SHELL SHARE

Presented By:

Lucky S. Pathan (210303105790)
Swet R. Patel (210303105787)
Vasu D. Raknoliya (210303105794)

Guided By:

Assit. Prof. Shreya Dholaria

CSE Department

PIET, Parul University

OUR TEAM

Lucky Pathan

Create a Python commond line interface for custom commonds,

Swet Patel

Front-end development for shell and server request

Vasu Rakholiya

Create the Backend of the server and Generate Log files with user DataLinks

Introduction

Shell Share is a python based project, which helps us to connect the cybersecurity team of a company with a local network (LAN) and help them find problem in the network.

As a member finds the bug he can make an announcement to the rest of the group.

Each member network data is been tracked and a detail log file is generated with a 15 min backup of the network stream, when a bug is found we can retrieve the log and network file for detail report about the bug.

As the automate testing not ensure full protection of the network, this tool for manual scanning of the network helps user to form a proper report with communicating with other users.

Problem Statement

Aim of out project is create a python tool which can help cyber security teams to clear a network with less hassle and form a super easy to communicate with other group members.

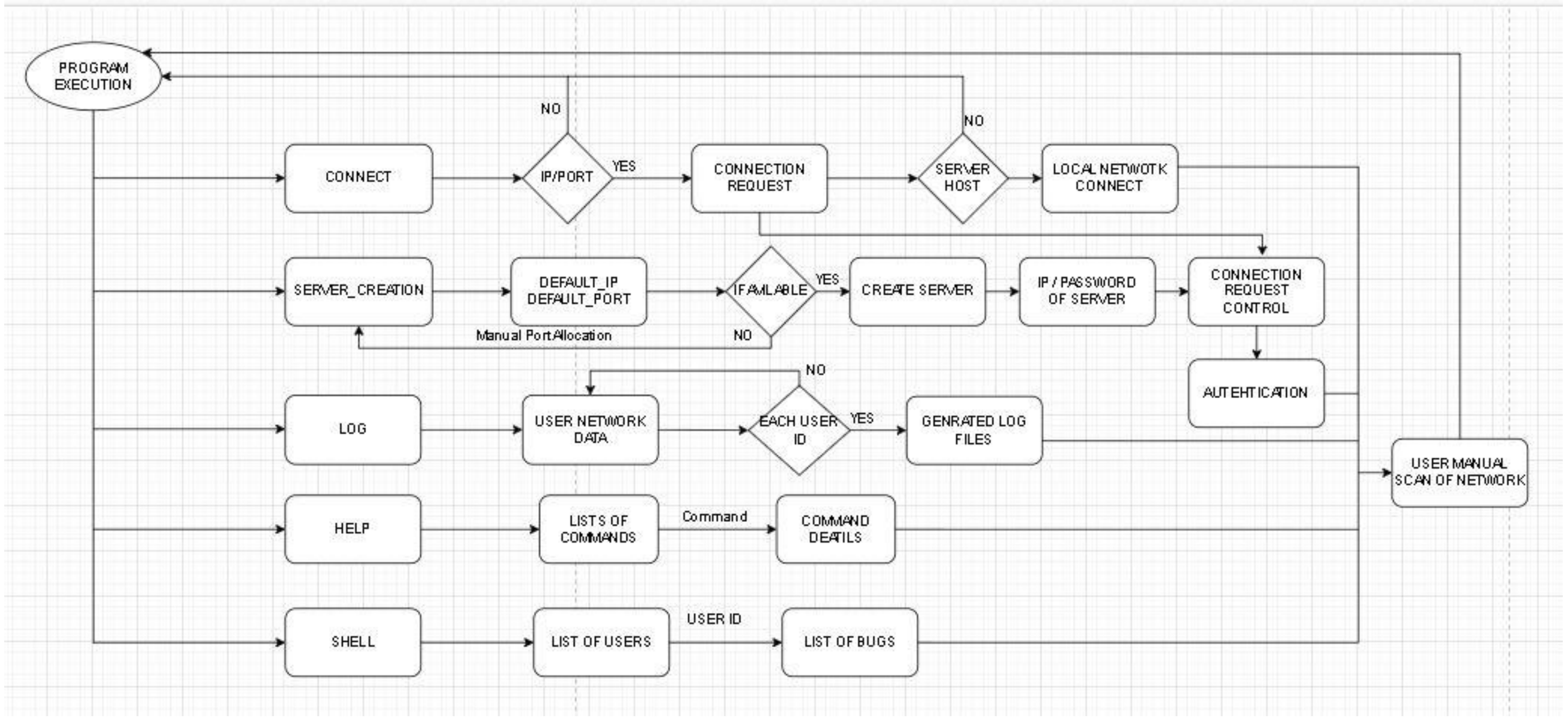
First we need to create a manual command line code for the special commands that we will implement in this project for ease of report generation and bug found list.

