Penetration Testing Tool Project Report

Tool Name: **Advanced Network Scanner**
Purpose: A Python-based penetration testing tool that automates network discovery, port scanning, and vulnerability assessment using Nmap.

---

1. Installation Instructions

Prerequisites

- Python 3.x
- Nmap installed (sudo apt install nmap on Kali Linux)
- Required Python packages

Installation Steps

1. **Clone the repository (if applicable)**

bash

Copy

```
git clone https://github.com/your-repo/cybersecurity-tool.git
cd cybersecurity-tool
```

2. **Set up a virtual environment (recommended)**

bash

Copy

```
python3 -m venv venv
source venv/bin/activate
```

3. **Install dependencies**

bash

Copy

```
pip install python-nmap requests prettytable jinja2
```

4. **Run the tool**

bash

Copy

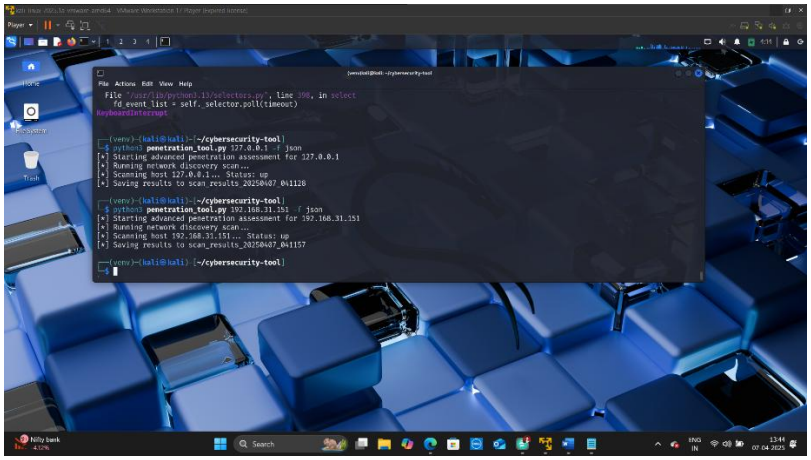
python3 penetration_tool.py <TARGET_IP/RANGE> -o <OUTPUT_DIR> -f <FORMATS>

Example:

bash

Copy

python3 penetration_tool.py 192.168.1.1-100 -o scan_results -f json html



---

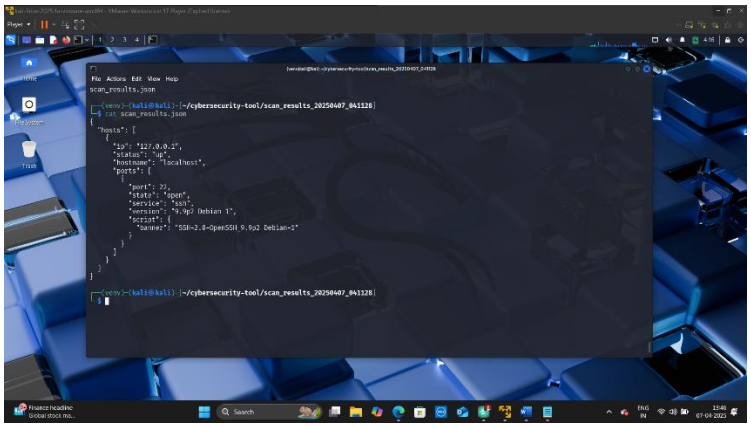
2. Code Breakdown & Explanation

Key Components

1. **PenetrationTool Class**
    - Handles scanning, parsing, and reporting.
    - Uses python-nmap for Nmap integration.
2. **_network_discovery()**
    - Performs host discovery (-sn) followed by port scanning (-sV --script).
    - Checks host status before scanning ports.
3. **_get_port_info()**
    - Extracts port details (TCP/UDP) and service versions.
4. **_parse_nmap_scripts()**
    - Handles both **dictionary** (modern) and **string** (legacy) Nmap script outputs.
5. **_save_results()**
    - Supports multiple formats (JSON, CSV, HTML).
    - Uses PrettyTable for clean HTML report

- **3. Screenshots of Execution**

Scan Execution



Running the tool against 127.0.0.1 with JSON output.

JSON Output

json

cat scan_results.json


```
{
  "hosts": [
    {
      "ip": "127.0.0.1",
      "status": "up",
      "hostname": "localhost",
      "ports": [
        {
          "port": 22,
          "state": "open",
          "service": "ssh",
          "version": "9.9p2 Debian 1",
          "script": {
            "banner": "SSH-2.0-OpenSSH_9.9p2 Debian-1"
          }
```

```
      }
    ]
  }
 ]
}
```


Example JSON output showing open ports.

4. Challenges Faced & Solutions

Challenge	Solution
Nmap script output sometimes returns a dict instead of a str, causing crashes.	Added checks for isinstance(script_output, dict) and handled both formats.
Hosts not responding caused KeyError in scan results.	Implemented if host in self.scanner.all_hosts() before processing.
CSV/HTML output formatting was messy.	Used PrettyTable for structured HTML tables and proper CSV escaping.
Slow scans on large networks.	Optimized Nmap arguments (-T4 for faster scans).

5. Future Improvements

1. Enhanced Scanning Features

- **Parallel scanning** for large networks (multithreading).
- **Customizable Nmap arguments** via command line (--scan-type).

2. Better Reporting

- **PDF reports** using ReportLab.
- **Vulnerability scoring** (CVSS integration).

3. Automation & Integration

- **Schedule scans** (e.g., daily checks).
- **API integration** (e.g., VulnDB, Shodan).

4. User Experience

- **Interactive CLI** with --help and progress bars.
- **GUI version** using PyQt or Tkinter.

Conclusion

This tool simplifies network reconnaissance by automating Nmap scans and generating structured reports. Future versions could include exploit suggestions and integration with SIEM tools.