enforce complex passwords that are difficult to decode even if packet data is captured.
   - ➔ Regular Updates and Patches: Keep all network software and devices updated to protect against vulnerabilities that might be exploited to capture network traffic.

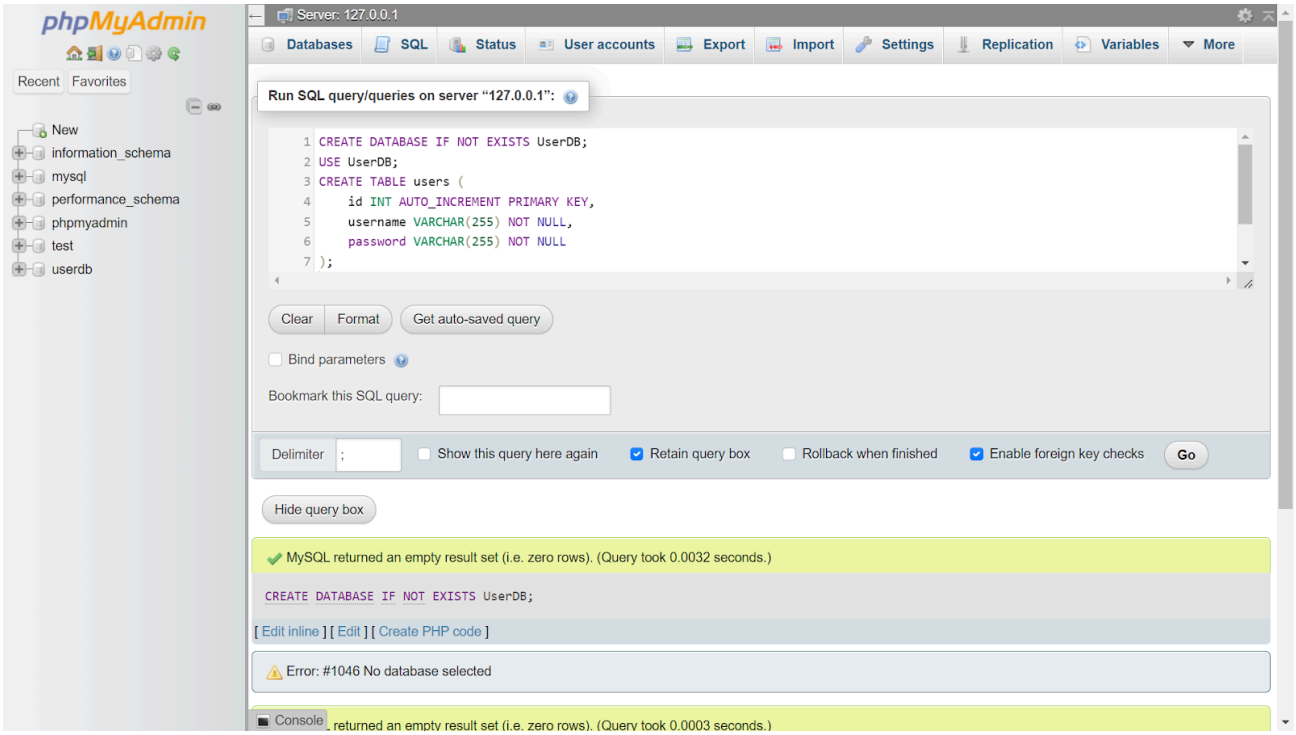
Result:






```
CREATE DATABASE IF NOT EXISTS UserDB;
USE UserDB;
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
```

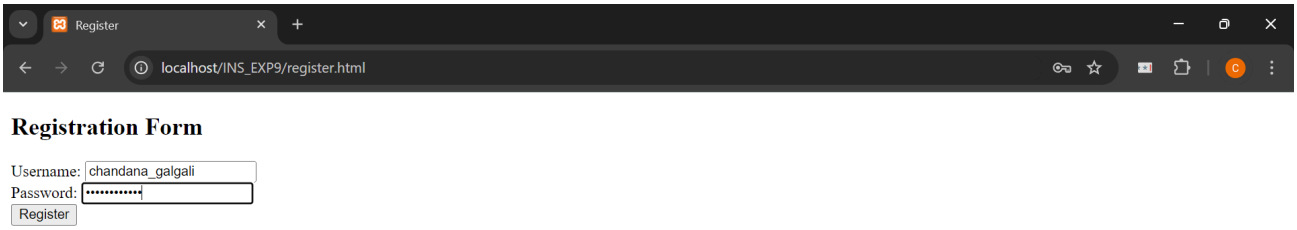
```
    username VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL
);
```



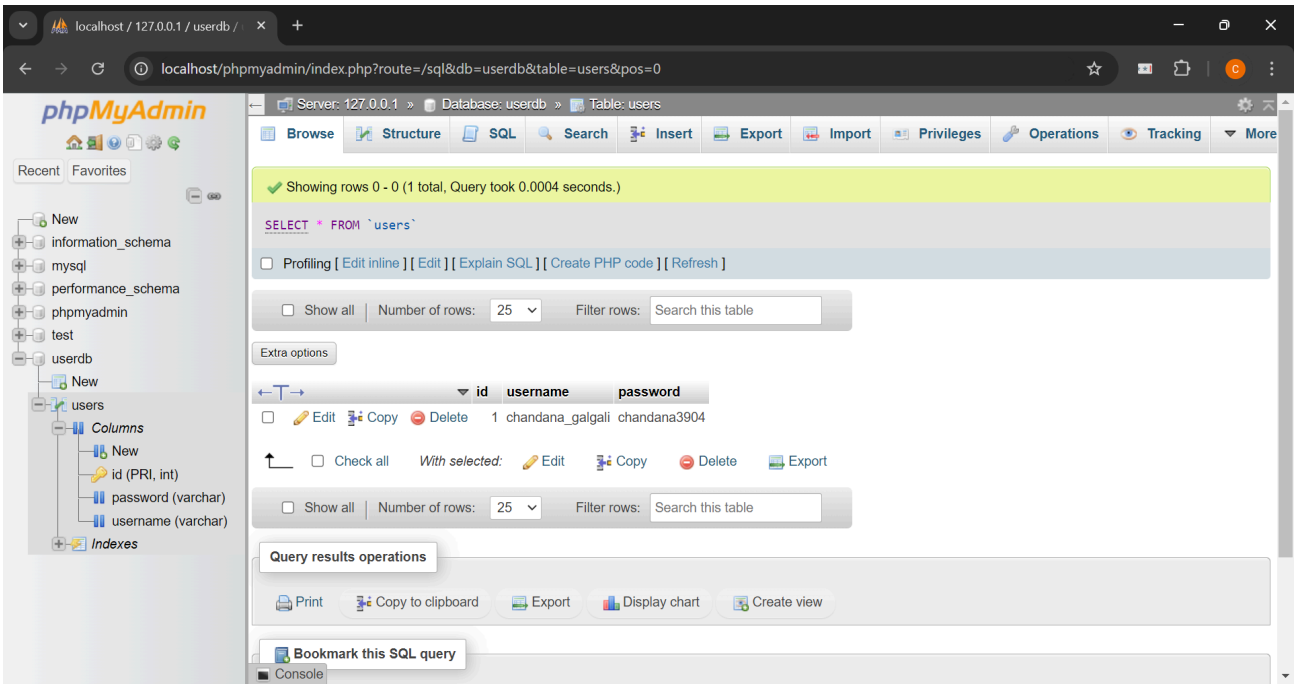

```
<!DOCTYPE html>
<html>
<head>
    <title>Register</title>
</head>
<body>
    <h2>Registration Form</h2>
    <form action="register.php" method="get">
        Username: <input type="text" name="username"><br>
        Password: <input type="password" name="password"><br>
        <input type="submit" value="Register">
    </form>
</body>
</html>
```

```
<?php
$servername = "localhost";
```

```php
$username = "root";  // Update your database username
$password = "";  // Update your database password
$dbname = "UserDB";

$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Insecure: Password should be hashed in a real application
$sql = "INSERT INTO users (username, password)