Parul® University NAAC A++ ACCREDITED UNIVERSITY

Summary

Sr. number	Paper Title	Publisher	Year	Take-away points
01	Implementation And Design Of Network Security	Nwauba Nnaemek Kennedy	2023	Basic encryption network with host identification
02	Active Network Monitor/Net Tracer	Lokesh Rao		Network monitoring tool, Enhance Network scanning
03	Server deployment with Python	Pietro Grandinetti		Make a custom server for request and response
04	Argparse Tutorial Python	Tshepang Lekhonkhobe		command-line options, arguments and sub-commands, components of the Argparse
05	Socket Programming in Python (Guide)	Nathan Jennings		Socket Programming

Flowchart of the system



Tools And Technology To Be Used In Project

HTTP.server

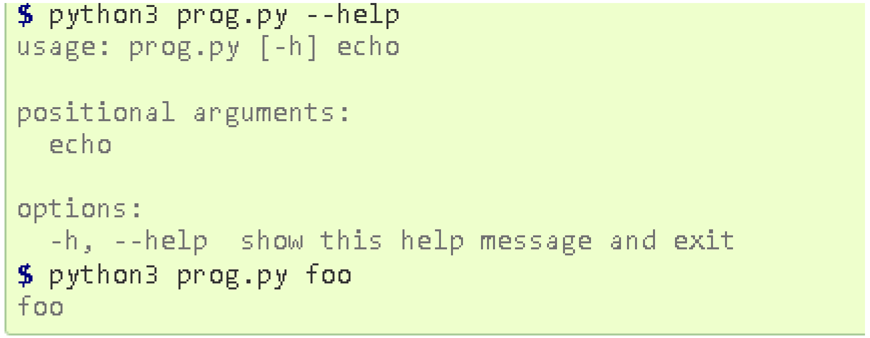
Argparse

Python

Log File Generator

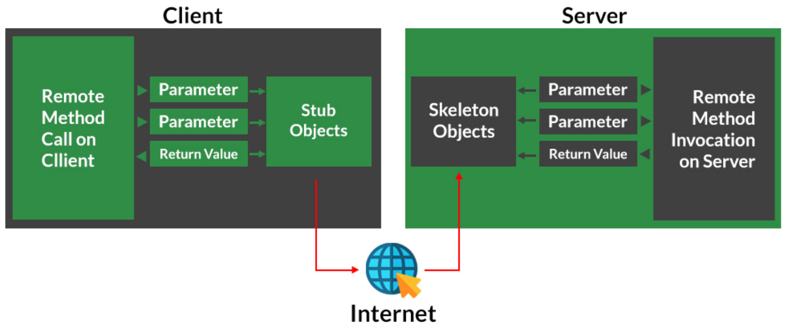
Implementation Details

1. **Python** : The creation of this custom commands we use a python library Argparse. As there are two other modules that fulfill the same task, namely getopt (an equivalent for getopt() from the C language) and the deprecated optparse. Note also that argparse is based on optparse, and therefore very similar in terms of usage. When we use argparse.ArgumentParser() and parser.parse_args() . Running the script without any options results in nothing displayed to stdout. When we add a parser.add_argument("echo") in the command output.

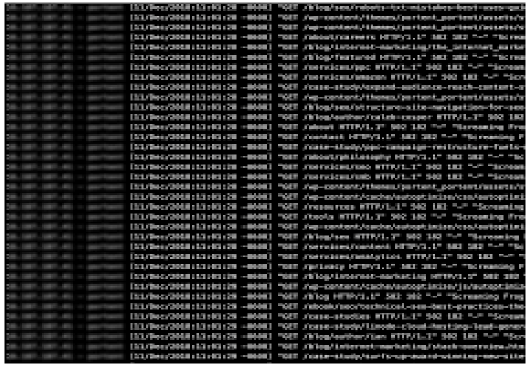


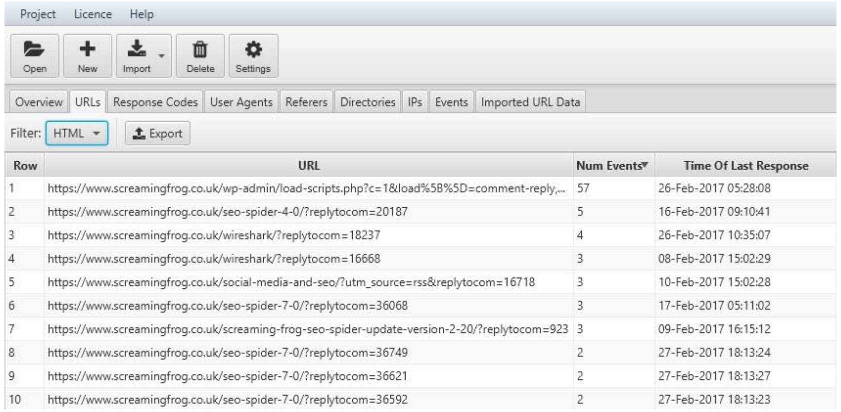
```
$ python3 prog.py --help
usage: prog.py [-h] echo

positional arguments:
  echo

options:
  -h, --help  show this help message and exit
$ python3 prog.py foo
foo
```

2. **Pryo Library**: Pyro is a library that enables you to build applications in which objects can talk to each other over the network, with minimal programming effort. You can just use normal Python method calls, with almost every possible parameter and return value type, and Pyro takes care of locating the right object on the right computer to execute the method. It is designed to be very easy to use, and to generally stay out of your way. But it also provides a set of powerful features that enables you to build distributed applications rapidly and effortlessly. Pyro is a pure Python library and runs on many different platforms and Python versions. It works between different system architectures and operating systems. Can use IPv4, IPv6 and Unix domain sockets. Optional secure connections via SSL/TL.



3. **Log File :** It is a file to which the Web server writes information each time a user requests a resource from that particular site. Log file data can offer valuable information insight into web site usage. It represents the activity of many users over a potentially long period of time . This logs collect data on the server in the files of specific formats. Measures hold information about web site usage by recording how users visit the web site and how active they are. Depending on the log format structure, different data is stored. Usually logs contain data such as: client's IP address, URL of the page requested, time when the request was send to server.


[illegible]



Filter: HTML

Row	URL	Num Events	Time Of Last Response
1	https://www.screamingfrog.co.uk/wp-admin/load-scripts.php?c=1&load%5B%5D=comment-reply,...	57	26-Feb-2017 05:28:08
2	https://www.screamingfrog.co.uk/seo-spider-4-0/?replytocom=20187	5	16-Feb-2017 09:10:41
3	https://www.screamingfrog.co.uk/wireshark/?replytocom=18237	4	26-Feb-2017 10:35:07
4	https://www.screamingfrog.co.uk/wireshark/?replytocom=16668	3	08-Feb-2017 15:02:29
5	https://www.screamingfrog.co.uk/social-media-and-seo/?utm_source=rss&replytocom=16718	3	10-Feb-2017 15:02:28
6	https://www.screamingfrog.co.uk/seo-spider-7-0/?replytocom=36068	3	17-Feb-2017 05:11:02
7	https://www.screamingfrog.co.uk/screaming-frog-seo-spider-update-version-2-20/?replytocom=923	3	09-Feb-2017 16:15:12
8	https://www.screamingfrog.co.uk/seo-spider-7-0/?replytocom=36749	2	27-Feb-2017 18:13:24
9	https://www.screamingfrog.co.uk/seo-spider-7-0/?replytocom=36621	2	27-Feb-2017 18:13:27
10	https://www.screamingfrog.co.uk/seo-spider-7-0/?replytocom=36592	2	27-Feb-2017 18:13:23

