Wireshark for Pentester: Password Sniffing

April 28, 2021 By Raj Chandel

Many people wonder if Wireshark can capture passwords. The answer is undoubtedly yes! Wireshark can capture not only passwords, but any type of data passing through a network – usernames, email addresses, personal information, pictures, videos, or anything else. Wireshark can sniff the passwords passing through as long as we can capture network traffic. But the question is, what kind of passwords are they? Or, more precisely, which network protocols' passwords can we obtain? That is the subject of this article.

Table of Contents

Plain text network protocols

So, how is it possible for Wireshark to capture passwords? This is due to the fact that some network protocols do not use encryption. These protocols are referred to as clear text (or plain text) protocols. Because clear text protocols do not encrypt communication, all data, including passwords, is visible to the naked eye. Anyone who is in a position to see the communication (for example, a man in the middle) can eventually see everything.

In the sections that follow, we'll take a closer look at these protocols and see examples of captured passwords using Wireshark.

Disclaimer: To protect client data, all screenshots have been censored and/or modified.

Trace Files

To, get hands-on with these labs you can download all the trace files from **here**.

1. Capture HTTP Password
2. Monitoring HTTPS Packets over SSL or TLS
3. Capture Telnet Password
4. Capture FTP Password
5. Capture SMTP Password
6. Analyzing SNMP Community String
7. Capture MSSQL Password
8. Capture PostgreSQL Password

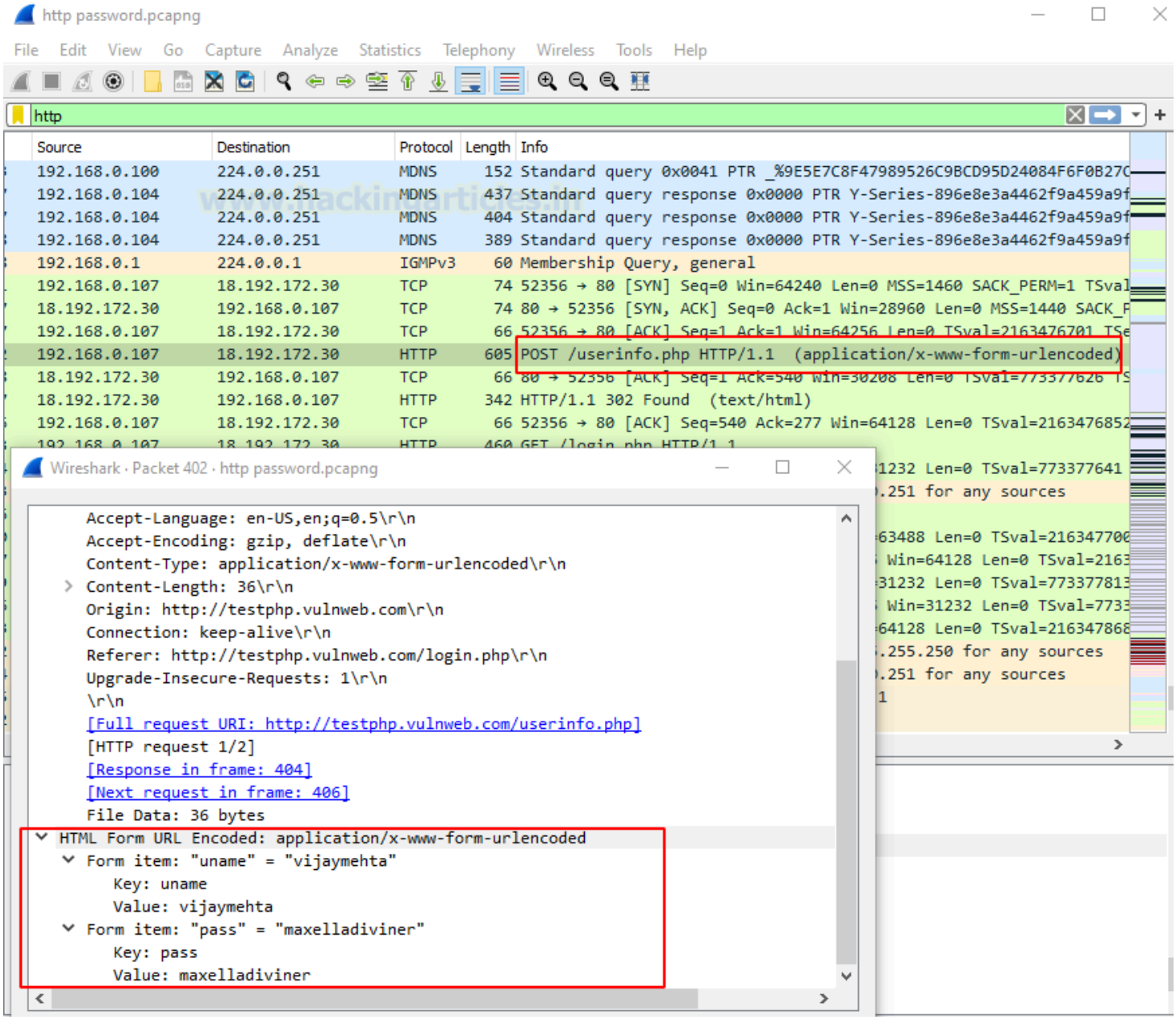
Source of some of the trace files: – **Wireshark.org**

Capture HTTP Password

No introduction is certainly needed for the Hypertext Transfer Protocol (HTTP). It usually works on port 80/TCP, and as it is a text protocol, it does not give the communication parties much or no privacy. Anyone who's able to communicate can catch everything, including passwords, via that channel.

While all major browser vendors have made considerable efforts to prevent the use of HTTP as far as possible, during penetration testing, HTTP can be used on internal media.

Here is an example of login credentials captured in a POST request in an HTTP communication:

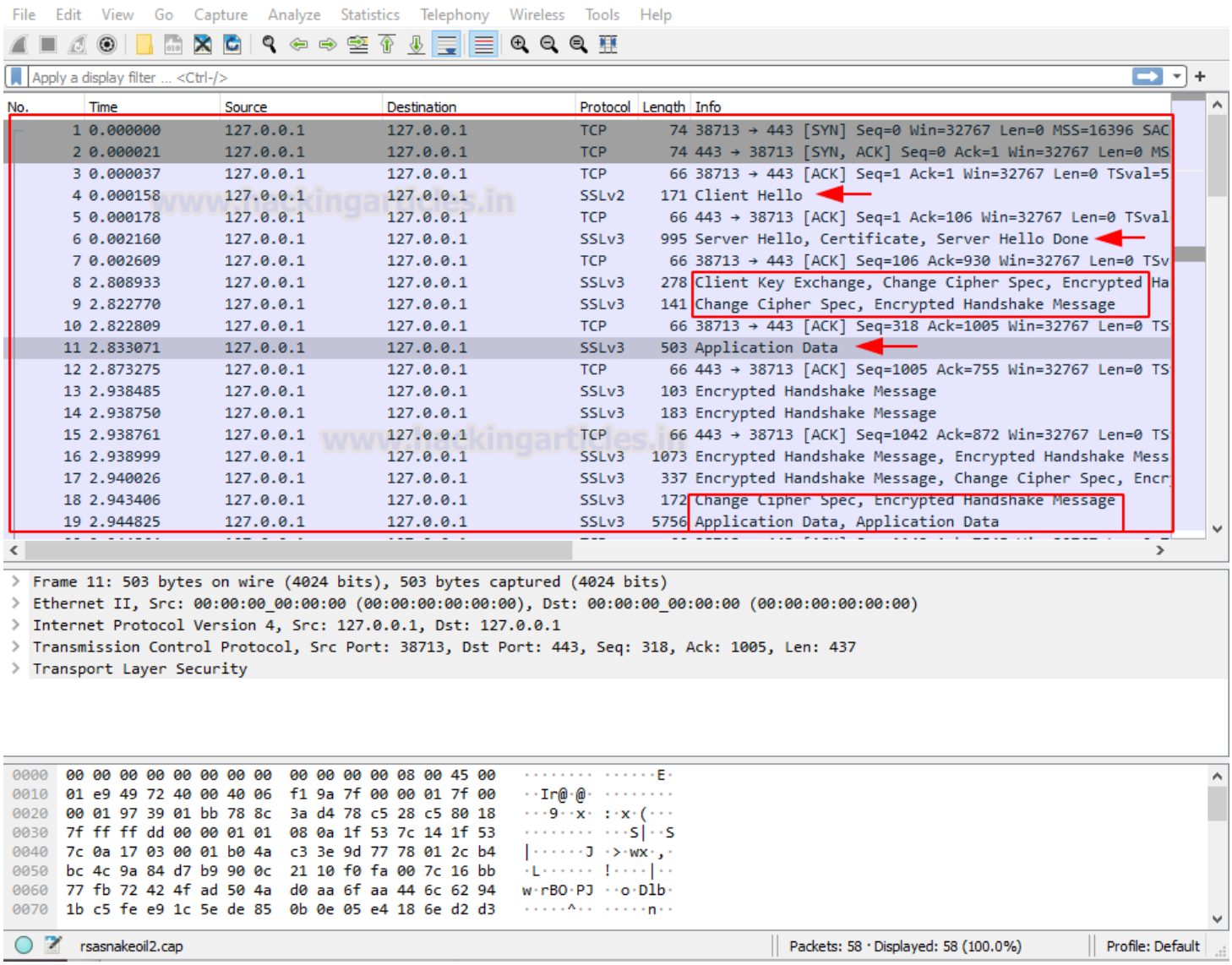


Monitoring HTTPS packets over SSL or TLS

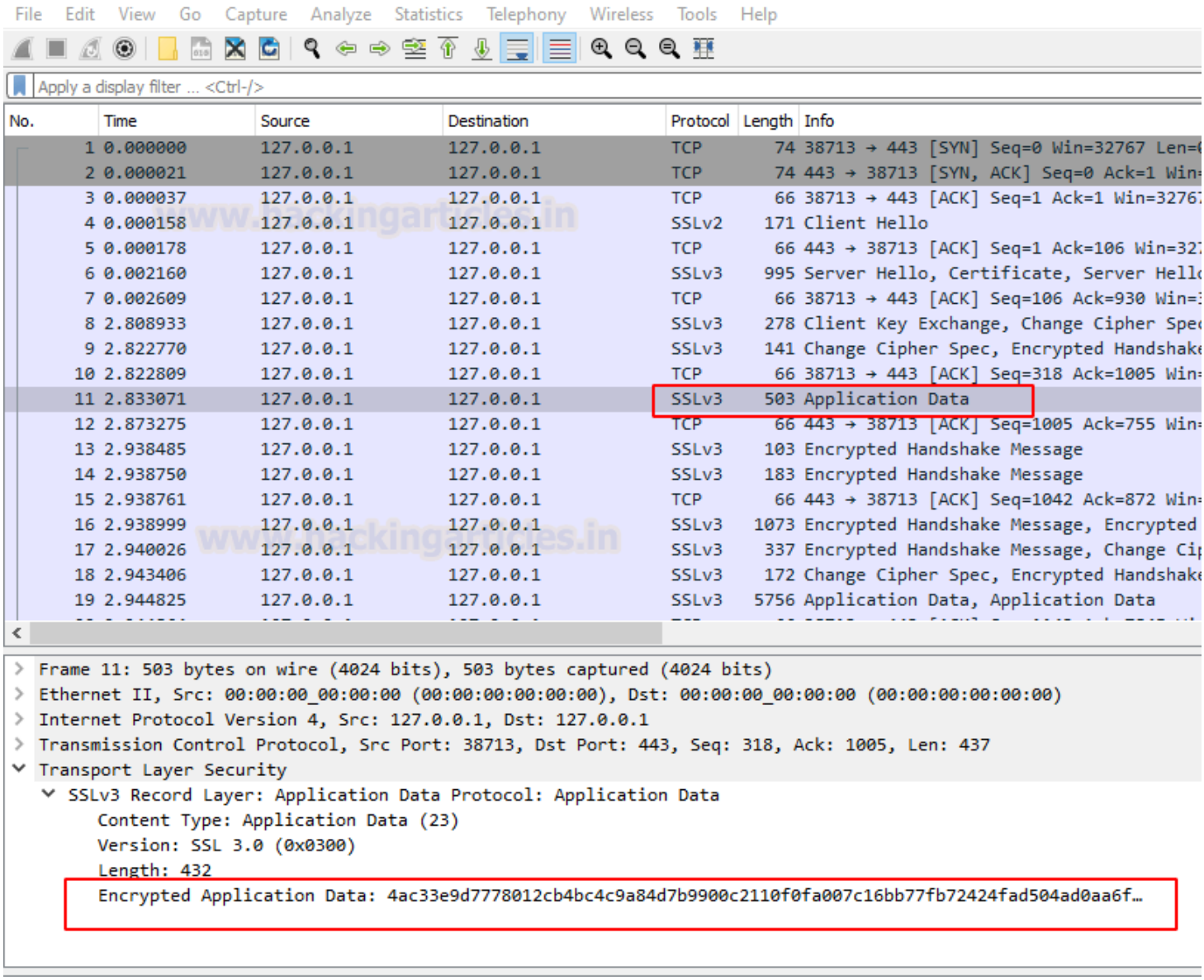
Dissect HTTPS Packet Captures

Open the provided HTTPS/TLS.pcapng file. Where you can see

- The 3-way handshake is happening
- Hello from SSL Client and the ACK from server
- Server Hello and then ACK
- Exchanging some key and Cipher information
- Started Exchanging Data

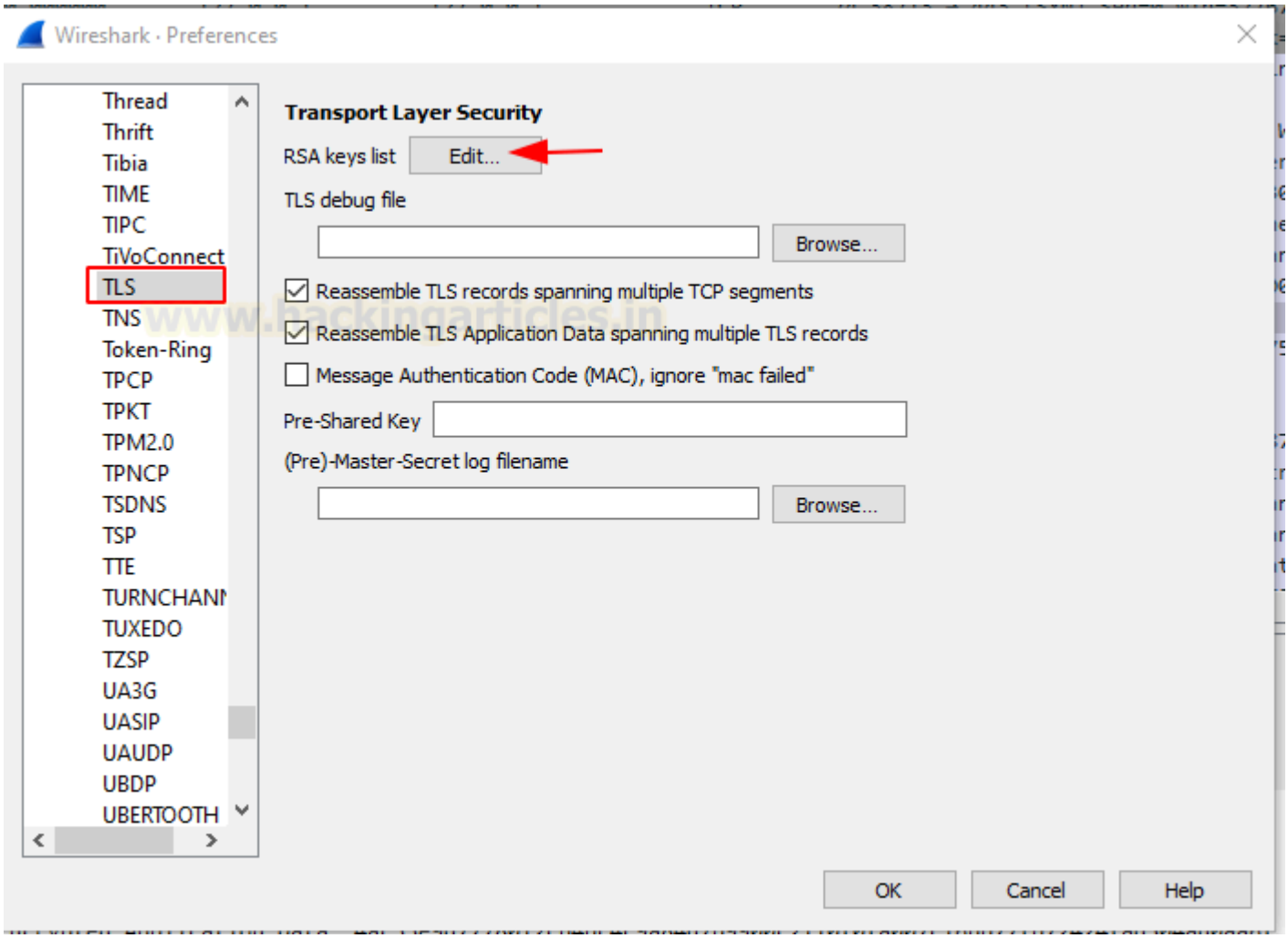


Then, if we click on any application data, that data is unreadable to us. However, with Wireshark, we can decrypt that data… all we need is the server's Private Key. ***Don't worry we have already provided the key along with the PCAP file.***



