 ISSN: 3107-6513

Automated Nmap Scanner for Structured Network Reconnaissance

Adhithya S
B.Sc. Digital and Cyber Forensic Science, Rathinam College of Arts and Science, Coimbatore, India
adhithyasukumaran623@gmail.com

Abstract—This paper presents an automated Nmap scanner developed with Python and Pandas to support structured network reconnaissance, repeated exposure monitoring, and attack surface assessment in controlled environments. The framework automates seven core scan phases: host status detection, operating system fingerprinting, full port scanning, SYN stealth scanning, service and version detection, aggressive scanning, and safe Nmap Scripting Engine checks. Each scan cycle is stored in structured JSON format and normalized into Pandas DataFrames for comparison and reporting. The system reduces repetitive analyst effort, improves consistency across repeated assessments, and helps identify meaningful changes such as newly exposed ports, service modifications, and stable baseline behavior.

Keywords—Nmap, Python automation, Pandas, network reconnaissance, attack surface assessment, exposure monitoring, vulnerability scanning, cybersecurity reporting

I. INTRODUCTION

Network reconnaissance is a fundamental activity in cybersecurity because it reveals which hosts are reachable, which ports are exposed, and which services are available for interaction or misuse. In many environments, analysts still run multiple Nmap commands manually, save outputs separately, and compare results by inspection. This process is repetitive, time-consuming, and difficult to maintain when the same targets must be reviewed more than once.

The automated Nmap scanner in this study addresses that problem by combining Nmap as the scanning engine, Python as the orchestration layer, and Pandas as the analysis layer. The objective is to transform one-time scanning into a structured workflow that supports baseline creation, repeated exposure monitoring, and report-ready output for internal or laboratory assessment [1]-[3].

II. BACKGROUND

A. Nmap

Nmap is a widely used network discovery and security auditing tool that supports host discovery, open-port identification, service enumeration, version detection, operating system fingerprinting, traceroute, and script-based checks [1]. Its output is reliable for both one-time reconnaissance and repeated automated assessment, making it suitable for attack surface analysis.

B. Python

Python coordinates the complete workflow by launching scan phases, collecting output, storing results, and supporting repeated execution. Without automation, the analyst must re-enter commands, manage files manually, and compare scan results visually. Python standardizes these steps and makes repeated assessment practical [2].

C. Pandas

Pandas converts semi-structured scan output into tabular data that can be filtered, compared, and documented efficiently. By loading JSON-based scan records into DataFrames, the framework improves readability and enables easier identification of newly opened ports, removed services, and unchanged baseline conditions [3].

III. SYSTEM METHODOLOGY

The scanner is organized into seven scan phases that progressively build a complete exposure profile of the target. Instead of depending on a single scan mode, the framework combines lightweight discovery, deeper enumeration, and safe script-based checks to improve completeness and analytical value.

A. Host Status Detection

Host status detection is the first phase because it confirms whether a target is active and reachable before deeper probing begins. This improves efficiency by limiting advanced scans to live systems only.

B. OS Detection

Operating system detection uses Nmap fingerprinting to estimate the host platform by analyzing responses to crafted packets. This context helps analysts interpret open ports and services more accurately because expected exposure often varies between Windows and Linux hosts.

C. Full Port Scan

The full port scan examines the complete TCP port range to identify open, closed, and filtered ports. This phase is essential for accurate attack surface assessment because quick scans may miss services that operate on uncommon or non-standard ports.

D. SYN Stealth Scan

The SYN stealth scan uses half-open probing to identify port states without completing the full TCP three-way handshake. It provides faster visibility with lower overhead and is useful for repeated scanning where speed and reliability are both important.

E. Service and Version Detection

Service and version detection enriches the scan results by identifying the application behind each open port and, when possible, the version that is running. This turns raw port

numbers into actionable findings that can be prioritized for review or remediation.

F. Aggressive Scan

Aggressive scanning combines deeper service detection, operating system identification, traceroute, and selected script-based observations. Although more resource-intensive than lighter scans, it helps build a fuller profile of important systems during detailed assessment.