VALUES ('".$_GET['username']."', '".$_GET['password']."')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```



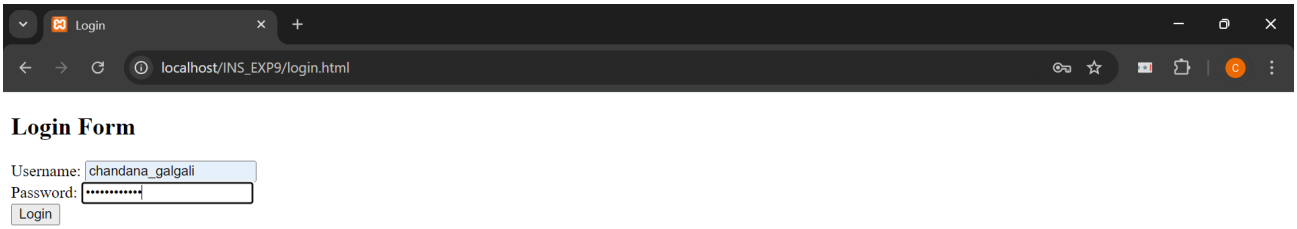
New record created successfully

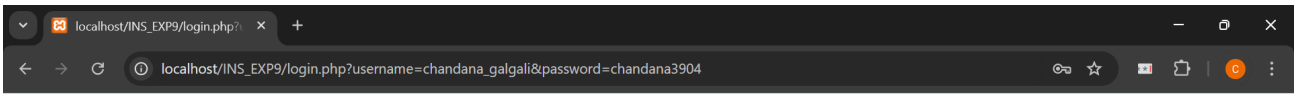


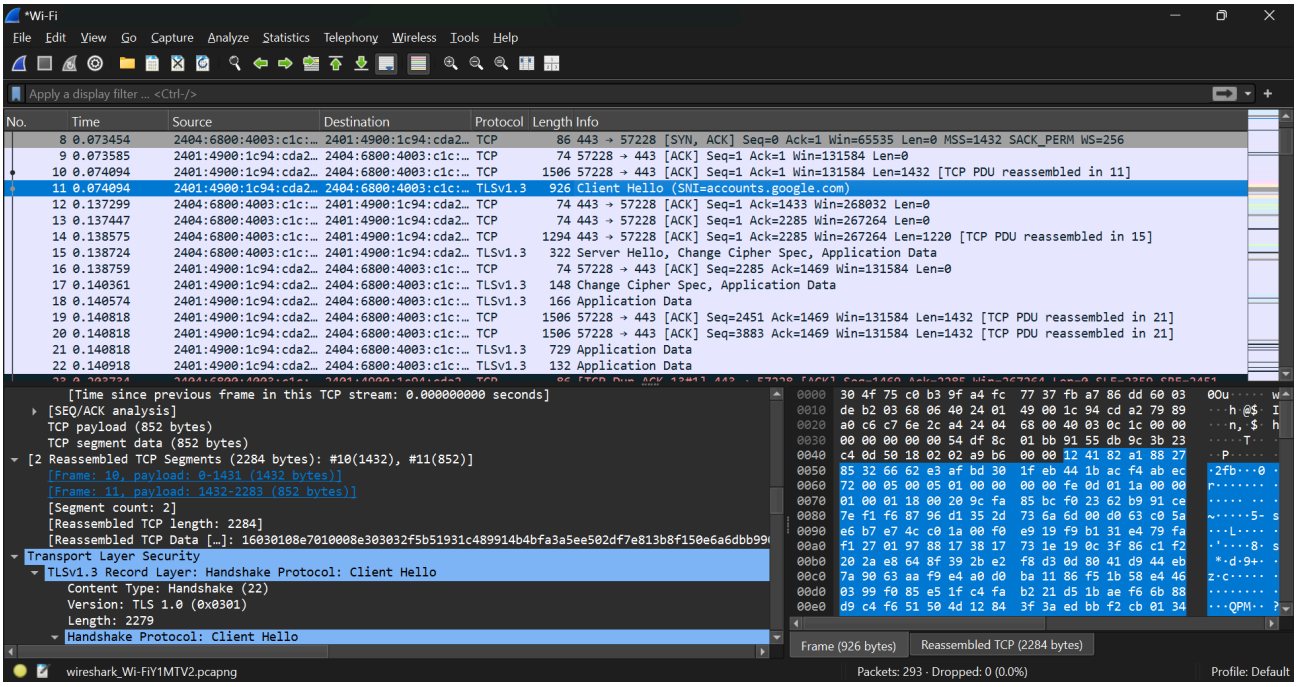

```
<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
</head>
<body>
    <h2>Login Form</h2>
```

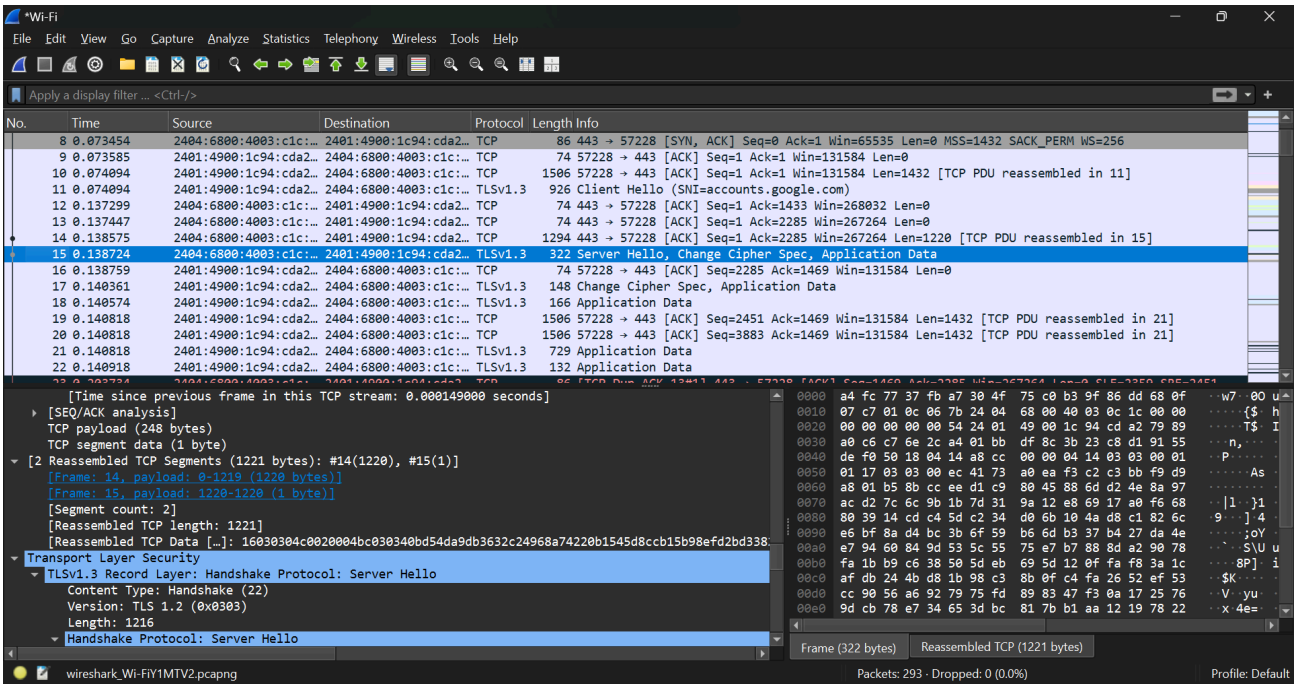
```
    <form action="login.php" method="get">
        Username: <input type="text" name="username"><br>
        Password: <input type="password" name="password"><br>
        <input type="submit" value="Login">
    </form>
</body>
</html>
```

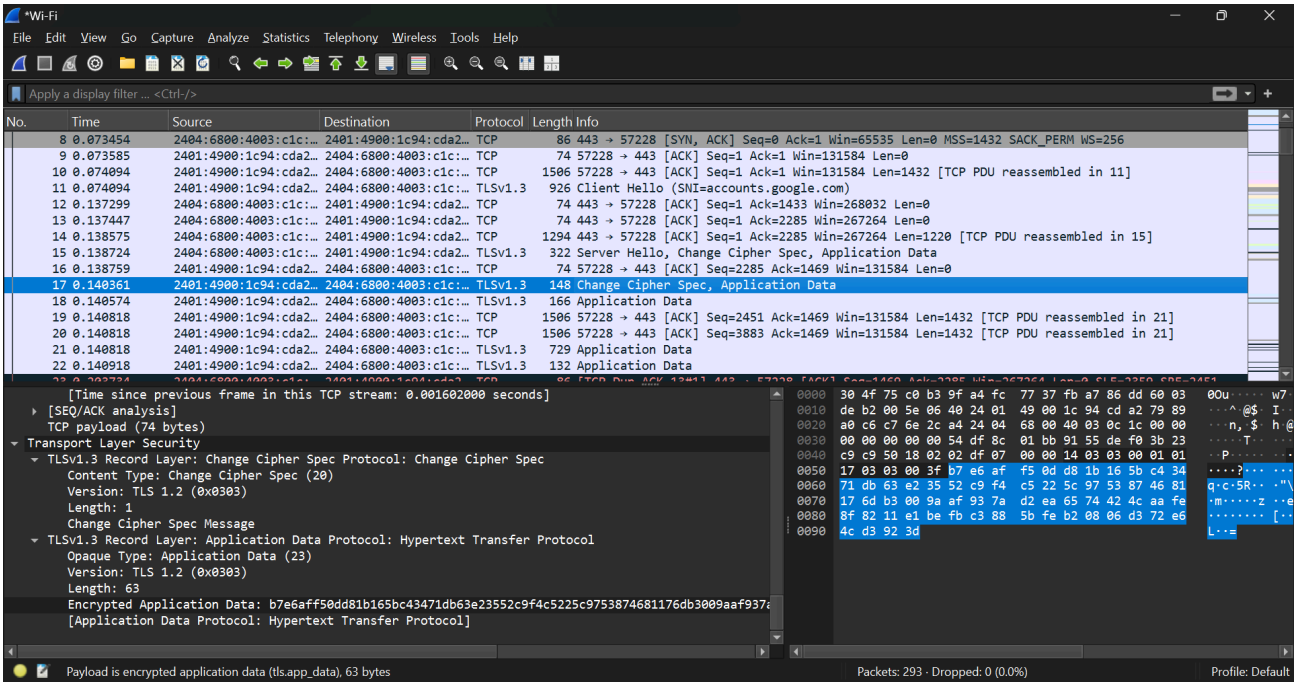
```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "UserDB";

$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT id FROM users WHERE username = '".$_GET['username']."' AND
password = '".$_GET['password']."'";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    echo "Login successful";
} else {
    echo "Invalid username or password";
}

$conn->close();
?>
```

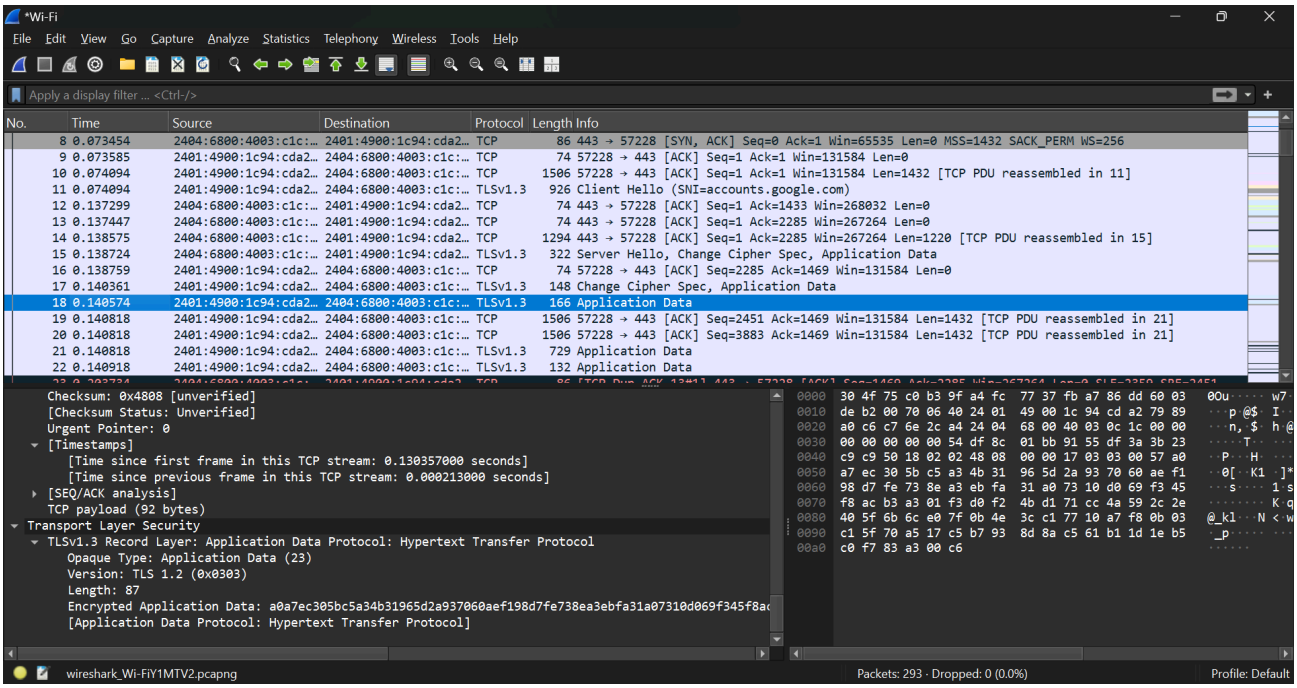

Login
localhost/INS_EXP9/login.html
Login Form
Username: chandana_galgali
Password:
Login


localhost/INS_EXP9/login.php?
localhost/INS_EXP9/login.php?username=chandana_galgali&password=chandana3904
