MULTI THREADED PROXY SERVER CLIENT

Team Members:
Muskaan Goel
Lysetti Lakshmi Poojitha
Meghana Shree M
Muskaan Sinha

Introduction:

Multithreading is the ability of a central processing unit to provide multiple threads of execution concurrently, supported by the operating system.

A multithreaded server is any server that has more than one thread. As a transport requires its own thread, multithreaded servers also have multiple transports. The number of thread-transport pairs that a server contains defines the number of requests that the server can handle in parallel.

A proxy server is a computer system or router that functions as a relay between client and server. The word proxy means "to act on behalf of another," and a proxy server acts on behalf of the user. All requests to the Internet go to the proxy server first, which evaluates the request and forwards it to the Internet.

If a server uses a number of sub-processes in order to serve multiple requests by allocating each socket to one sub-process then it is known as a multi-threaded server. This is how the server handles multiple connections at a time.

Basic Terminology:

- MULTITHREADING:-. Allows access to two or more threads. Execution occurs in more than one thread of control, using parallel or concurrent processing. User-level or Application-level threads.
- OPERATING SYSTEM:- is software that communicates with the hardware and allows other programs to run. It is comprised of system software, or the fundamental files your computer needs to boot up and function.
- PROXY SERVER:- A proxy server is a computer system or router that functions as a relay between client and server. It helps prevent an attacker from invading a private network and is one of several tools used to build a firewall.
- CLIENT:- A client is a computer or a program that, as part of its operation, relies on sending a request to another program or a computer hardware or software that accesses a service made available by a server (which may or may not be located on another computer).

PROXY:-A proxy is an application that “breaks” the connection between client and server. ... Proxy servers are available for common Internet services; for example, a Hypertext Transfer Protocol (HTTP) proxy used for Web access and a Simple Mail Transfer Protocol (SMTP) proxy used for e-mail.

ROUTER:-A router is a device that forwards data packets along networks. A router is connected to at least two networks, commonly two LANs or WANs or a LAN and its ISP's network. Routers are located at gateways, the places where two or more networks connect.

SERVER:-A terminology server is a piece of software providing a range of terminology-related software services through an applications programming interface to its client applications.

About the Project:

In this project we are supposed to build a multi-threaded Web proxy server that is capable of delivering Web content on behalf of a remote Web server. When a user's browser is configured to connect via a proxy *server*, their browser establishes a connection to the proxy server's machine and forwards its complete request to the proxy server rather than the "real" Web server. The proxy server accepts user's HTTP requests and forwards them to their final destination - essentially it introduces an extra "hop" between the user's browser (or the client) and the Web server.

There are several reasons why people want to use a proxy server instead of connecting to the remote Web server directly:

1) users want to anonymize themselves from the Web server;

2) a corporation might want to monitor or restrict employees' Web surfing;

3) a proxy can be used to locally cache data in order to reduce the amount of global traffic.

Multi-threading is not crucial to the functionality of a Web proxy server, but it is important to allow the server to process multiple simultaneous requests in parallel, a must-have feature for a practical server.

The server will be able to handle multiple simultaneous service requests in parallel. This means that the Web server is multi-threaded. In the main thread, the server listens to a fixed port. When it receives a TCP connection request, it sets up a TCP connection through another port and services the request in a separate thread.

<u>CODE:</u>

The program needs two classes

1.Proxy Server

2.HTTP request

<u>Language used:</u> Python

Advantages Of A Proxy Server

1. Makes You Anonymous Online

Many people use proxy servers to help mask their public addresses. This way, anyone trying to capture data from your computer will find it harder to track you.

2. Proxy Servers Provide An Extra Layer Of Protection

Proxy servers can be configured to block access to sites and web pages known to contain malicious code in the form of malware, phishing links, and viruses. The proxy servers can also be used to restrict access only to websites allowed by an institution, workplace, or an organization. This, therefore, prevents unauthorised access to pages or sites that could land a user in trouble online.

3. Faster Page Loading Speeds

A proxy server also works by caching data from the websites you access. Should you wish to come back to the site, the server will return the request faster than it would if it had to connect to the internet again. This means faster loading speeds for such a website.

Disadvantages Of A Proxy Server

1. **The proxy provider might keep track of your online activity**

   Although a proxy server might come in handy in masking your public IP address, some providers might still be able to see and monitor your activity online. For this reason, it would be advisable to research on a proxy provider before using their service.

2. **Your connection might not be encrypted**

   Although most proxy servers will provide you with some level of anonymity, many of these do not encrypt your connection. Most providers only use an SSL certificate to encrypt data passing through their servers. This isn't however enough considering attackers today can use SSL stripping to decrypt such connections. Although the page might show SSL encryption on your site, your data might not be as secure when going through the proxy server. Consider investing in a proxy server that encrypts all connections.

About the Code:

We are importing the packages such as : sys,socket,_thread,traceback and ssl.

We have three functions used in our code :

def main() : In the main function we are entering the listening port, which is 4444 for us. Then the socket is initialised and binded successfully. The multithreaded proxy server is started successfully.Then the command is given for shutting it down.

def conn_string(): This variable conn_string is initialised to conn,data and addr.Then utf-8 is used which is a variable-width character encoding used for electronic communication.

def proxy_server():In this function the server starts working and the websites are accessed using the multithreaded proxy server.The request is sent to the server in 127.0.0.1 format and then the code starts running.

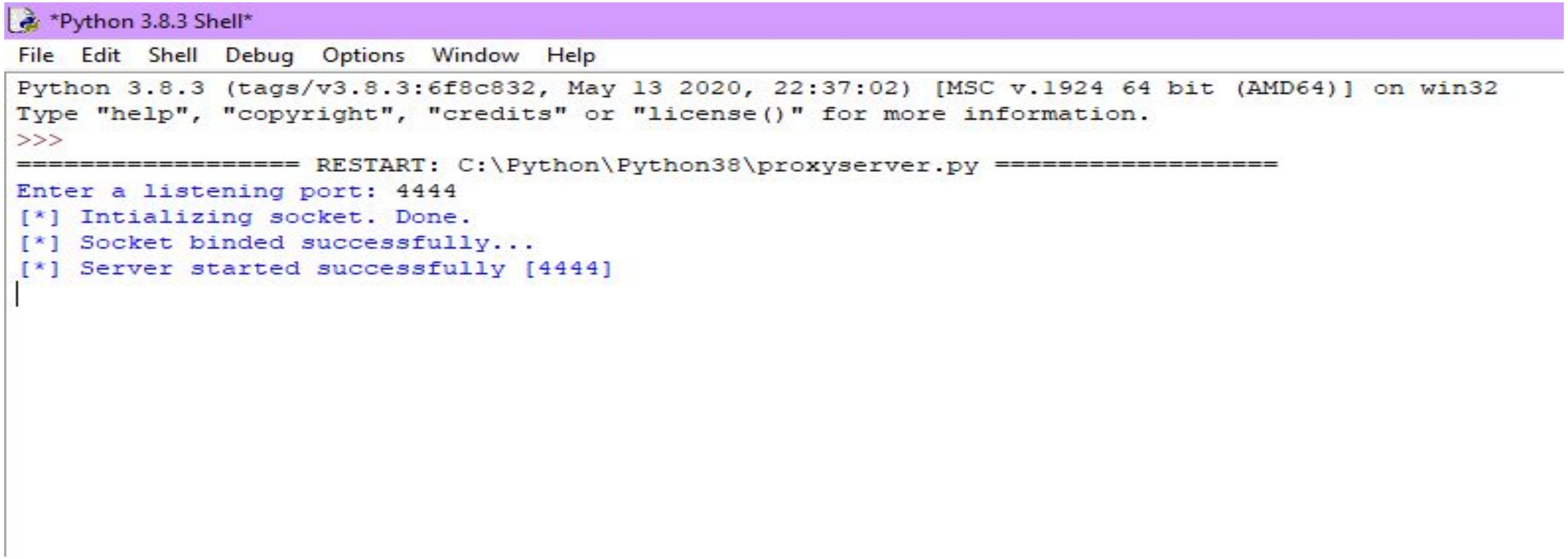
Code Snippet:

```
def main():

   global listen_port, buffer_size, max_conn

   try:

      listen_port = int(input("Enter a listening port: 
"))

   except KeyboardInterrupt:

      sys.exit (0)

 max_conn = 10000

   buffer_size = 10000

   try:

 s=socket.socket(socket.AF_INET, 
socket.SOCK_STREAM)

      s.setsockopt(socket.SOL_SOCKET, 
socket.SO_REUSEADDR, 1)

      s.bind(("", listen_port))

      s.listen(max_conn)

      print("[*] Intializing socket. Done.")

      print("[*] Socket binded successfully...")

      print("[*] Server started successfully 
[{}]".format(listen_port))

   except Exception as e:

      Print

sys.exit(2)
```

OUTPUT:

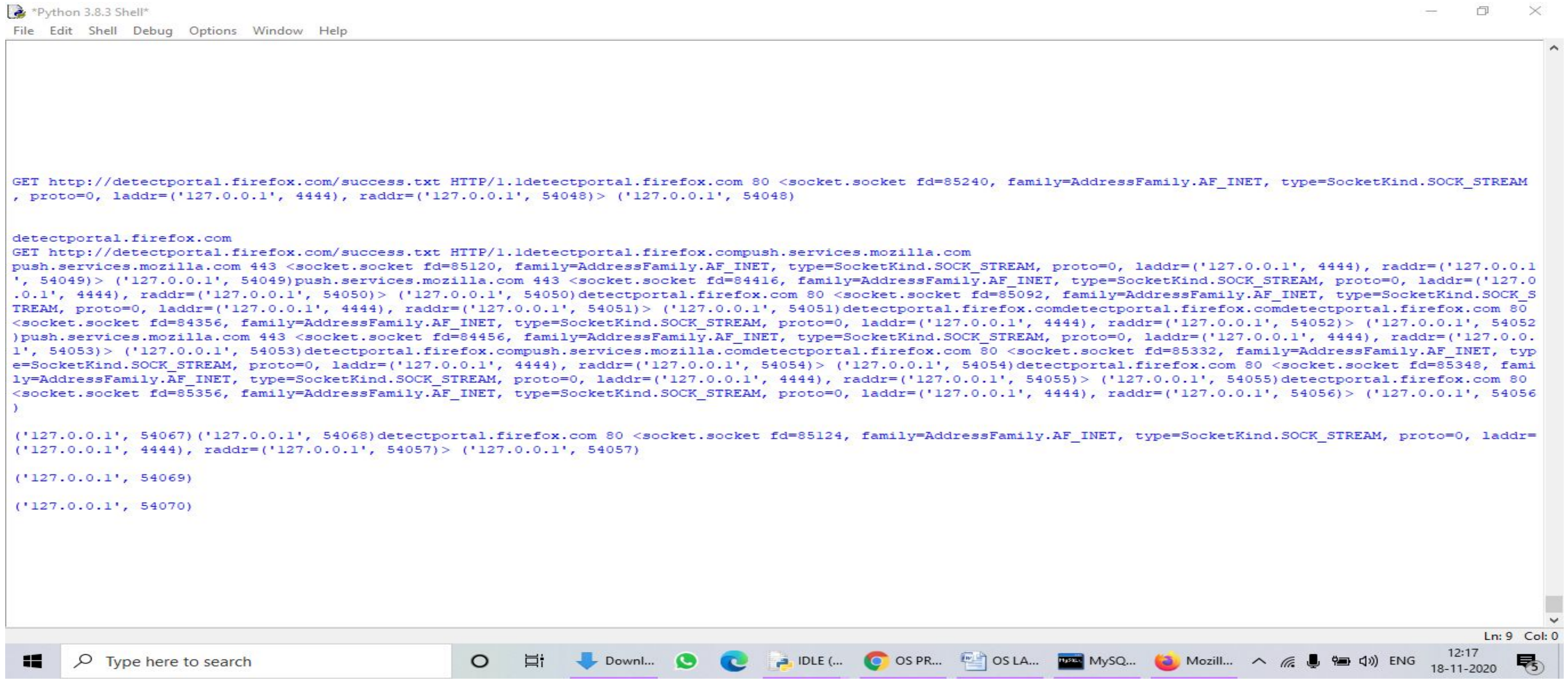
The first part of the output where we enter the listening port and the socket is initialised and binded.

The server starts processing.



```
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: C:\Python\Python38\proxyserver.py ==================
Enter a listening port: 4444
[*] Intializing socket. Done.
[*] Socket binded successfully...
[*] Server started successfully [4444]
```

Final Output



```
GET http://detectportal.firefox.com/success.txt HTTP/1.1detectportal.firefox.com 80 <socket.socket fd=85240, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM
, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1', 54048)> ('127.0.0.1', 54048)


detectportal.firefox.com
GET http://detectportal.firefox.com/success.txt HTTP/1.1detectportal.firefox.compush.services.mozilla.com
push.services.mozilla.com 443 <socket.socket fd=85120, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1
', 54049)> ('127.0.0.1', 54049)push.services.mozilla.com 443 <socket.socket fd=84416, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0
.0.1', 4444), raddr=('127.0.0.1', 54050)> ('127.0.0.1', 54050)detectportal.firefox.com 80 <socket.socket fd=85092, family=AddressFamily.AF_INET, type=SocketKind.SOCK_S
TREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1', 54051)> ('127.0.0.1', 54051)detectportal.firefox.comdetectportal.firefox.comdetectportal.firefox.com 80
<socket.socket fd=84356, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1', 54052)> ('127.0.0.1', 54052
)push.services.mozilla.com 443 <socket.socket fd=84456, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.
1', 54053)> ('127.0.0.1', 54053)detectportal.firefox.compush.services.mozilla.comdetectportal.firefox.com 80 <socket.socket fd=85332, family=AddressFamily.AF_INET, typ
e=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1', 54054)> ('127.0.0.1', 54054)detectportal.firefox.com 80 <socket.socket fd=85348, fami
ly=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1', 54055)> ('127.0.0.1', 54055)detectportal.firefox.com 80
<socket.socket fd=85356, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=('127.0.0.1', 4444), raddr=('127.0.0.1', 54056)> ('127.0.0.1', 54056
)

('127.0.0.1', 54067)('127.0.0.1', 54068)detectportal.firefox.com 80 <socket.socket fd=85124, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, laddr=
('127.0.0.1', 4444), raddr=('127.0.0.1', 54057)> ('127.0.0.1', 54057)

('127.0.0.1', 54069)

('127.0.0.1', 54070)
```

THANK YOU