Some Features

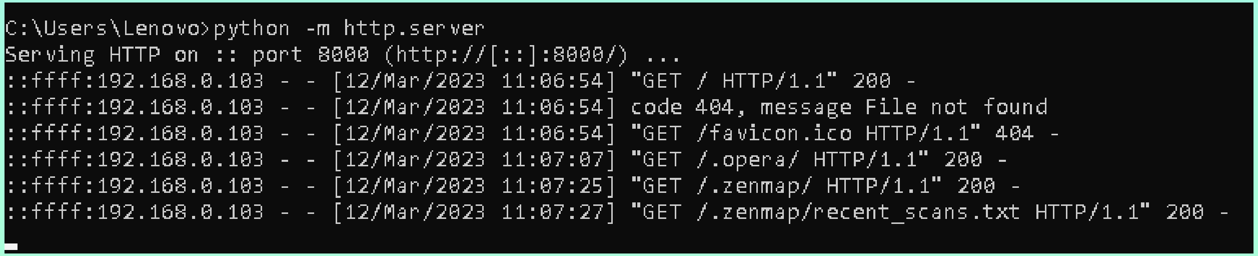
1. Log Creation
2. Network Packet Tracer
3. Host Custom Commands
4. Easy Python Deployment
5. Secure Python Environment
6. LAN Server For Connection

Screen shots of implemented work

Server Creation in Python

(I) Local Server

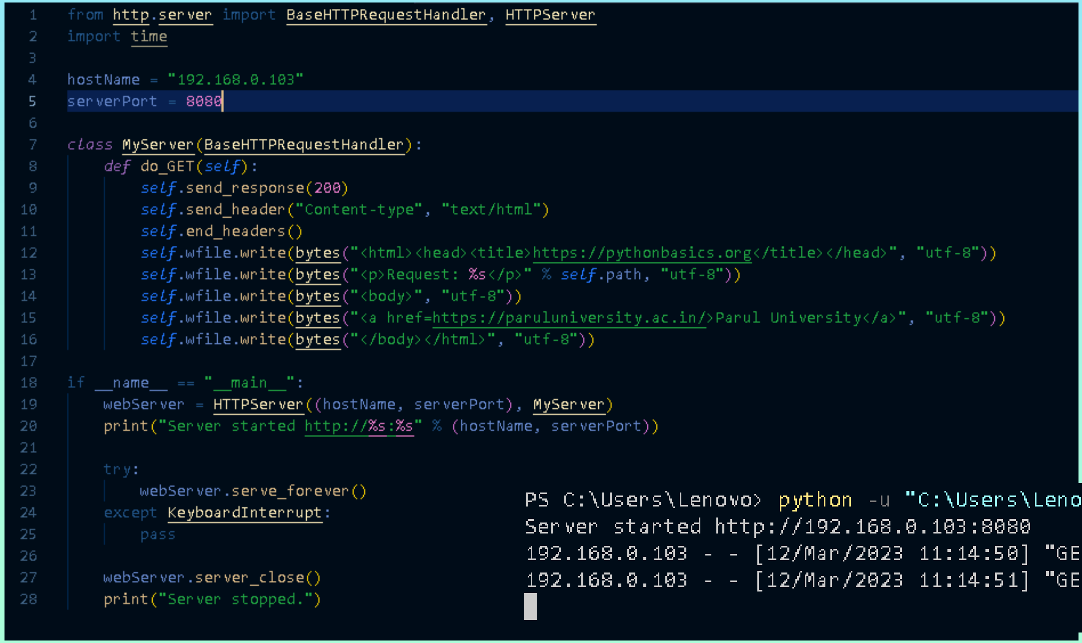
In this server client machine act as server ,we can make local server using one line code. your system becomes the server to the client that is your browser, and the files are stored locally on your system. The module you'll be using to create a web server is Python's http server. it can only be used as a static file server. You'll need a Python web framework, to run dynamic web servers.