Login successful


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
8 0.073454 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 86 443 → 57228 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1432 SACK_PERM WS=256
9 0.073585 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 74 57228 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
10 0.074094 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 1506 57228 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=1432 [TCP PDU reassembled in 11]
11 0.074094 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 926 Client Hello (SNI=accounts.google.com)
12 0.137299 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 74 443 → 57228 [ACK] Seq=1 Ack=1433 Win=268032 Len=0
13 0.137447 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 74 443 → 57228 [ACK] Seq=1 Ack=2285 Win=267264 Len=0
14 0.138575 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 1294 443 → 57228 [ACK] Seq=1 Ack=2285 Win=267264 Len=1220 [TCP PDU reassembled in 15]
15 0.138724 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TLSv1.3 322 Server Hello, Change Cipher Spec, Application Data
16 0.138759 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 74 57228 → 443 [ACK] Seq=2285 Ack=1469 Win=131584 Len=0
17 0.140361 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 148 Change Cipher Spec, Application Data
18 0.140574 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 166 Application Data
19 0.140818 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 1506 57228 → 443 [ACK] Seq=2451 Ack=1469 Win=131584 Len=1432 [TCP PDU reassembled in 21]
20 0.140818 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 1506 57228 → 443 [ACK] Seq=3883 Ack=1469 Win=131584 Len=1432 [TCP PDU reassembled in 21]
21 0.140818 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 729 Application Data
22 0.140918 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 132 Application Data
[Time since previous frame in this TCP stream: 0.000000000 seconds]
[SEQ/ACK analysis]
TCP payload (852 bytes)
TCP segment data (852 bytes)
[2 Reassembled TCP Segments (2284 bytes): #10(1432), #11(852)]
[Frame: 10, payload: 0-1431 (1432 bytes)]
[Frame: 11, payload: 1432-2283 (852 bytes)]
[Segment count: 2]
[Reassembled TCP length: 2284]
[Reassembled TCP Data [...]: 16030108e7010008e303032f5b51931c489914b4bfa3a5ee502df7e813b8f150e6a6dbb99
Transport Layer Security
TLSv1.3 Record Layer: Handshake Protocol: Client Hello
Content Type: Handshake (22)
Version: TLS 1.0 (0x0301)
Length: 2279
Handshake Protocol: Client Hello
0000 30 4f 75 c0 b3 9f a4 fc 77 37 fb a7 86 dd 60 03
0010 de b2 03 68 06 40 24 01 49 00 1c 94 cd a2 79 89
0020 a0 c6 c7 6e 2c a4 24 04 68 00 40 03 0c 1c 00 00
0030 00 00 00 00 00 54 df 8c 01 bb 91 55 db 9c 3b 23
0040 c4 0d 50 18 02 02 a9 b6 00 00 12 41 82 a1 88 27
0050 85 32 66 62 e3 af bd 30 1f eb 44 1b ac f4 ab ec
0060 72 00 05 00 05 01 00 00 00 00 fe 0d 01 1a 00 00
0070 01 00 01 18 00 20 9c fa 85 bc f0 23 62 b9 91 ce
0080 7e f1 f6 87 96 d1 35 2d 73 6a 6d 00 d0 63 c0 5a
0090 e6 b7 e7 4c c0 1a 00 f0 e9 19 f9 b1 31 e4 79 fa
00a0 f1 27 01 97 88 17 38 17 73 1e 19 0c 3f 86 c1 f2
00b0 20 2a e8 64 8f 39 2b e2 f8 d3 0d 80 41 d9 44 eb
00c0 7a 90 63 aa f9 e4 a0 d0 ba 11 86 f5 1b 58 e4 46
00d0 03 99 f0 85 e5 1f c4 fa b2 21 d5 1b ae f6 6b 88
00e0 d9 c4 f6 51 50 4d 12 84 3f 3a ed bb f2 cb 01 34
Frame (926 bytes)
Reassembled TCP (2284 bytes)
wireshark_Wi-FiY1MTV2.pcapng
Packets: 293 · Dropped: 0 (0.0%)
Profile: Default


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
8 0.073454 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 86 443 → 57228 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1432 SACK_PERM WS=256
9 0.073585 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 74 57228 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=0
