6A

```
# Copyright 2001-2020 Sourcefire, Inc. All Rights Reserved.
#
# This file contains (i) proprietary rules that were created, tested and certified by
# Sourcefire, Inc. (the "VRT Certified Rules") that are distributed under the VRT
# Certified Rules License Agreement (v 2.0), and (ii) rules that were created by
# Sourcefire and other third parties (the "GPL Rules") that are distributed under the
# GNU General Public License (GPL), v2.
#
# The VRT Certified Rules are owned by Sourcefire, Inc. The GPL Rules were created
# by Sourcefire and other third parties. The GPL Rules created by Sourcefire are
# owned by Sourcefire, Inc., and the GPL Rules not created by Sourcefire are owned by
# their respective creators. Please see http://www.snort.org/snort/snort-team/ for a
# list of third party owners and their respective copyrights.
#
# In order to determine what rules are VRT Certified Rules or GPL Rules, please refer
# to the VRT Certified Rules License Agreement (v2.0).
#
#------------
# LOCAL RULES
#------------
ipvar GOOGLE 8.8.8.8
alert tcp any any -> $GOOGLE any (msg:"Test google.com";sid:1000005;)
```

```
09/23-09:52:38.158828  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.158828  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.164626  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.164763  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.170996  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.170996  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.176658  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.176658  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.182709  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.182709  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.188428  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.188428  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.194684  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.194684  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.200668  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.200668  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.206579  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.206579  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.212257  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.212257  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.218165  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.218165  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.224575  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.224575  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.230216  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.230216  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.236836  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.236836  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.242617  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.242617  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.247852  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.247852  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.253698  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.253698  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.259752  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.259752  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.265543  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.265543  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.271458  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.271661  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.180.4:443 -> 192.168.72.221:23606
09/23-09:52:38.276179  [**] [1:1000005:0] Test google.com [**] [Priority: 0] {TCP} 142.250.193.228:443 -> 192.168.72.221:23732
```

Date :

E6 → Reacting In Real Time [6A Task]

Page No. : 17
Expt. No. : 6a

Objective

To customize the real time rules and getting the alerts

Procedure

i) Open Snort Folder, navigate to rules and open local.rules file

ii) Add following alert rule :-

```
ipvar GOOGLE 8.8.8.8
alert tcp any any -> $GOOGLE any (msg:"Testing Google";sid:101;)
```

iii) Save it, open cmd as admin

iv) Navigate to bin folder of Snort using 'cd' command

v) Execute the following :-

```
Snort -i <Interface Index> -c<Config file path> -A console
```