To Decrypt the Encrypted Application Data over TLS or SSL Navigate to

Edit > Preference > Protocol > TLS

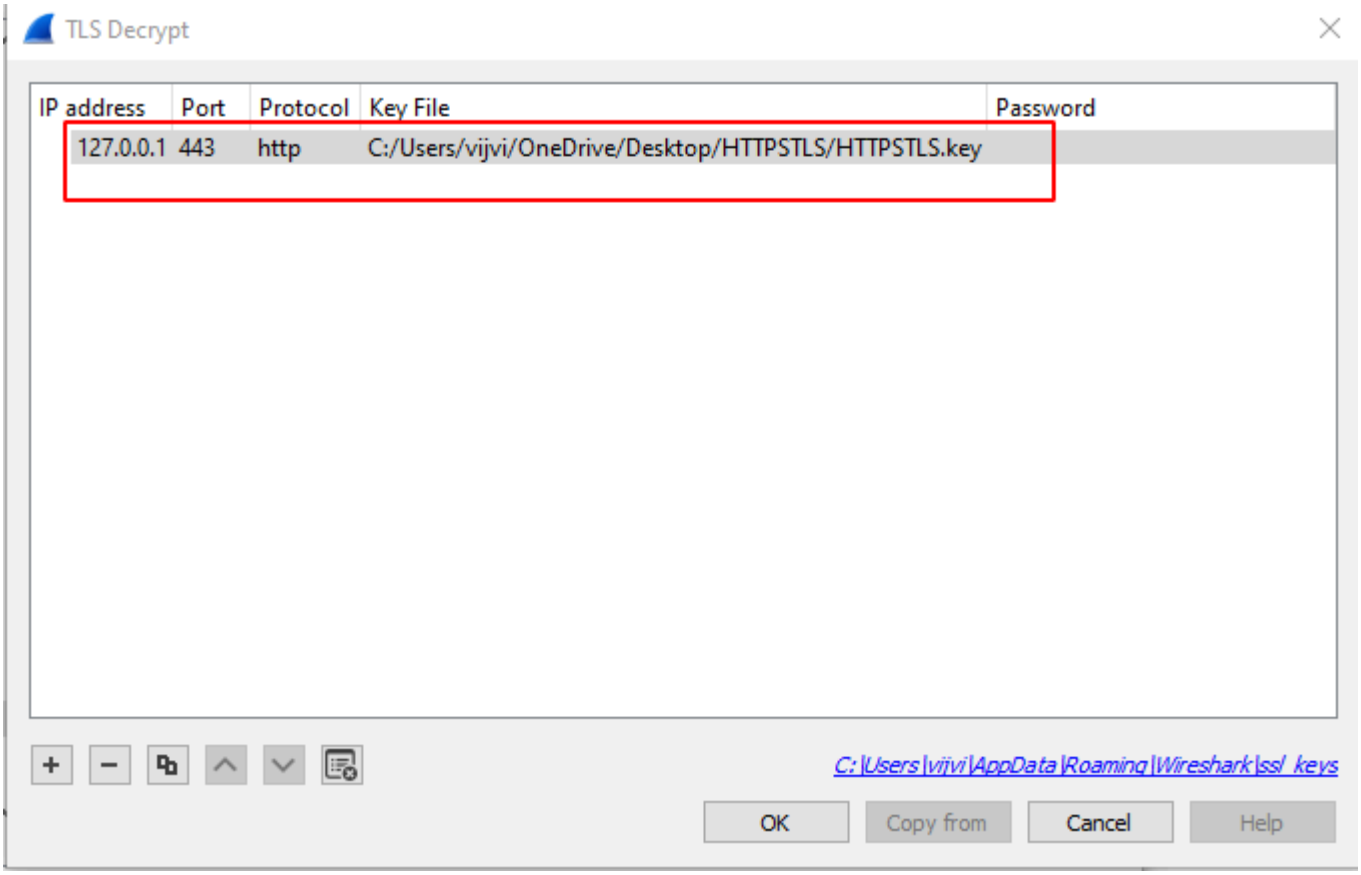


And add these values

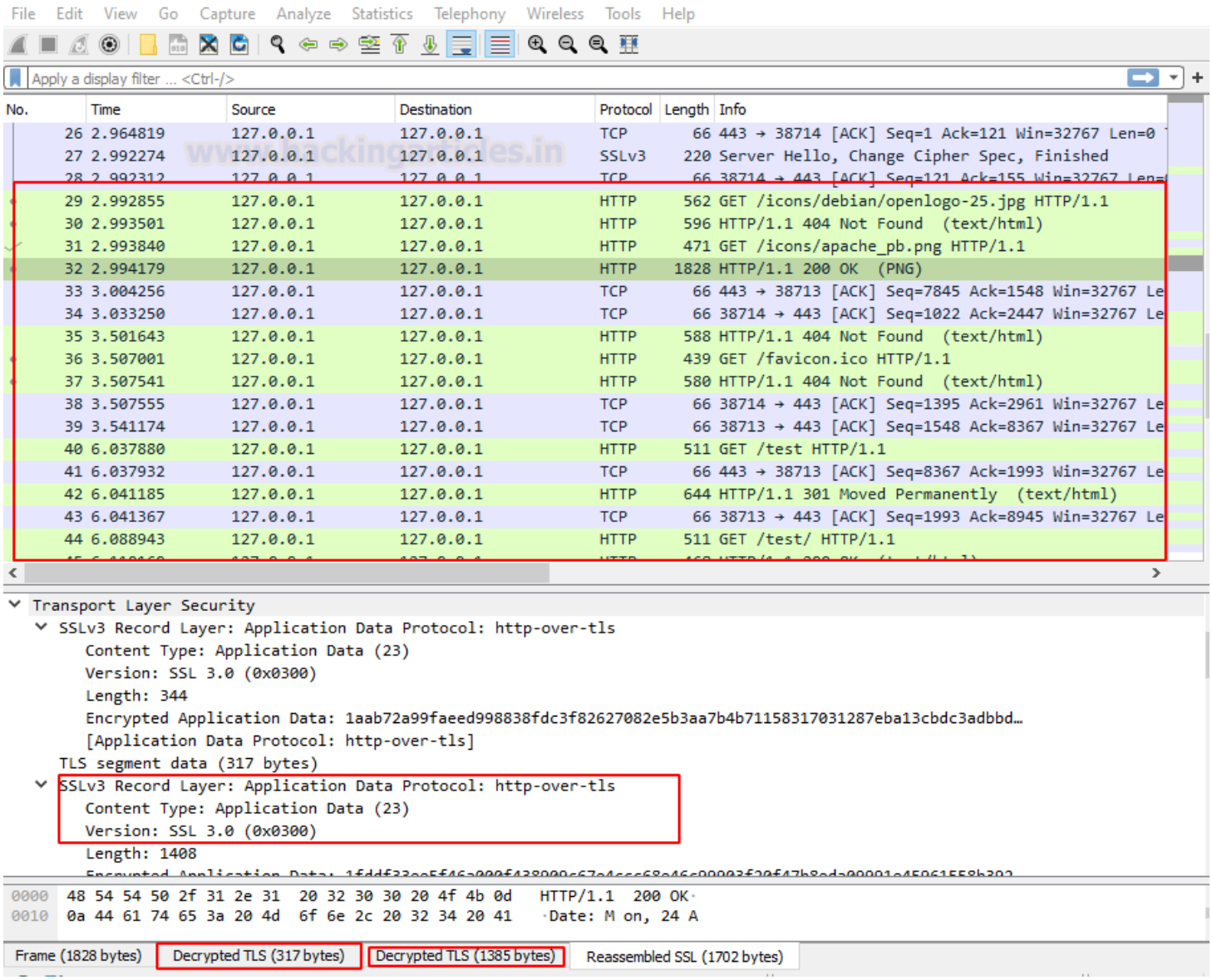
IP address: 127.0.0.1

Port: 443

Key File:



Hurray!!! As you can see, we have Successfully decrypted the Data over the TLS.



Capture Telnet Password

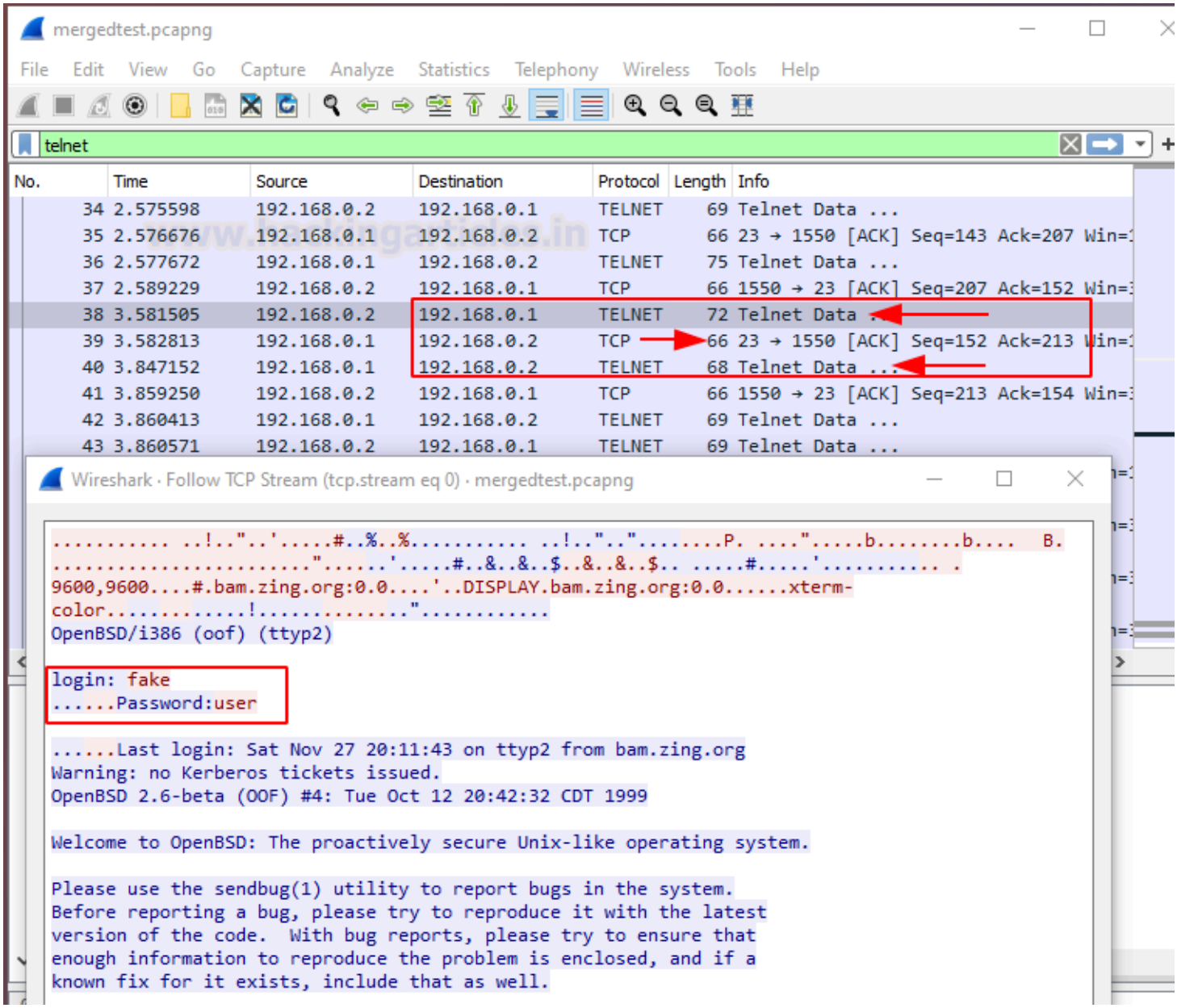
No introduction is required for Telnet protocol using port tcp/23. It is mainly used for administrative convenience and is known for its insecurity. Since encryption is not available, privacy or unauthorized access protection is not available. Telnet is still used today, however…

Telnet is a protocol used for administration on a wide range of devices. Telnet is the only option for some devices, with no other options (e.g. there is no SSH nor HTTPS web interface available). This makes it extremely difficult for organizations to completely eliminate it. Telnet is commonly seen on:

- Video Conferencing Systems
- Mainframes
- Network equipment
- Storage and Tape systems
- Imaging devices

- Legacy IP based Phones

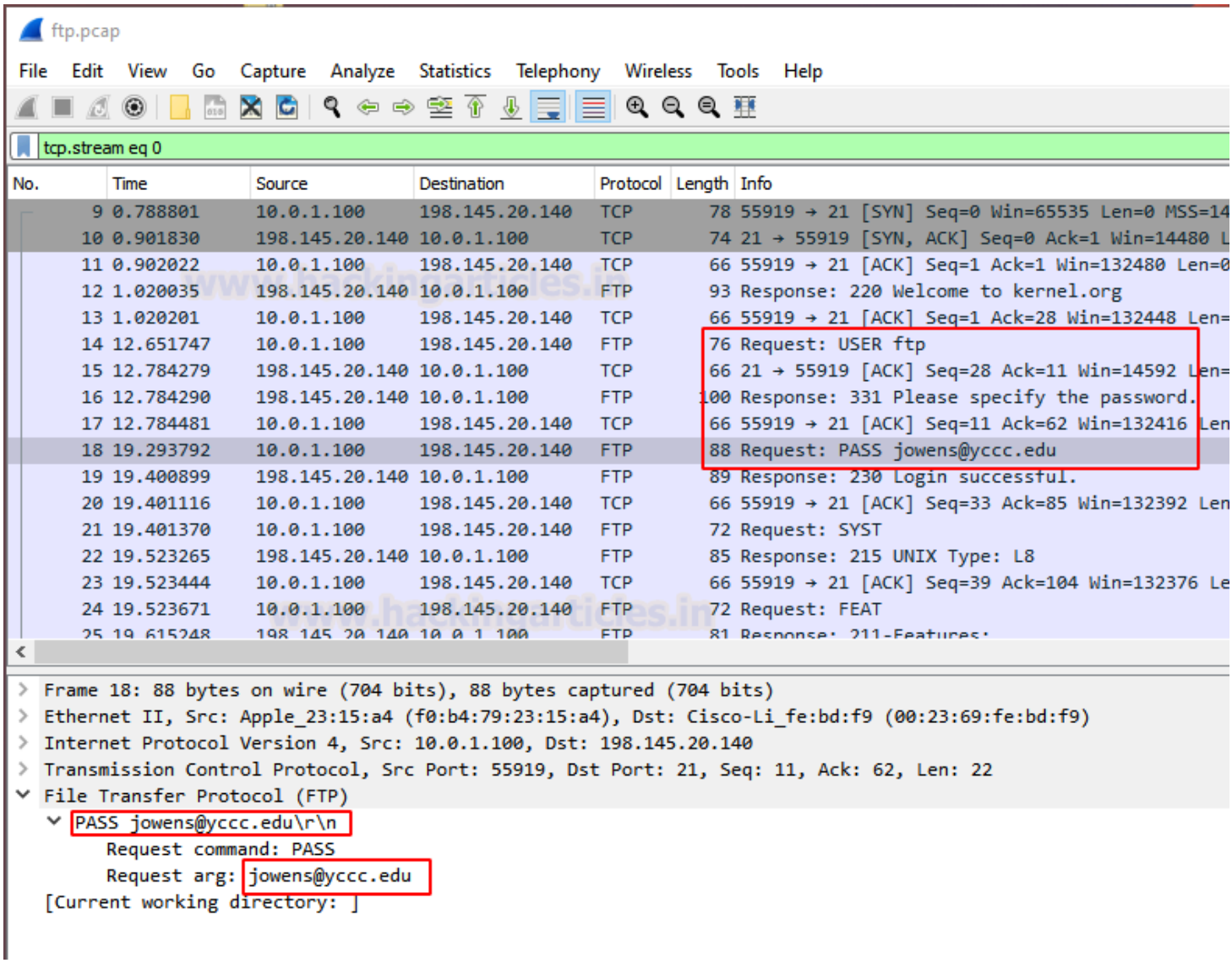
Since telnet is a plain-text protocol, an opponent can wake up to the communication and capture it all, including passwords. The following screenshot shows an example of a telnet communication with the captured password:



So, that now you can see an attacker completely overtake the Mainframe System.

Capture FTP Password

File Transfer Protocol (FTP) usually uses the TCP/20 or the TCP/21 ports. Although this protocol is very old, it is still used in their networks by some organizations. FTP is a plain text protocol so a well-positioned attacker can capture FTP login credentials with Wireshark very easily. This screenshot shows a captured FTP password with Wireshark as an example:



As you can see by sitting in a network, we can easily capture FTP credentials.