10 0.074094 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 1506 57228 → 443 [ACK] Seq=1 Ack=1 Win=131584 Len=1432 [TCP PDU reassembled in 11]
11 0.074094 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 926 Client Hello (SNI=accounts.google.com)
12 0.137299 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 74 443 → 57228 [ACK] Seq=1 Ack=1433 Win=268032 Len=0
13 0.137447 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 74 443 → 57228 [ACK] Seq=1 Ack=2285 Win=267264 Len=0
14 0.138575 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TCP 1294 443 → 57228 [ACK] Seq=1 Ack=2285 Win=267264 Len=1220 [TCP PDU reassembled in 15]
15 0.138724 2404:6800:4003:c1c:... 2401:4900:1c94:cda2... TLSv1.3 322 Server Hello, Change Cipher Spec, Application Data
16 0.138759 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 74 57228 → 443 [ACK] Seq=2285 Ack=1469 Win=131584 Len=0
17 0.140361 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 148 Change Cipher Spec, Application Data
18 0.140574 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 166 Application Data
19 0.140818 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 1506 57228 → 443 [ACK] Seq=2451 Ack=1469 Win=131584 Len=1432 [TCP PDU reassembled in 21]
20 0.140818 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TCP 1506 57228 → 443 [ACK] Seq=3883 Ack=1469 Win=131584 Len=1432 [TCP PDU reassembled in 21]