vi) Open browser and browse google

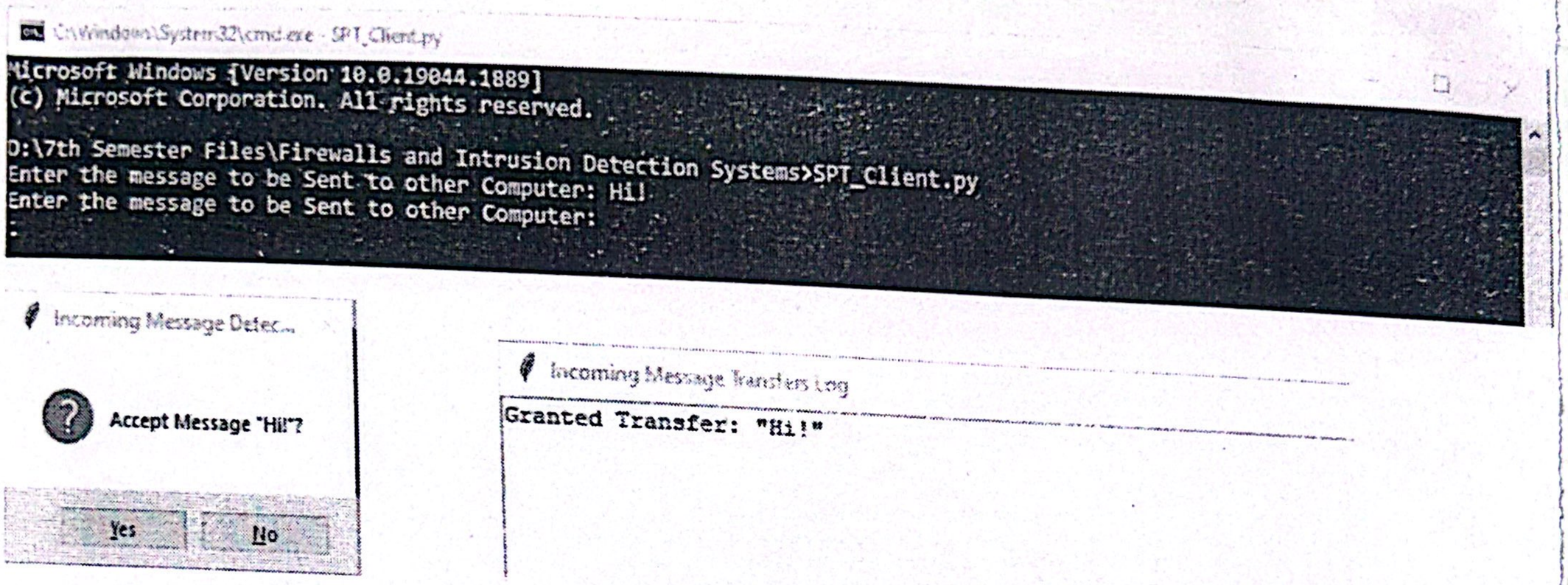
vii) All alerts are Shown in real time on the CLI