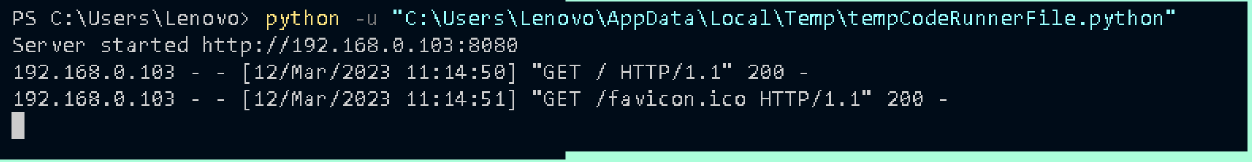
```
C:\Users\Lenovo>python -m http.server
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
::ffff:192.168.0.103 - - [12/Mar/2023 11:06:54] "GET / HTTP/1.1" 200 -
::ffff:192.168.0.103 - - [12/Mar/2023 11:06:54] code 404, message File not found
::ffff:192.168.0.103 - - [12/Mar/2023 11:06:54] "GET /favicon.ico HTTP/1.1" 404 -
::ffff:192.168.0.103 - - [12/Mar/2023 11:07:07] "GET /.opera/ HTTP/1.1" 200 -
::ffff:192.168.0.103 - - [12/Mar/2023 11:07:25] "GET /.zenmap/ HTTP/1.1" 200 -
::ffff:192.168.0.103 - - [12/Mar/2023 11:07:27] "GET /.zenmap/recent_scans.txt HTTP/1.1" 200 -
```

(II) Custom Server

In this custom server we use **local machine IP Address** as host name and use port no **8080**



```python
from http.server import BaseHTTPRequestHandler, HTTPServer
import time

hostName = "192.168.0.103"
serverPort = 8080

class MyServer(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.send_header("Content-type", "text/html")
        self.end_headers()
        self.wfile.write(bytes("<html><head><title>https://pythonbasics.org</title></head>", "utf-8"))
        self.wfile.write(bytes("<p>Request: %s</p>" % self.path, "utf-8"))
        self.wfile.write(bytes("<body>", "utf-8"))
        self.wfile.write(bytes("<a href=https://paruluniversity.ac.in/>Parul University</a>", "utf-8"))
        self.wfile.write(bytes("</body></html>", "utf-8"))

if __name__ == "__main__":
    webServer = HTTPServer((hostName, serverPort), MyServer)
    print("Server started http://%s:%s" % (hostName, serverPort))

    try:
        webServer.serve_forever()
    except KeyboardInterrupt:
        pass

    webServer.server_close()
    print("Server stopped.")
```



```
PS C:\Users\Lenovo> python -u "C:\Users\Lenovo\AppData\Local\Temp\tempCodeRunnerFile.python"
Server started http://192.168.0.103:8080
192.168.0.103 - - [12/Mar/2023 11:14:50] "GET / HTTP/1.1" 200 -
192.168.0.103 - - [12/Mar/2023 11:14:51] "GET /favicon.ico HTTP/1.1" 200 -
```

Module & classes use in server creation

- HTTP.SERVER

The HTTP server is a standard module in the Python library that has the classes used in client-server communication. it is a simple, zero-configuration command-line static HTTP server. It is powerful enough for production usage, but it's simple and hackable enough to be used for testing, local development and learning.

- BaseHTTPRequestHandler and HTTPServer

This two classes derived from HTTP. server library. This class is used to handle the HTTP requests that arrive at the server. By itself, it cannot respond to any actual HTTP requests, it must be subclass to handle each request method . BaseHTTPRequestHandler provides a number of class and instance variables, and methods for use by subclasses.

Conclusion

It also provide some methods like:

- Handle()
- send_error()
- send_response()
- send_header()
- end_header()
- log_error()
- log_message()

Our application offers an efficient and user-friendly platform for companies to conduct comprehensive resource checkups, thereby mitigating the recurrence of similar issues. Additionally, it provides detailed logs for individual users, facilitating the creation of documentation pertaining to bugs, loopholes, and errors.

References

[1]Brandon Rhodes, John Goerzen & Tim Bower, 2023, Foundations of Python Network Programming , 2rd ed., Prentice- APress.

[2]Nathan Jennings, 2023 . Socket Programming in Python 1(1.2), pp. 0-30.

[3]Pietro Grandest, 2022. Server deployment with Python: From A to Z , Prentice-Codementor.

[4]Panagiotis Chartas, 2022. Villain Tool (Updated 12 January 2023) Available at: https://github.com/t3l3machus/Villain [Accessed 30 January 2023].

[5]Python Documentation , 2022 . argparse — Parser for command-line options, arguments and sub-commands (3.2), pp. 2-30.

THANKS!

Do you have any questions?