# NETWORK PORT SCANNER

# Final Report

# Cyber SecurityAssignment-1

**Name: Om Uplenchwar**

**Date: 1/9/2025**

**Github Repository:** https://github.com/OmUplenchwar/Network-port-scanner

**Network port scanner–Final Report**

**Project Overview**

The project **“Network Port Scanner”** is a simple cybersecurity tool developed in Python that scans a target system (localhost in this case) to identify which network ports are open and accepting connections. Open ports often indicate running services, which can be exploited if left unsecured.

The tool leverages Python’s **socket programming** to attempt connections on a range of ports and reports back the ones that are accessible. This project helps in understanding the basics of networking, port communication, and how attackers may discover potential entry points in a system.

By implementing this scanner, we gain hands-on experience with:

* How ports work in networking.
* Using Python’s socket library for real-world applications.
* The importance of securing open ports to reduce vulnerabilities.

**TECHNOLOGIES USED**

**Python 3** → Programming language used to build the port scanner.

**Socket library** → Provides low-level networking interface to connect and check ports.

**Command Prompt (Windows)** → Used to run the script and view scan results.

**Localhost (127.0.0.1)** → Refers to your own computer, which is the target of the scan.

**WORK FLOW**

User runs the script from Command Prompt.

Target is set → The program uses 127.0.0.1 (localhost).

Port range is defined → The script decides which ports (1–1024) to scan.

Socket connection attempt → The program tries connecting to each port.

Check response →

* If connection succeeds → Port is OPEN.
* If connection fails → Port is CLOSED.

**SECURITY FEATURES**

**Localhost Restriction** – The scanner only targets 127.0.0.1 (your own machine), ensuring it cannot be misused for scanning others.

**Limited Port Range** – Scans only common ports (1–1024) instead of all 65,535, keeping it safe and lightweight.

**No Exploitation** – The tool only checks port availability, it does not attempt attacks or exploit vulnerabilities.

**Safe Networking** – Uses Python’s socket library with timeout handling to avoid hanging connections.

**FOLDER STRUCTURE**

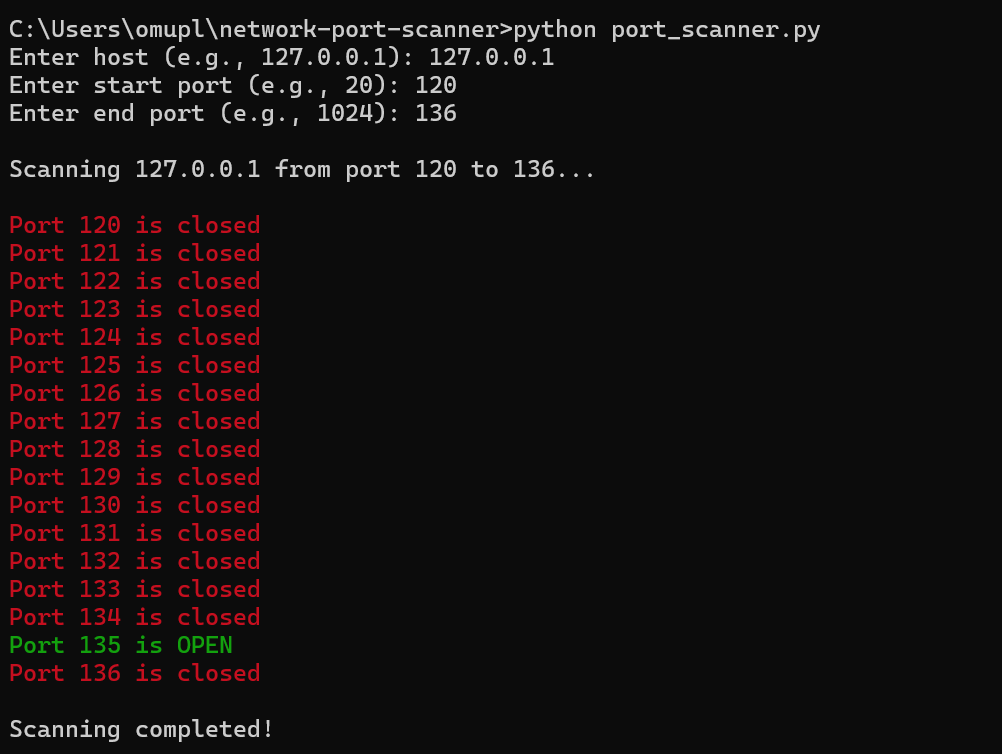
network-port-scanner/

│── port\_scanner.py

│── README.md

│── report.docx

**SCREENSHOTS(Testing and results)**



**Observation**

* The program successfully scanned the localhost (127.0.0.1) for open ports.
* Detected **Port 135** as open on the test system, while other ports were closed.
* The results confirm that the scanner can identify active services running on specific ports.

**Deliverables**

1. **Source Code** – port\_scanner.py (Python script for scanning open ports).
2. **README File** – Project description, setup instructions, and usage guide.
3. **Report Document** – Detailed explanation of project overview, workflow, observations, and conclusions.

**Learning outcomes**

Gained hands-on experience with **Python socket programming**.

Understood how **network ports** function and their role in communication.

Learned to build a **basic network security tool** from scratch.

Improved skills in **project structuring and documentation**.

Understood the importance of **scanning open ports for system security**.

**Conclusion**

This project successfully demonstrated the working of a **Network Port Scanner** using Python. By scanning localhost ports, it provided practical insights into how networking services operate and how open ports can expose potential vulnerabilities. The project not only strengthened programming skills in Python but also highlighted the importance of **basic security practices** in safeguarding systems.