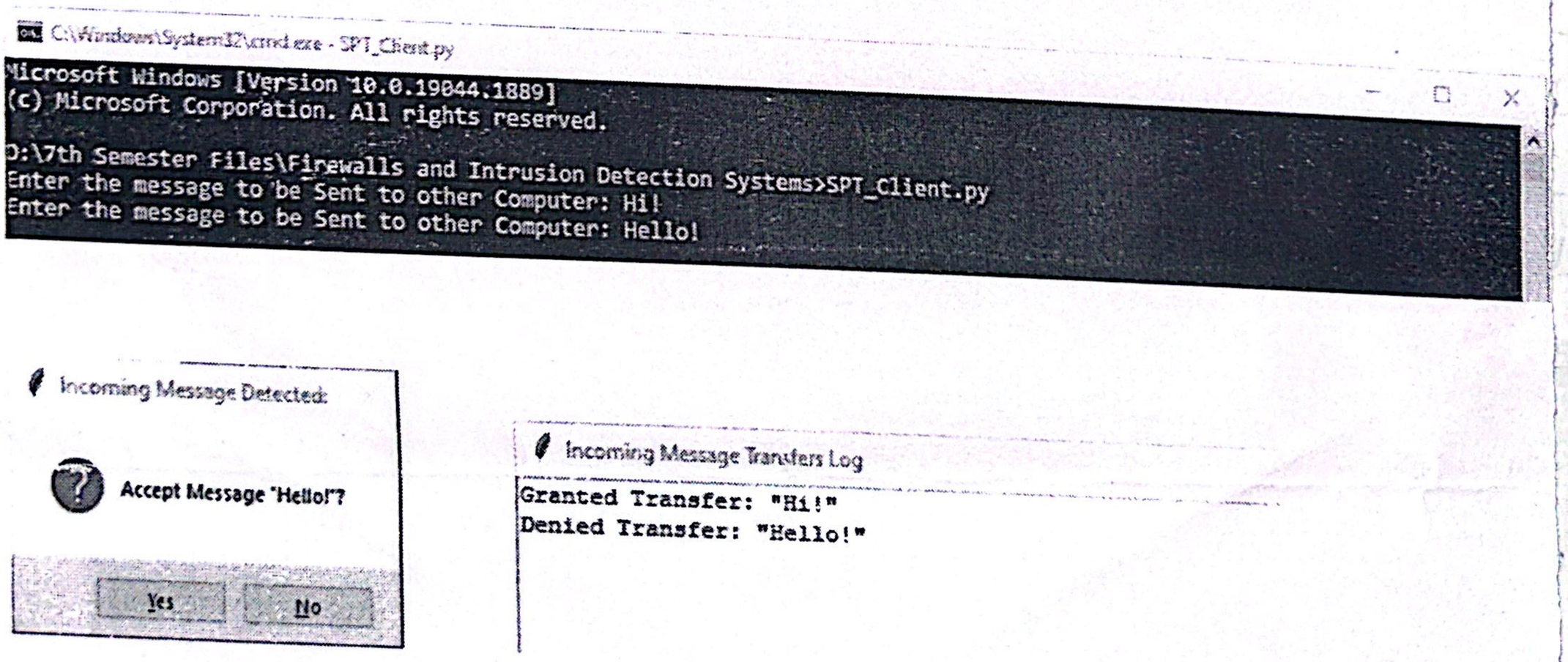
Result

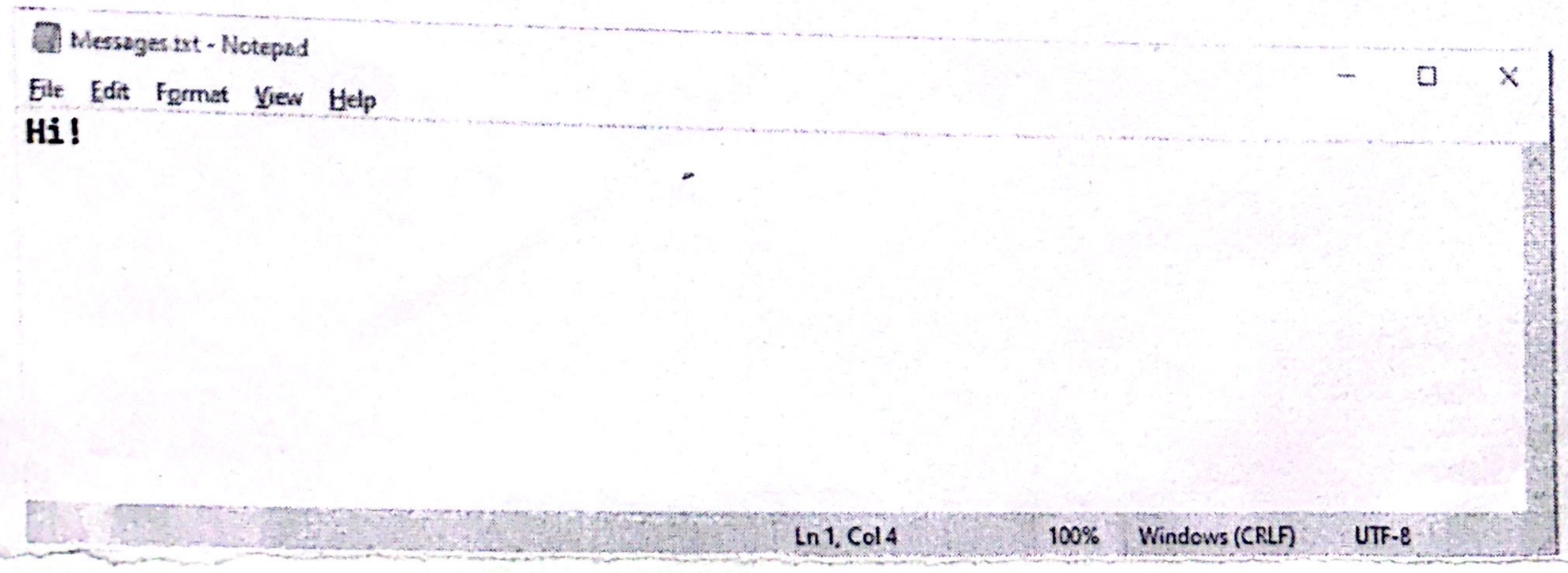
Successfully logged "realtime" threats by writing appropriate rules and running them on snort IDS tools

Teacher's Signature:

6B


C:\Windows\System32\cmd.exe - SPT_Client.py
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.
D:\7th Semester Files\Firewalls and Intrusion Detection Systems>SPT_Client.py
Enter the message to be Sent to other Computer: Hi!
Enter the message to be Sent to other Computer:
Incoming Message Detec...
Accept Message "Hi!"?
Yes
No
Incoming Message Transfers Log
Granted Transfer: "Hi!"