21 0.140818 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 729 Application Data
22 0.140918 2401:4900:1c94:cda2... 2404:6800:4003:c1c:... TLSv1.3 132 Application Data
[Time since previous frame in this TCP stream: 0.000149000 seconds]
[SEQ/ACK analysis]
TCP payload (248 bytes)
TCP segment data (1 byte)
[2 Reassembled TCP Segments (1221 bytes): #14(1220), #15(1)]
[Frame: 14, payload: 0-1219 (1220 bytes)]
[Frame: 15, payload: 1220-1220 (1 byte)]
[Segment count: 2]
[Reassembled TCP length: 1221]
[Reassembled TCP Data [...]: 16030304c0020004bc030340bd54da9db3632c24968a74220b1545d8ccb15b98efd2bd338
Transport Layer Security
TLSv1.3 Record Layer: Handshake Protocol: Server Hello
Content Type: Handshake (22)
Version: TLS 1.2 (0x0303)
Length: 1216
Handshake Protocol: Server Hello
0000 a4 fc 77 37 fb a7 30 4f 75 c0 b3 9f 86 dd 68 0f
0010 07 c7 01 0c 06 7b 24 04 68 00 40 03 0c 1c 00 00
0020 00 00 00 00 00 54 24 01 49 00 1c 94 cd a2 79 89
0030 a0 c6 c7 6e 2c a4 01 bb df 8c 3b 23 c8 d1 91 55
0040 de f0 50 18 04 14 a8 cc 00 00 04 14 03 03 00 01
0050 01 17 03 03 00 ec 41 73 a0 ea f3 c2 c3 bb f9 d9
0060 a8 01 b5 8b cc ee d1 c9 80 45 88 6d d2 4e 8a 97
0070 ac d2 7c 6c 9b 1b 7d 31 9a 12 e8 69 17 a0 f6 68
0080 80 39 14 cd c4 5d c2 34 d0 6b 10 4a d8 c1 82 6c