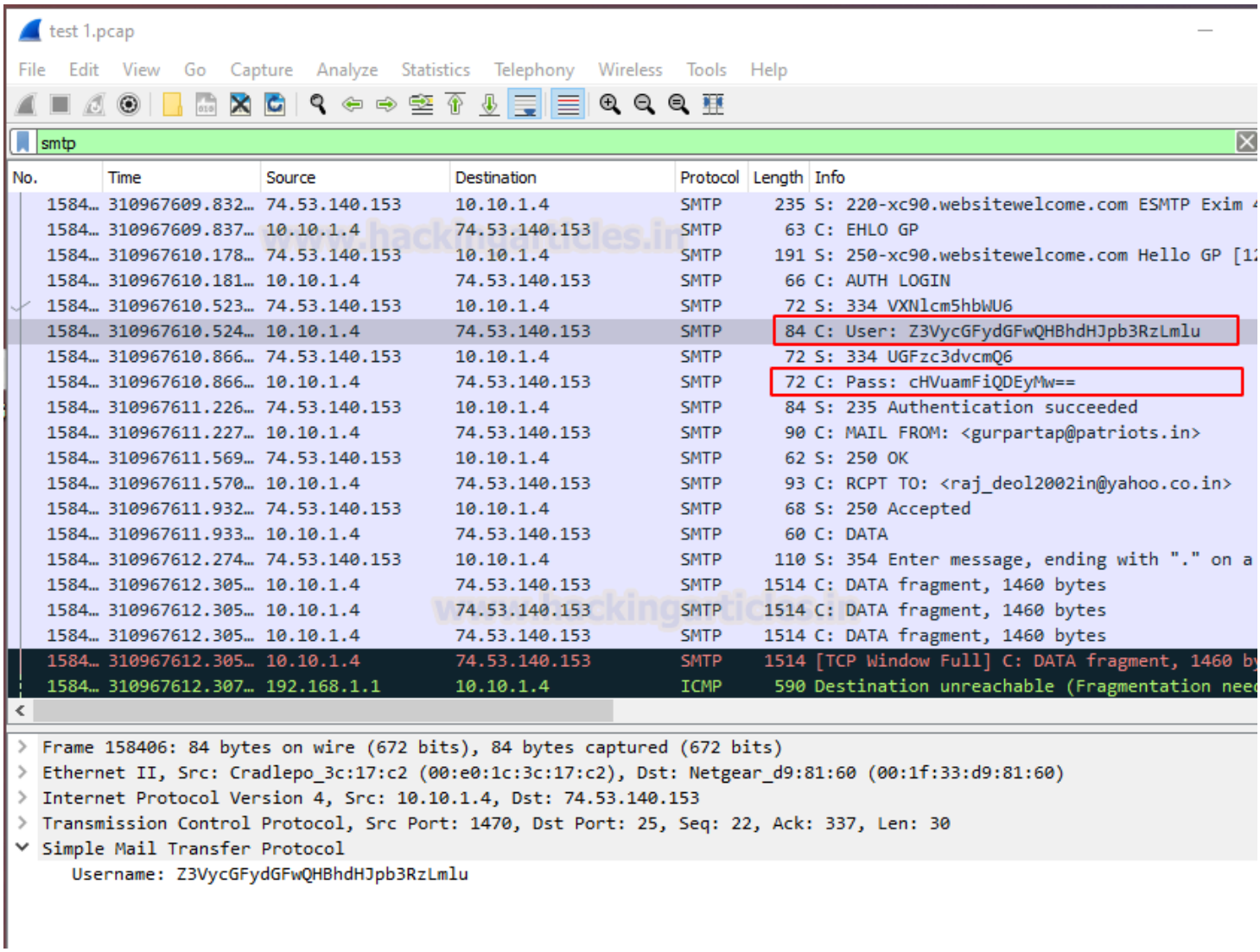
Capture SMTP password

For many decades, we have also been accompanied by SMTP (Simple Mail Transfer Protocol). It uses TCP/25 and although the port TCP/464 is secure, today the port TCP/25 is almost opened on each mail server because of reverse compatibility.

Many TCP/25 servers need the command 'STARTTLS' to begin the encryption of SSL/TLS before any attempts are made to authenticate it. However, mail servers still support plain text authentication across the unencrypted channel within certain organizations. Mostly because of heritage systems in your internal networks.

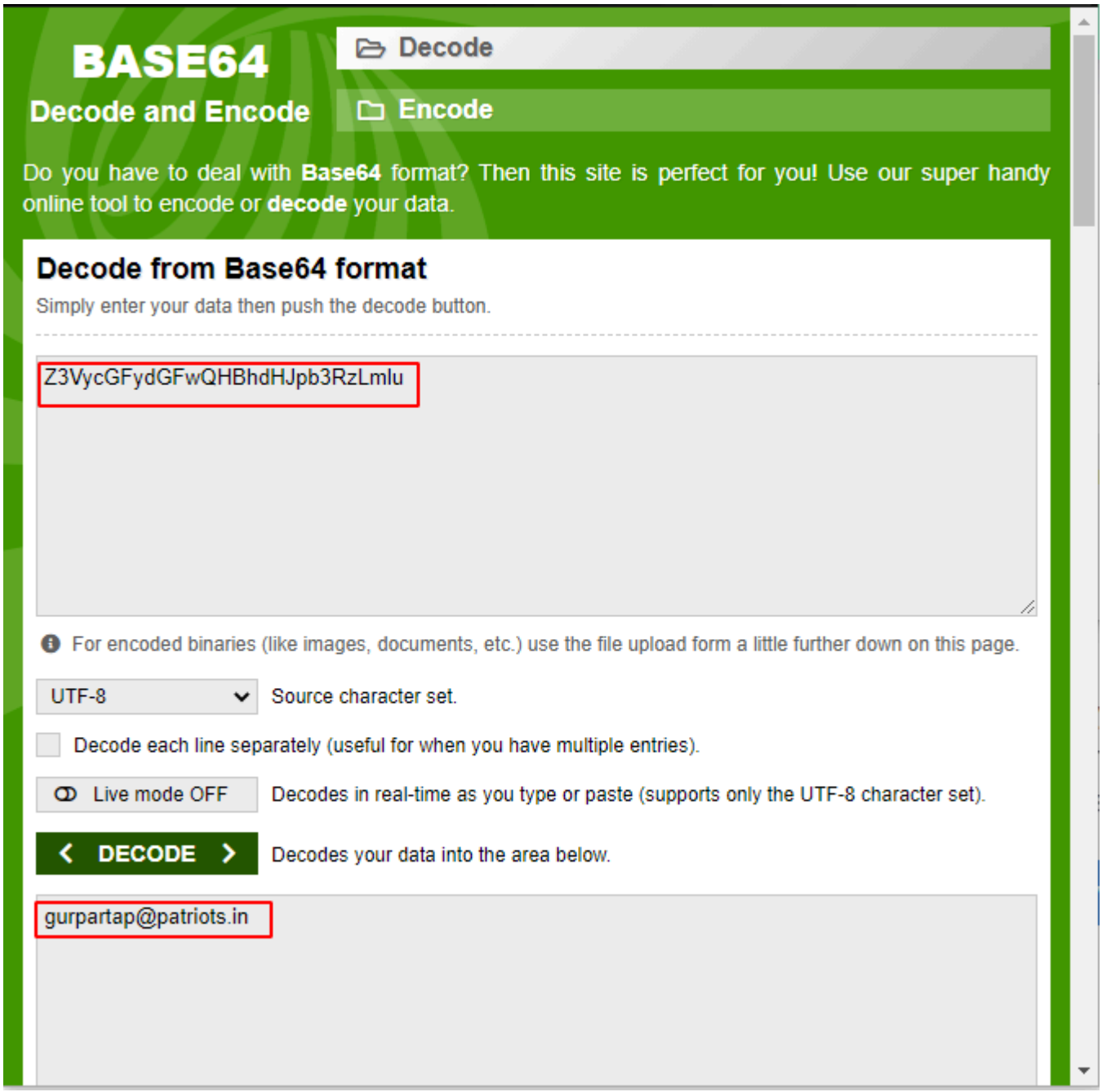
If someone is using plain text authentication during an SMTP transaction, the credentials can be sniffed from a well-positioned attacker. The attacker must only decode the username and password from base64. SMTP uses Base 64 encoding for the transaction to encode the username and password.

A captured SMTP credentials can be seen in the following screenshot with Wireshark and the consequent base64 decoder using the base64 utility.



There are many methods available to decode the base64 strings. For this, I'm using an online tool that is designed specifically for decoding such as base64decode.org or base64decode.net. But we should beware – we may not want to disclose private credentials on the Internet to other parties. In the course of penetration tests and offensive tests, sensitivity and privacy are especially crucial. This is particularly important.

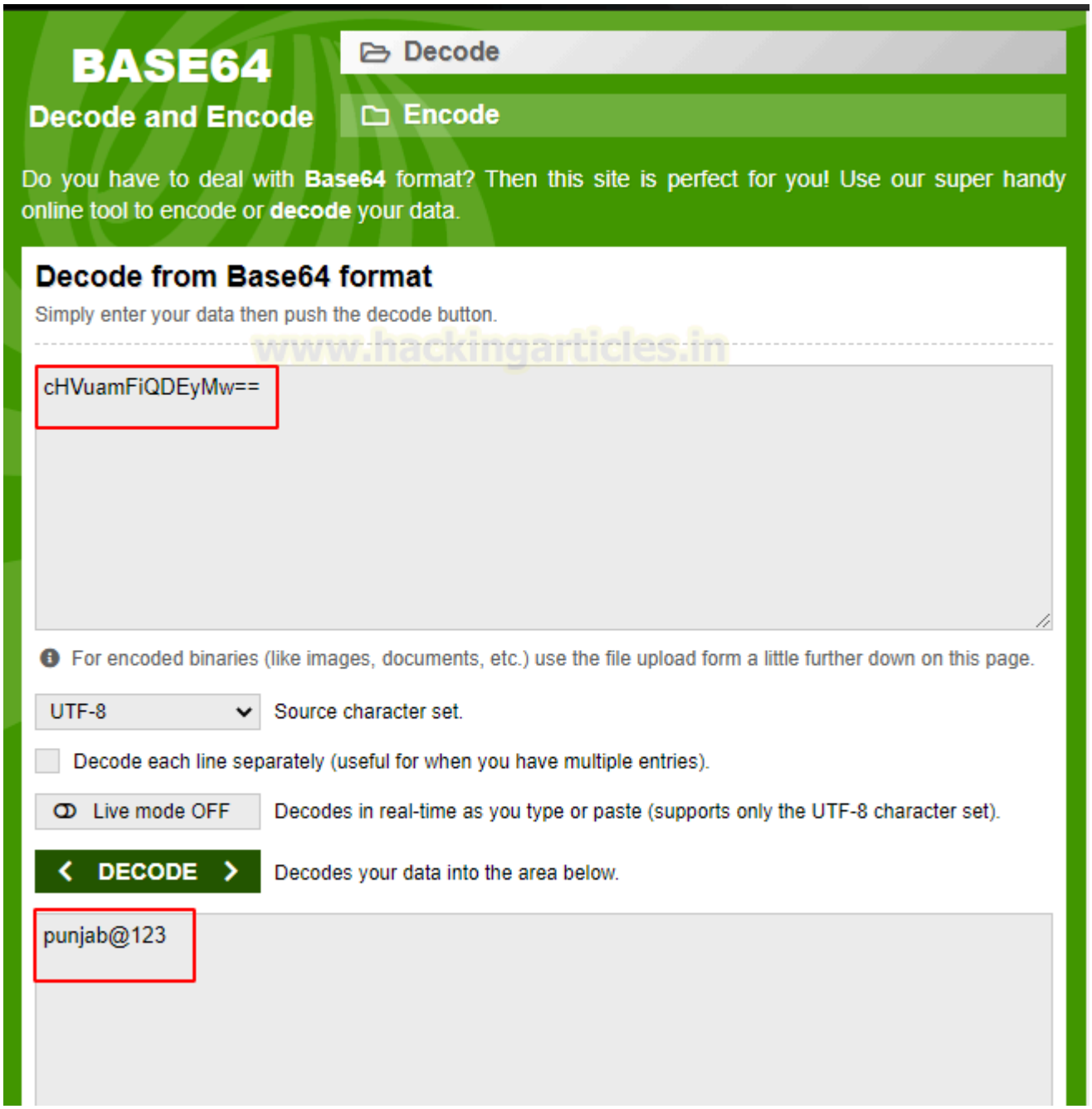
Now, just copy the value of strings of user and password and decode it via base64 decoder as shown below image. As of now, I'm decrypting the user string



User: – Z3VycGFydGFwQHBhdHJpb3RzLmlu

As you can see in the above screenshot we have successfully able to see the user name in clear text format. Similarly, we can decrypt the password

Password: – cHVuamFiQDEyMw==



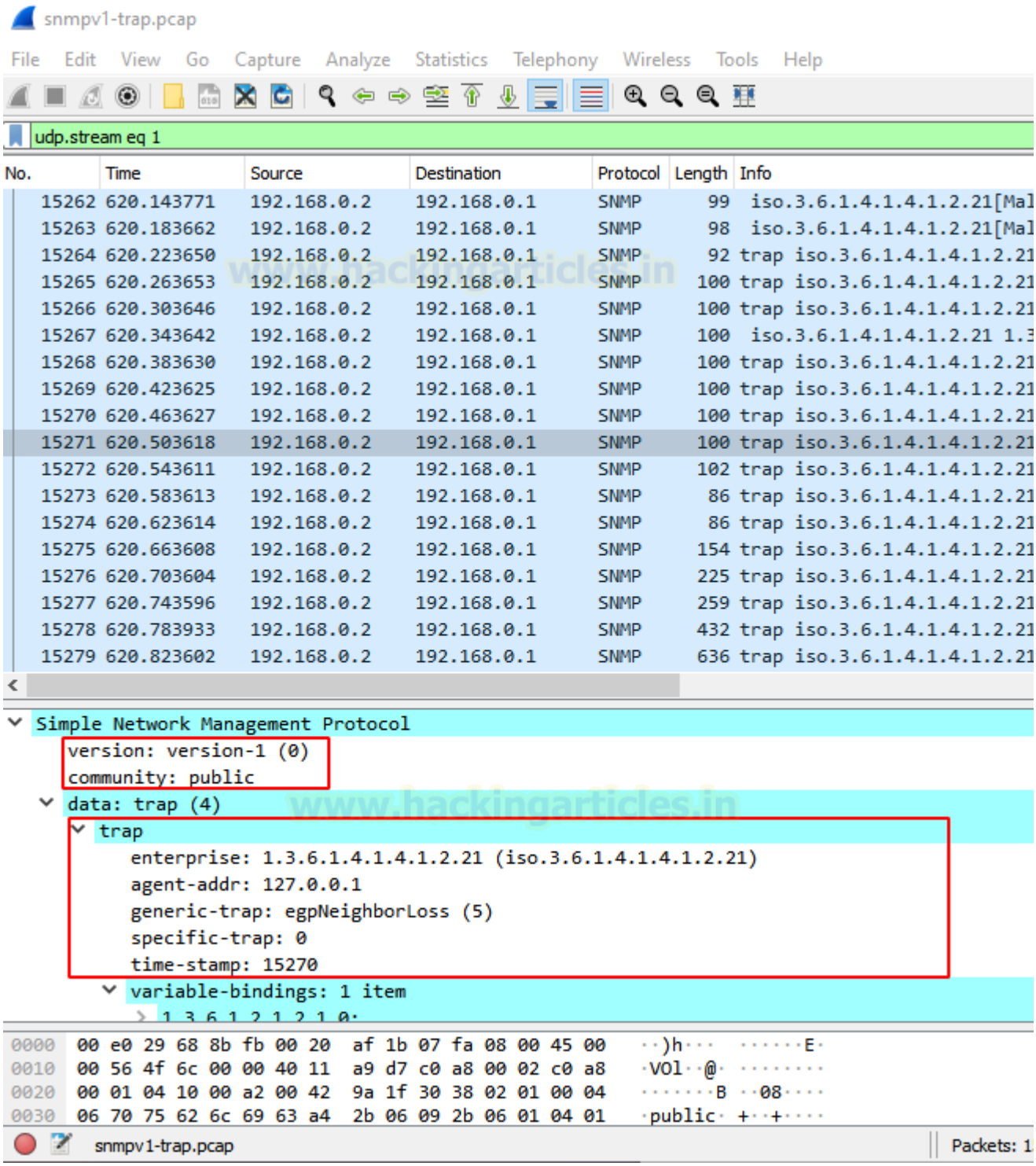
Hurray!!! Now we have got enough credentials to take over a system.

Analyzing SNMP Community String

Simple Network Management Protocol (SNMP) typically runs on port UDP/161. The main objective is network devices and their functions to manage and monitor. SNMP have 3 versions and the first 2 (v1 and v2c) versions are plain text. SNMP uses something that is equivalent to authentication, named community string. Therefore, it is almost the same to capture the SNMP community string as to capture credentials.

While SNMPv3 has been with us for nearly two decades, it takes time. In their internal networks, most organizations still use v1 or v2c. Typically this is due to the backwards compatibility in their networks with legacy systems.