C:\Windows\System32\cmd.exe - SPT_Client.py
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.
D:\7th Semester Files\Firewalls and Intrusion Detection Systems>SPT_Client.py
Enter the message to be Sent to other Computer: Hi!
Enter the message to be Sent to other Computer: Hello!
Incoming Message Detected:
Accept Message "Hello!"?
Yes
No
Incoming Message Transfers Log
Granted Transfer: "Hi!"
Denied Transfer: "Hello!"


Messages.txt - Notepad
File Edit Format View Help
Hi!
Ln 1, Col 4
100%
Windows (CRLF)
UTF-8

Date :

E6B Ensuring Safety of Packet Transfer Process (Subtask)

Page No. : 18

Expt. No. : 6b

Aim

To Create an automated Client Server program in python that demonstrates Secure packet transfer

Procedure & Codes

1) Use a suitable Python Compiler for running the codes & debugging

2) below is the codes

Server.py :-

```
import socket
import thinker as tk
from thinker import messagebox
import threading
import sys

Server_Socket = Socket.Socket(Socket.AF_INET, Socket.SOCK_STREAM)
Server_Socket.bind (("0.0.0.0", 12345))
Server_Socket.listen(1)
Obtained_Messages = []

root = tk.Tk()
root.title ("Inc message Transfer by")
text-area = tk.Text (root)
text-area.pack()

def update_text_area(text):
```

Teacher's Signature:

```
text_area.insert(tk.END, text)
text_area.see(tk.END)

def handle_message(client_socket):
    while True:
        data = client_socket.recv(1024).decode()
        if not data:
            break

        response = messagebox.askyesno("Incoming msg detected:",
        f"Accept message \"{data}\"?")
        if response:
            update_text_area(f"Granted Transfer: \"{data}\"\n")
            obtained_Messages.append(data)
        else
            update_text_area(f"Denied Transfer: \"{data}\"\n")
            print("Msg granted: ", end: " ")
            print(obtained_Messages)

Recieved
    with open("Messages.txt", "w") as f:
        for Msg in obtained_Messages:
            f.write(f"{Msg}\n")

print("Waiting for incoming connection...")

def accept_connections():
    while True:
        try:
```

Teacher's Signature:

Date :

Page No. : 20
Expt. No. : 6b

```
client_socket, client_address = server_socket.accept()
print(f"Accepted connection from {client_address}")

client_thread = threading.Thread(target=handle_message,
args=(client_socket,))
client_thread.start()
except KeyboardInterrupt:
    print("Server interrupted. Closing server socket.")
    server_socket.close()
    sys.exit()

accept_thread=threading.Thread(target=accept_connections)
accept_thread.start()

try:
    root.mainloop()
except KeyboardInterrupt:
    print("Server Interrupted. Exiting.")
    sys.exit()
```

client.py

```
import socket
import threading
client_socket=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(("192.168.72.250", 12345))

def send_message():
```

Teacher's Signature:

Date :

Page No. : 21
Expt. No. : 6b

```
while True:
    message = input("Enter msg to be sent to other computer : ")
client_Socket.send(message.encode())

Send_thread = threading.Thread(target = send_message)
Send_thread.start()
```

3) Save & debug codes. Save server file as No Console Py exec file.
4) Open task scheduler, under actions, create a task and name it.
5) Under actions, in new dialog box, click on new and add the path in which your python exec file resides.
6) In Argument (optional) field, give name of server file with extension
7) In start in (optional) field enter path of server file.
8) In the "triggers" category, select time and frequency during which code can start automatically to execute.
9) Once finalized, click ok
automation is done successfully, boot on time will execute server py file

Result

Thus we have implemented Packet transfer in python and accepted/recieved msgs sent from Client accordingly thru a dialog box in both MANUAL as well as AUTOMATED Way.

Teacher's Signature: