**Internship Project Report**

## Title: GUI-Based Port Scanner with Service Detection and Multithreaded Execution

**Intern Name:** Abinaya Balakrishnan  
**Internship Period:** July 10, 2025 –July 10,2025  
**Department:** Cybersecurity and ethical hacking  
**Organization:** Tamizhan skills

**I. Introduction**

This report presents the development of a **Port Scanner with a Graphical User Interface (GUI)**, implemented using **Python’s socket and threading modules with Tkinter for the frontend**. This project was designed to provide an interactive, user-friendly way to assess a target IP address or domain for open network ports and associated services.

Port scanning is a fundamental technique in cybersecurity, often used in both penetration testing and network diagnostics. Through this project, I explored low-level socket programming, multithreading, and real-time scanning feedback—all packaged within an intuitive GUI.

**II. Background and Relevance**

Every device connected to a network communicates using ports—numbered gateways that handle different types of traffic. Attackers often scan these ports to discover vulnerabilities and entry points into systems. Tools like **Nmap** and **Zenmap** are commonly used by security professionals, but they often require command-line expertise.

This project replicates core port-scanning functionality using Python, giving users a simplified, **GUI-based alternative**. It also identifies common services like HTTP, FTP, and SSH, helping beginners understand what services are running on which ports.

The scanner uses **multithreading**, which significantly improves performance by scanning multiple ports concurrently.

**III. Objectives**

* Build a **custom port scanner** that scans any IP/domain for open TCP ports
* Identify services running on those ports (HTTP, SSH, etc.)
* Develop an **intuitive GUI** using Tkinter
* Use **multithreading** for fast concurrent scanning
* Demonstrate **ethical hacking basics** in a user-friendly tool
* Enhance awareness of **network exposure** and how services can be misused

**IV. Tools and Technologies Used**

| **Tool/Module** | **Purpose** |
| --- | --- |
| **Python 3.x** | Programming language |
| **Tkinter** | GUI development |
| **socket** | Low-level network communication (TCP) |
| **threading** | Running multiple scans concurrently |
| **ScrolledText** | To display scan output within the GUI |

**V. flowchart**

User Inputs Target

Domain/IP

Start and End Port Range

Start Multithreaded Scan

Each thread scans 1 port

Uses socket.connect()

Output in Real-Time

- OPEN or CLOSED

- With service name (e.g., HTTP)

**VI. Key Modules and Features**

**1. Graphical Interface**

* Built using **Tkinter**
* Accepts:
  + Target IP/domain
  + Port range (start to end)
* Displays results in a scrollable output pane
* Includes error checking (e.g., invalid IP, port inputs)

**2. Multithreading (Concurrency)**

* Each port scan runs in a **separate thread**
* Enables rapid scanning of large port ranges
* Prevents UI from freezing during scanning
* Uses threading.Thread() with print\_lock for safe output

**3. Service Detection**

* Associates port numbers with known services  
  Example:
* Port 80 (HTTP) is OPEN
* Port 22 (SSH) is CLOSED
* Helps identify what type of service might be exposed

**4. Real-Time Feedback**

* Users see results dynamically as scanning progresses
* Helpful for debugging and learning purposes
* Optionally shows **closed ports** for completeness

**VII. Security Relevance and Applications**

| **Use Case** | **How the Project Helps** |
| --- | --- |
| **Basic Vulnerability Check** | Detects open ports that attackers may target |
| **Student Awareness Tool** | Demonstrates how port scanning works visually |
| **Ethical Hacking Practice** | Beginner-friendly tool for internal assessments |
| **Network Troubleshooting** | Helps identify if a service is accessible |

**VIII. Testing and Output Examples**

| **Test Case** | **Result** |
| --- | --- |
| Valid IP with 20–100 ports | Displayed open/closed ports with service names |
| Invalid IP address | Handled with proper error message |
| Scanning google.com(test target) | Detected open port 80(HTTP), closed others |
| Input non-numeric ports | Error: "Ports must be valid numbers" |

**Sample Output:**

Scanning google.com (142.251.222.142) from port 20 to 100.

[OPEN] Port 80 (HTTP)

[CLOSED] Port 21 (FTP)

[CLOSED] Port 25 (SMTP)

[CLOSED] Port 20 (FTP-Data)

[CLOSED] Port 24 (Unknown)

[CLOSED] Port 22 (SSH)

[CLOSED] Port 23 (Telnet)

[CLOSED] Port 27 (Unknown)

[CLOSED] Port 26 (Unknown)

Scanning completed.

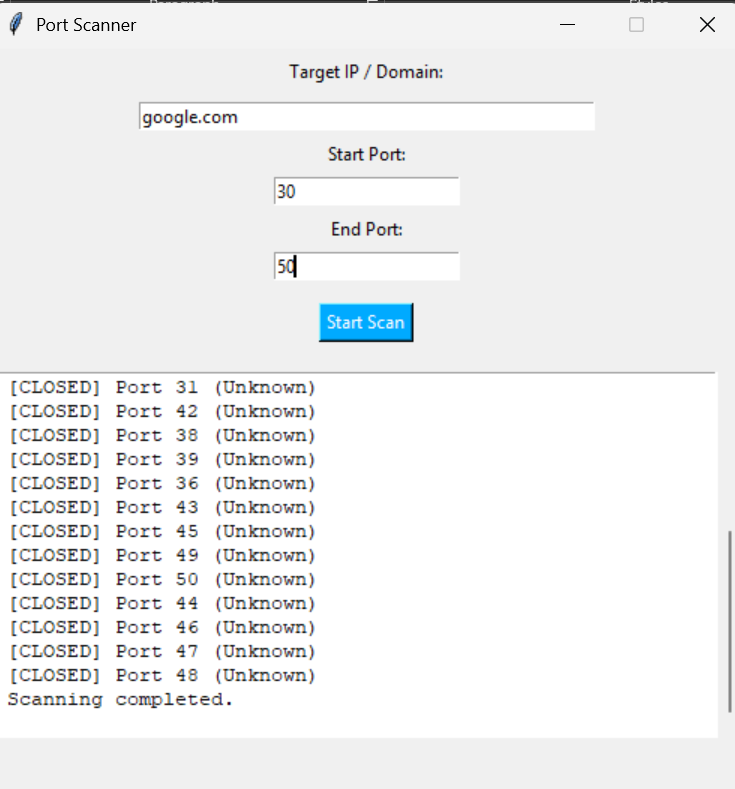
**IX. Challenges Faced & Solutions**

| **Challenge** | **Solution** |
| --- | --- |
| GUI freezing during scans | Implemented threading for parallel scans |
| Overlapping thread output | Used threading.Lock (print\_lock) to sync |
| Resolving domain to IP | Used socket.gethostbyname() |
| Making GUI scroll output clean | Used ScrolledText widget |
| Displaying service names clearly | Added dictionary mapping ports to services |

**X. What Makes This Project Stand Out**

**User-Friendly Interface:** Unlike CLI tools, this project allows even non-tech users to scan IPs easily.  
**Multithreading for Speed:** Handles 100+ ports without freezing or delay.  
**Service Identification:** Helps users understand not just what port is open—but what service it might be running.  
**Real-Time Feedback:** Immediate visibility into scan progress.  
**Educational Value:** Great for students, ethical hacking learners, and cybersecurity demonstrations.

**XI. Screenshots**



**XII. References**

1. Python socket Module – <https://docs.python.org/3/library/socket.html>
2. Tkinter GUI Docs – <https://docs.python.org/3/library/tkinter.html>
3. Common Port Numbers – <https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers>
4. OWASP Top 10 – <https://owasp.org/www-project-top-ten/>