0090 e6 bf 8a d4 bc 3b 6f 59 b6 6d b3 37 b4 27 da 4e
00a0 e7 94 60 84 9d 53 5c 55 75 e7 b7 88 8d a2 90 78
00b0 fa 1b b9 c6 38 50 5d eb 69 5d 12 0f fa f8 3a 1c
00c0 af db 24 4b d8 1b 98 c3 8b 0f c4 fa 26 52 ef 53
00d0 cc 90 56 a6 92 79 75 fd 89 83 47 f3 0a 17 25 76
00e0 9d cb 78 e7 34 65 3d bc 81 7b b1 aa 12 19 78 22
Frame (322 bytes)
Reassembled TCP (1221 bytes)
wireshark_Wi-FiY1MTV2.pcapng
Packets: 293 · Dropped: 0 (0.0%)
Profile: Default





Outcomes: Understand Security issues related to Software, Web and Networks.

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)

The Network Sniffing experiment using Wireshark provided profound insights into the workings of network packets and their flow across various interfaces. By observing the transmission of data in real-time, students gain a hands-on understanding of how protocols interact and how data moves through a network. This experiment underlines the importance of network security measures and the role of sophisticated tools like Wireshark in diagnosing, troubleshooting, and ensuring the integrity of network data. The skills acquired from this practical application are crucial for future network administrators and cybersecurity professionals in safeguarding against potential network vulnerabilities and attacks.
