UDP client-server demo

Eavesdrop attack -
- Develop a UDP client server program using any programming language
- Use TCP -dump utility to capture the data transfer between the client and server

Below is a **simple and clean implementation** of the assignment:

✅ EAVESDROP ATTACK DEMO (UDP Client–Server + tcpdump)

This assignment shows how UDP packets can be **captured (eavesdropped)** using `tcpdump`.

1️⃣ UDP SERVER CODE (Python)

```python
# udp_server.py
import socket

def main():
    host = "0.0.0.0"
    port = 5000

    server_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    server_socket.bind((host, port))

    print(f"UDP Server running on port {port}...")

    while True:
        data, addr = server_socket.recvfrom(1024)
        print(f"Received from {addr}: {data.decode()}")

        server_socket.sendto(b"Message received", addr)

if __name__ == "__main__":
    main()
```

2️⃣ UDP CLIENT CODE (Python)

```python
# udp_client.py
import socket

def main():
    server_ip = "127.0.0.1"
    server_port = 5000

    client_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    message = input("Enter message: ")
    client_socket.sendto(message.encode(), (server_ip, server_port))

    data, addr = client_socket.recvfrom(1024)
    print("Server response:", data.decode())

if __name__ == "__main__":
    main()
```

3️⃣ RUNNING THE PROGRAM

Step 1 — Start Server

```bash
python3 udp_server.py
```

Step 2 — Run Client in another terminal

```bash
python3 udp_client.py
```

Send a message like:

```nginx
Hello server
```

4️⃣ CAPTURE TRAFFIC USING tcpdump (Eavesdropping)

Open a 3rd terminal and run:

```bash
sudo tcpdump -i any udp port 5000 -A
```

Explanation:

- **`-i any`** → captures on all interfaces
- **`udp port 5000`** → filter only our UDP packets
- **`-A`** → print payload in ASCII (readable)

You will see something like:

```nginx
IP 127.0.0.1.40422 > 127.0.0.1.5000: UDP, length 12
Hello server
```

✔️ This shows the **actual message content** → **EAVESDROP SUCCESSFUL**
✔️ UDP is *not encrypted*, so attackers can read your traffic.

5️⃣ OUTPUT EXAMPLE FOR ASSIGNMENT

You can include this in your report:

Client Sends:

```arduino
Hello Server
```

tcpdump captured:

```arduino
12:41:22.123456 IP 192.168.1.5.40422 > 192.168.1.10.5000: UDP, length 12
Hello Server
```

Conclusion:

Since UDP has **no encryption**, an attacker can capture and read packets using simple tools like **tcpdump**.

If you want, I can generate:
✅ PDF of this experiment
✅ Handwritten-style notes
✅ Diagram of attack flow

Just tell me!

give me code in java

Here is the **simplest Java implementation** for your assignment:

✅ Eavesdrop Attack – UDP Client–Server Program in Java

1️⃣ UDP SERVER (Java)

Create a file named **UDPServer.java**

```java
import java.net.DatagramPacket;
import java.net.DatagramSocket;

public class UDPServer {
    public static void main(String[] args) {
        try {
            DatagramSocket serverSocket = new DatagramSocket(5000);
            System.out.println("UDP Server is running on port 5000...");

            byte[] receiveBuffer = new byte[1024];
            byte[] sendBuffer;

            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer, receiveBuffer.length);
                serverSocket.receive(receivePacket);

                String message = new String(receivePacket.getData(), 0, receivePacket.getLength());
                System.out.println("Received: " + message);

                String reply = "Message received";
                sendBuffer = reply.getBytes();

                DatagramPacket sendPacket = new DatagramPacket(
                        sendBuffer,
                        sendBuffer.length,
                        receivePacket.getAddress(),
                        receivePacket.getPort()
                );
                serverSocket.send(sendPacket);
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2️⃣ UDP CLIENT (Java)

Create a file named **UDPClient.java**

```java
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;

public class UDPClient {
    public static void main(String[] args) {
        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress serverAddress = InetAddress.getByName("127.0.0.1");
            int serverPort = 5000;

            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter message: ");
            String message = scanner.nextLine();

            byte[] sendBuffer = message.getBytes();
            DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, serverAddress, serverPort);
            clientSocket.send(sendPacket);
            byte[] receiveBuffer = new byte[1024];
            DatagramPacket receivePacket = new DatagramPacket(receiveBuffer,
receiveBuffer.length);
            clientSocket.receive(receivePacket);

            String reply = new String(receivePacket.getData(), 0, receivePacket.getLength());
            System.out.println("Server response: " + reply);

            clientSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

3️⃣ RUNNING THE PROGRAM

Start server

```bash
javac UDPServer.java
java UDPServer
```

Run client in new terminal

```bash
javac UDPClient.java
java UDPClient
```

Enter any message you want.

4️⃣ CAPTURE TRAFFIC USING tcpdump

Run this in a third terminal:

```bash
sudo tcpdump -i any udp port 5000 -A
```

You will clearly see:

```arduino
Hello Server
```

✔️ This demonstrates an **eavesdropping attack on UDP**.

If you want:
✅ A **PDF** with handwritten-style notes

✅ **Diagram** of the attack
✅ **Full assignment explanation**

Just tell me!

whre is attaker

Here’s how the **attacker fits into the assignment**, and where you should place them in your report or demo.

✅ Where Is the Attacker in This Eavesdrop Attack?

In your assignment, the **attacker** is simply **another machine (or terminal)** that **captures packets** traveling between the client and the server.

✔️ Attacker = tcpdump user

The attacker is the person running this command:

```bash
sudo tcpdump -i any udp port 5000 -A
```

This attacker:

- Does **not** modify any data.
- Does **not** need to connect to the server.
- Simply **listens and captures** the communication.

This is called **passive eavesdropping**.

🔥 Visual Representation

```sql
+----------------+       UDP Packets       +----------------+
|    CLIENT      | ----------------------> |     SERVER     |
| (UDPClient)    | <---------------------- | (UDPServer)    |
+----------------+                         +----------------+
            \                                      /
             \                                    /
         (attacker listens using tcpdump)
                       |
                       v
               +----------------+
               |   ATTACKER     |
               | tcpdump/sniff  |
               +----------------+
```

The attacker **sniffs** the network and reads the messages like:

```arduino
Hello Server
Message received
```

🕶️ How Attacker Works in Your Demo

1️⃣ Run the server

2️⃣ Run the client

3️⃣ Run the attacker (tcpdump)

Attacker terminal:

```bash
sudo tcpdump -i any udp port 5000 -A
```

You will see clear-text packets, proving:

✔️ **UDP is not secure**
✔️ **Anyone on the network can read the message**

💡 If you want, I can give you:

- Attack flow diagram (PDF)
- Handwritten-style notes
- Complete assignment explanation (Aim, Theory, Algorithm, Code, Output, Result)
- Java code with attacker code section included

Just tell me!