An example of the SNMP community string captured using Wireshark is:



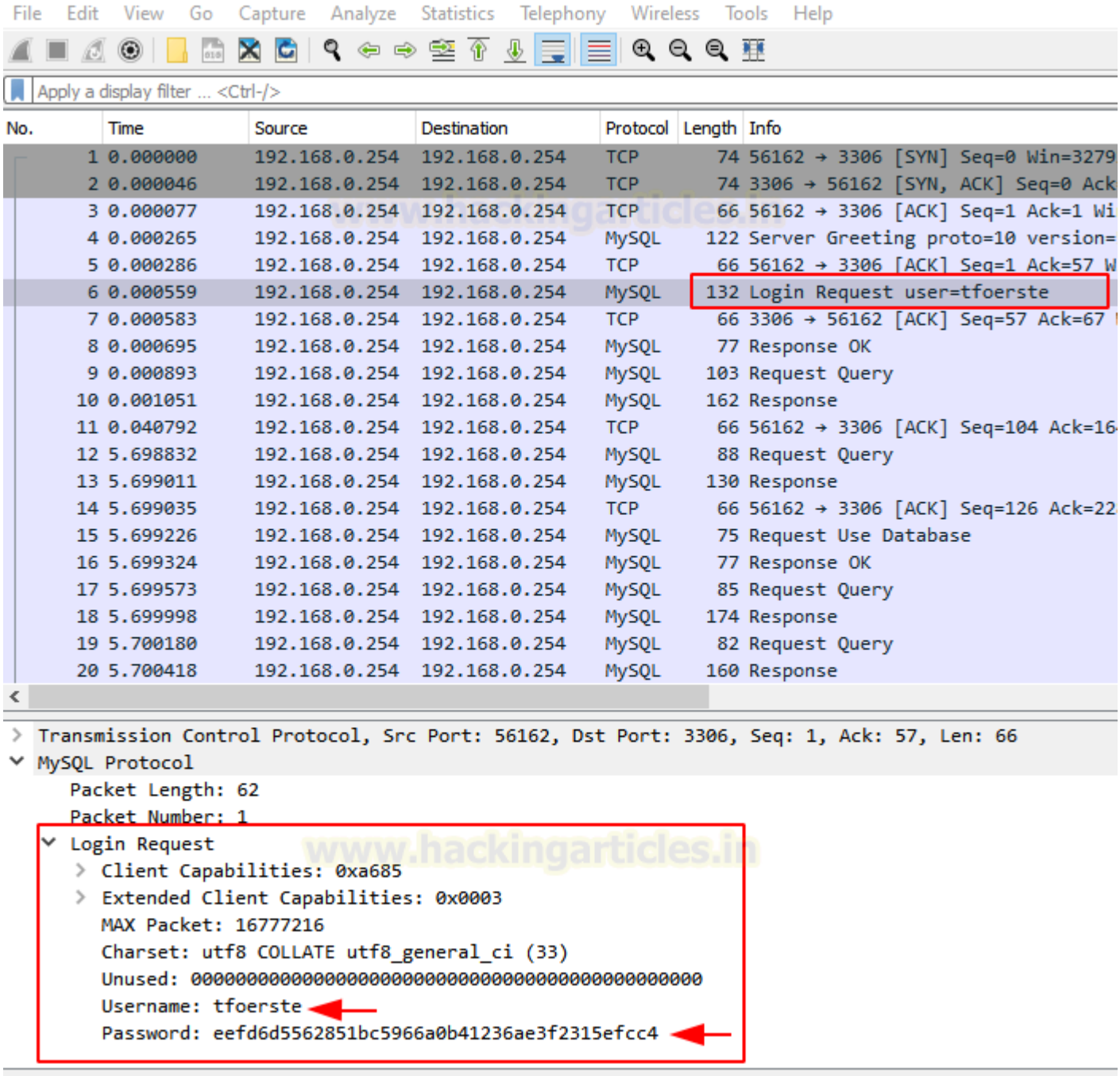
An attacker could now use the community string and collect detailed system information. This could enable the attacker to learn about the system insensitive detail and to make further attempts. Note that the community string sometimes also allows you to modify your remote system configuration (read/write access).

Capture MSSQL Password

The Microsoft SQL server usually runs on TCP/1433 port; this is yet another service we can use with Wireshark to capture the password. If the server is not configured using the ForceEncryption option, it is possible to record

plain text authentication directly or via a downgrade attack. MSSQL credentials can be easily captured by a man in the middle.

Here's an example of a Wireshark-captured MSSQL



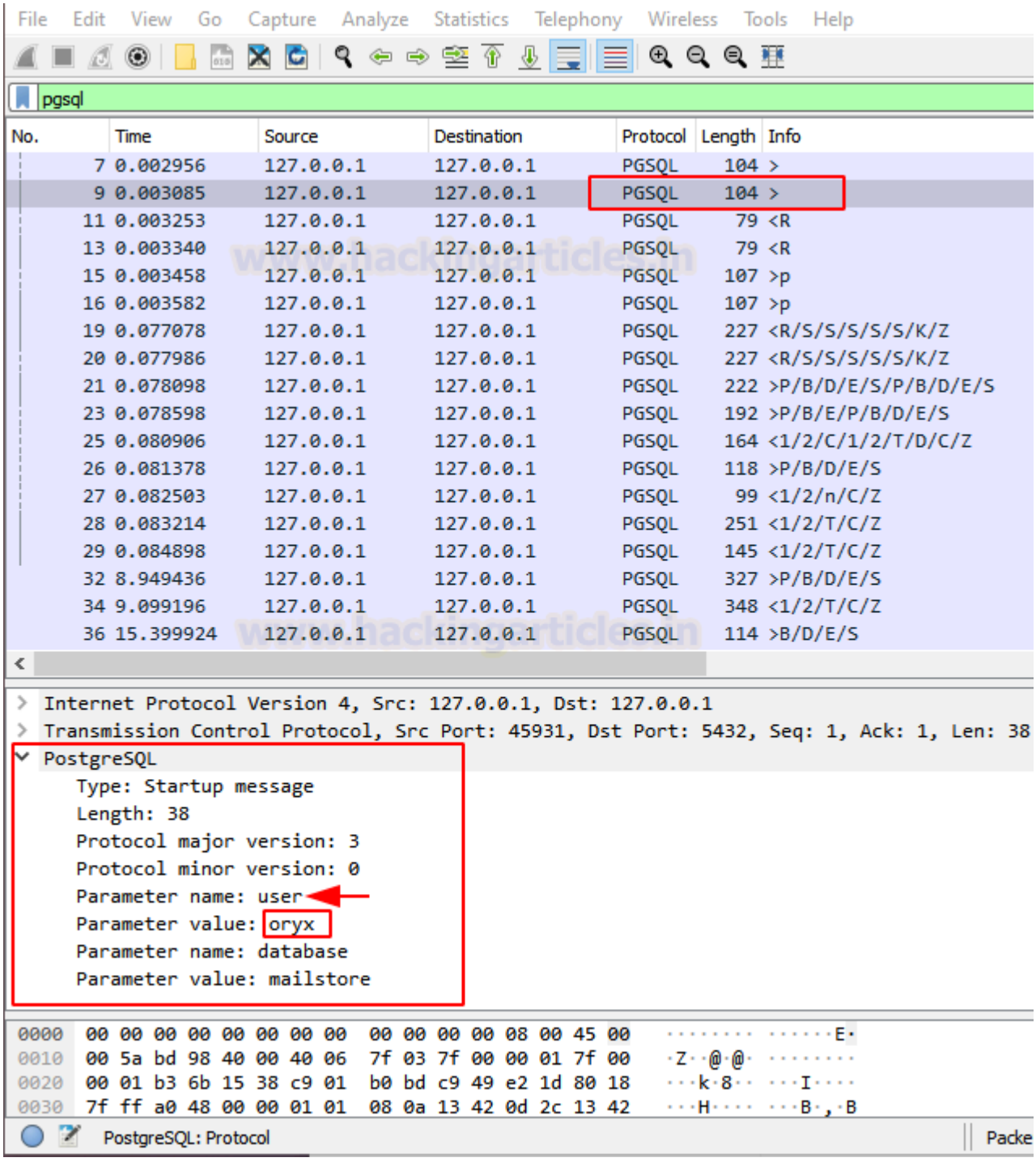
Now, we have a privileged account of the MSSQL server. Therefore, this would have a critical impact allowing the attacker to take complete control over the database server or it could also lead to remote command execution (RCE).

