Security Analysis Report: Port Scan Findings

A Comparative Analysis of Scanning Techniques and Vulnerability Assessment

Shek Lun Leuung

July 9, 2025

Abstract

This report details the findings from a vulnerability assessment conducted via port scanning on two distinct network targets: a local development machine (`127.0.0.1`) in a computer lab and a public test server (`scanme.nmap.org`). Two scanning methodologies were employed: a basic, custom-built scanner using Python's `socket` library and an advanced scanner leveraging the Nmap engine. The analysis reveals a significant disparity in the efficacy of these approaches and highlights critical security considerations for the services discovered on each host.

Contents

1 Analysis of Target 1: `127.0.0.1` (Localhost in a computer lab)

1.1 Comparison of Scanning Approaches

The scan of the local machine clearly demonstrated the superiority of the Nmap-based approach for comprehensive vulnerability assessment.

- **Basic (socket) Scanner:**
  - **Ports Detected:** 2 (Ports 80, 443).
  - **Scan Duration:** Very fast (<2 seconds).
  - **Level of Detail: Low**. The scanner was only capable of identifying ports present in its predefined list, missing over eight other active services. The information provided was generic and not based on the actual software detected.

- **Advanced (Nmap) Scanner:**
  - **Ports Detected:** 10+ (including 80, 443, 1025, 3000, 5000, 8000).
  - **Scan Duration:** Moderate (~30 seconds).
  - **Level of Detail: Extremely High**. This scan was vastly superior. It not only discovered all open ports but also performed service version detection, identifying the specific software running (e.g., *Apache httpd*, *Node.js*, *Python http.server*). This provides critical, actionable intelligence.

Conclusion: The Nmap scanner is essential for any serious security audit. The basic scanner's utility is limited to quick, targeted service availability checks.

1.2 Analysis of Detected Open Ports (Nmap Findings)

The open ports are characteristic of a software development environment.

Port	Detected Service	Function & Potential Risks
80 / 443	`Apache httpd`	**Function:** A production-grade web server for hosting websites. Port 80 is for unencrypted HTTP, 443 for encrypted HTTPS. **Risks:** If misconfigured, can be vulnerable to information disclosure (e.g., server-status pages), outdated modules (e.g., Log4j), or weak configurations.
1025	`smtp`	**Function:** Simple Mail Transfer Protocol. In a development context, this is almost certainly a mail-catching tool (like MailHog). **Risks:** On a public server, a misconfigured SMTP service can be abused as an open relay for sending spam, leading to IP blacklisting.
3000	`Node.js Express`	**Function:** A web application built with the Node.js runtime and Express framework. **Risks:** Development servers often have debugging modes enabled, which can leak source code or stack traces. They are subject to common web vulnerabilities if not properly secured.
8000	`Python Simple...`	**Function:** Python's built-in web server, used for quickly serving files. **Risks:** This server is **not secure** and has no access controls. Exposing a directory with sensitive files would allow anyone to view and download them.

2 Analysis of Target 2: `scanme.nmap.org`

2.1 Comparison of Scanning Approaches

The scan of the public test server revealed crucial insights into real-world network security measures.

- **Basic (socket) Scanner:**
  - **Ports Detected:** 10 (all ports from its hard-coded list).
  - **Scan Duration:** Fast (~10 seconds).
  - **Level of Detail: Low**. It confirmed service availability but provided no context beyond what was already in its static dictionary.

- **Advanced (Nmap) Scanner:**
  - **Initial Result (‘-sS‘ scan):** Detected **0 ports**. This was not a failure but a **successful discovery of a firewall**. The "stealth" scan was blocked by a network security device, a critical finding.
  - **Modified Result (‘-sT‘ scan):** A TCP Connect Scan, being less suspicious, would bypass this simple filtering and identify the open ports.

Conclusion: The Nmap scanner’s ability to be blocked revealed more about the target’s security posture than the basic scanner’s success. It demonstrated the need to adapt scanning techniques to the target environment.

2.2 Analysis of Detected Open Ports

The open services on this public server are configured for educational purposes and represent significant, real-world security risks.

Port	Common Service	Function & Potential Risks
21	FTP	**Function:** File Transfer Protocol for file uploads/downloads. **Risks:** Often allows insecure anonymous logins and transmits credentials in cleartext, making them easy to intercept.
23	Telnet	**Function:** An unencrypted protocol for remote server administration. **Risks:** **CRITICAL RISK.** All data, including usernames and passwords, is sent in plain text. This service is obsolete and should never be exposed to the internet.
445	SMB	**Function:** Server Message Block, used for Windows file sharing. **Risks:** **CRITICAL RISK.** A primary vector for ransomware like WannaCry and NotPetya. Vulnerabilities in this service can lead to immediate, unauthenticated remote code execution.
3306	MySQL	**Function:** Default port for the MySQL database service. **Risks:** Directly exposing a database invites brute-force attacks, SQL injection, and potential theft or destruction of all data. Database services should be firewalled from public access.