G. Safe NSE Vulnerability Scan

The Nmap Scripting Engine phase extends basic reconnaissance with safe script-based checks that can highlight weak configurations, protocol issues, or signs of security exposure without attempting intrusive exploitation [1].

IV. Implementation Architecture

The implementation architecture combines Nmap for scan execution, Python for automation, JSON for structured storage, and Pandas for normalization and comparison. The workflow begins with target definition and scan interval configuration, then launches the selected scan phases and stores each cycle as a separate record instead of overwriting prior output.

In the tested workflow, repeated scans were scheduled at fixed intervals such as 300 seconds. This allowed the system to compare one scan cycle with another and determine whether new ports appeared, services changed, or the exposure profile remained stable.

V. Scan Findings and Analysis

The scanner produces findings related to host availability, open ports, running services, service versions, and scan-to-scan stability. These findings represent the real network footprint visible from the scan vantage point and therefore support both exposure verification and security review.

A key analytical observation is that open ports do not all carry the same meaning. The significance of a finding depends on the service behind the port, the sensitivity of the access it enables, and whether that exposure is expected within the environment.

VI. Attack Surface and Risk Classification

In this study, attack surface is treated as the collection of reachable ports, services, and protocols that may provide paths for observation, access, misuse, or exploitation. The framework therefore goes beyond raw data collection and supports interpretation of which services are expected, which are sensitive, and which deserve stronger security attention.

Representative risk categories identified during the assessment are summarized below.

- **HTTP/HTTPS:** Web-facing application exposure; usually low to medium risk when the service is expected and properly managed.
- **SSH:** Remote administrative access; medium risk when exposed beyond the intended management boundary.
- **FTP/Telnet:** Legacy or weakly protected remote access and file transfer; high risk because credentials may be exposed in plain text.
- **SMB/RDP:** Administrative or file-sharing functionality; high risk because these services are often targeted for lateral movement and brute-force attempts.
- **Database ports:** Potential direct access to data services such as MySQL, PostgreSQL, Microsoft SQL Server, or Oracle; generally high risk when reachable unnecessarily.

VII. Manual and Automated Time Comparison

Manual scanning requires the analyst to run commands separately, inspect outputs one by one, save results manually, and later rework those findings into a report-friendly format. When repeated assessments are needed, this effort grows quickly because the same steps must be repeated for every scan cycle.

In contrast, the automated workflow links discovery, deeper scanning, result storage, parsing, and comparison into a single script-driven process. This reduces repetitive effort, improves consistency, and makes recurring monitoring more efficient because historical outputs are already preserved in structured form.

VIII. Results and Discussion

The automated framework successfully supported repeated scan cycles and stored each cycle as a separate JSON record for later analysis. This design moved the project beyond one-time reconnaissance and into repeated monitoring, which is more useful for baseline validation and controlled attack surface review.

When the JSON outputs were normalized into Pandas DataFrames, the results became easier to compare and document. Even when no major differences were detected across scan intervals, the repeated scans still provided value by establishing what normal exposure looked like for the target environment.

IX. Conclusion

The automated Nmap scanner demonstrates how a traditional reconnaissance tool can be extended into a repeatable monitoring and reporting framework through the integration of Nmap, Python automation, structured JSON storage, and Pandas-based analysis. Across seven scan phases, the system improves consistency, reduces repetitive manual effort, and preserves scan history for later comparison.

The project is especially useful for internal network visibility, repeated host assessment, and evidence-oriented reporting in controlled environments. Its broader contribution is that automation not only saves time but also improves analytical quality by standardizing scan execution and making findings easier to prioritize and communicate.

References

[1] G. F. Lyon, Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning. Nmap Project.
[2] Python Software Foundation, Python Documentation.
[3] W. McKinney, pandas: powerful Python data analysis toolkit.
[4] K. Scarfone and P. Mell, Guide to Intrusion Detection and Prevention Systems (IDPS). National Institute of Standards and Technology.
[5] OWASP Foundation, OWASP Testing Guide.
[6] Adhithya S., Automated Advanced Nmap Scanner Full Reconnaissance, project report, Rathinam College of Arts and Science, 2026.