Capture PostgreSQL Password

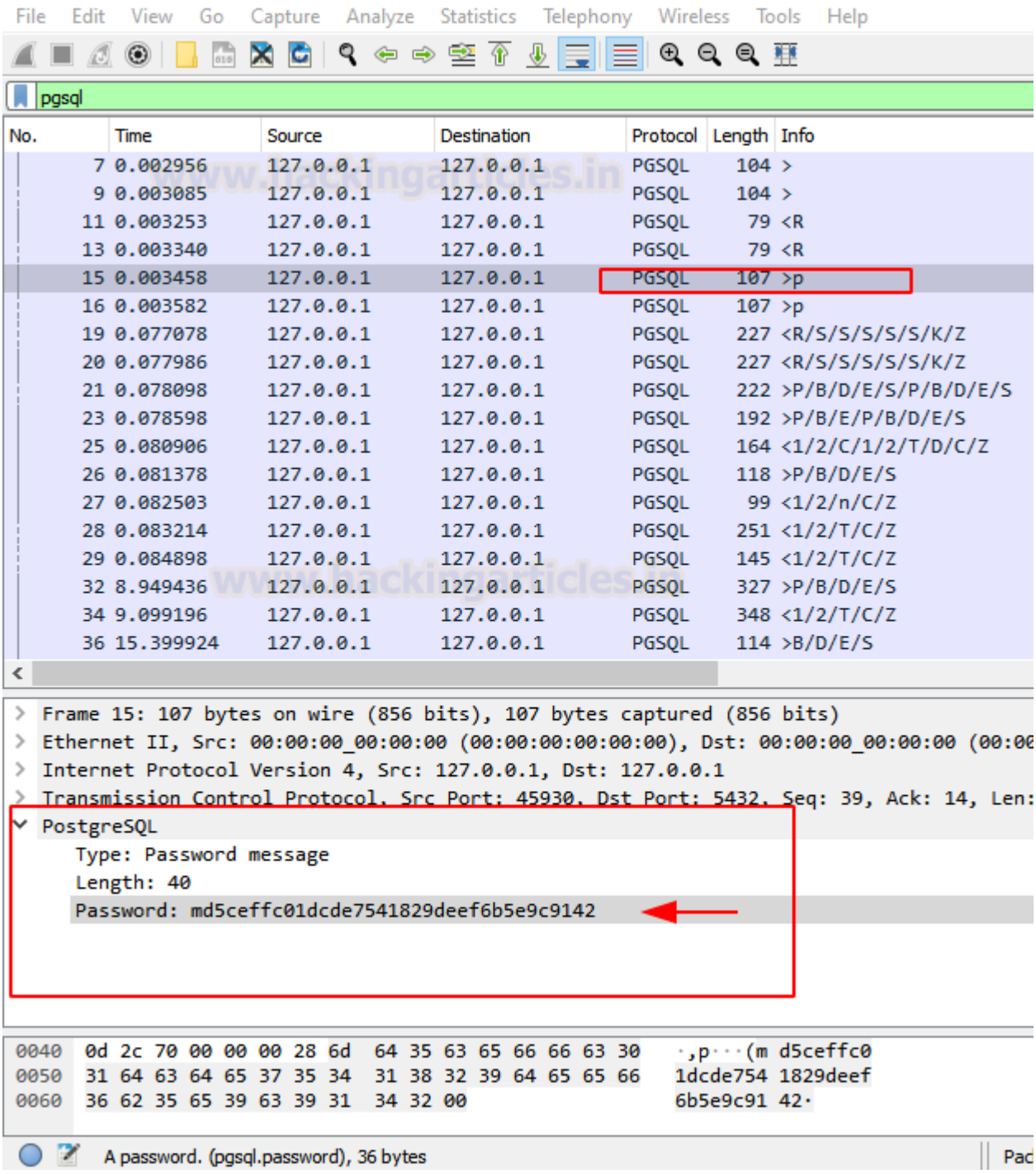
PostgreSQL is yet another widely used SQL database server. It runs on TCP port 5432 and accepts a variety of authentication methods. It is usually set to disallow clear-text authentication, but it can also be set to allow it. In

such cases, a well-positioned attacker could intercept network traffic and obtain the username and password.

It should be noted that PostgreSQL authentication occurs in multiple packets. The username and database name comes first:

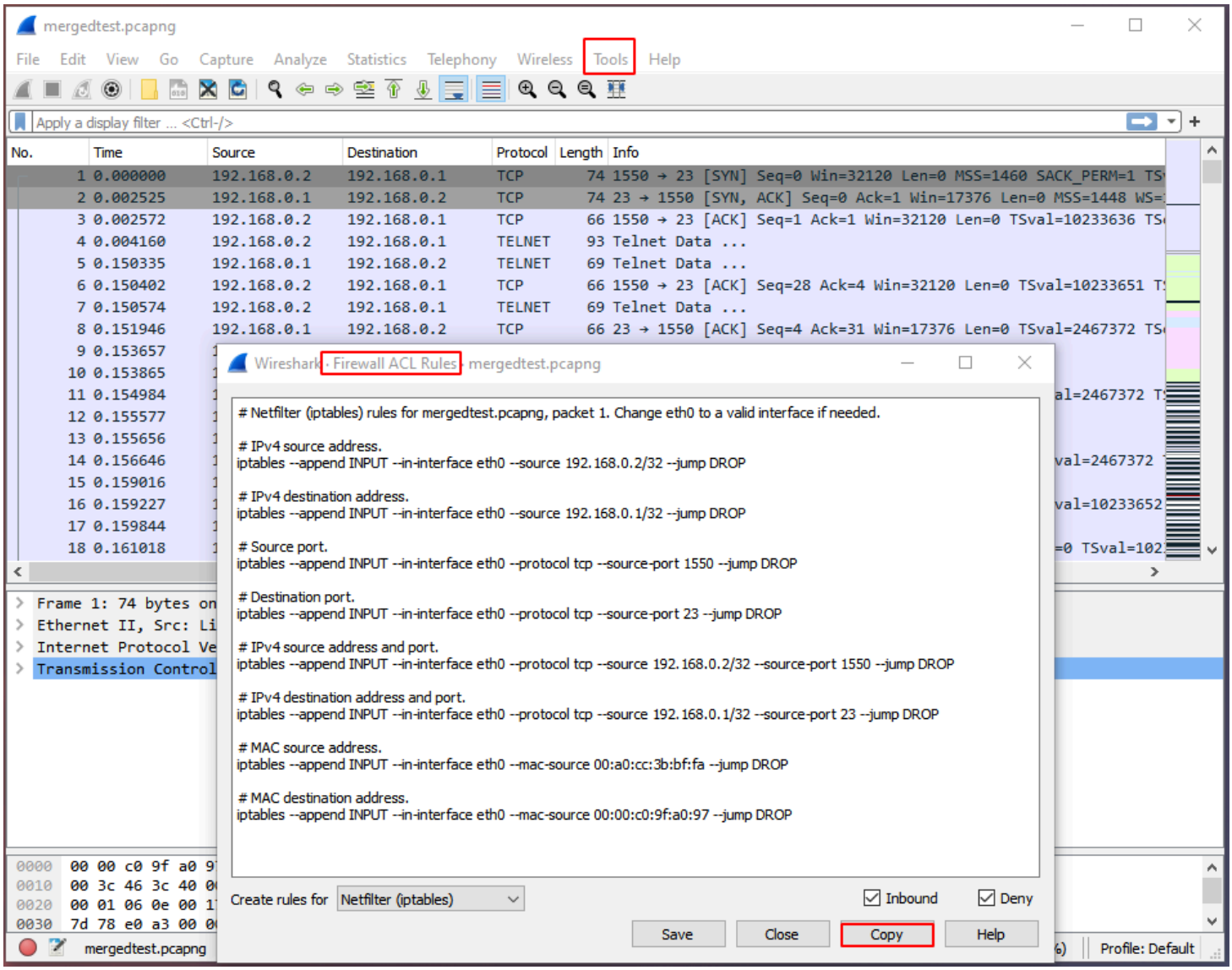


We can also see the PostgreSQL password in the following network packet:



Creating Firewall Rules with Wireshark

Although Wireshark cannot block network traffic, it can assist us in the development of firewall rules for our firewall. Wireshark will create firewall rules based on the traffic we're looking at. To block a packet, all we have to do is pick it and navigate through the menu:



Selected rules can now be copied and pasted directly into our firewall. The following firewalls' syntax is supported by Wireshark:

- Windows Firewall(netsh)
- IP Filter(ipfw)
- NetFilter (iptables)
- Packet Filter(pf)

Conclusion

Wireshark can catch authentication for a wide range of network protocols. There is a possibility as long as we have the ability to eavesdrop on network traffic and the communication is not encrypted. Passwords aren't the only thing that a well-placed attacker can capture; virtually any type of data passing through the network can be captured.