Dydaktyka Informatyki 19(2024)
ISSN 2083-3156; e-ISSN 2543-9847
DOI: 10.15584/di.2024.19.19
http://www.di.ur.edu.pl

Uniwersytet Rzeszowski

Jacek WOŁOSZYN[1], **Michał WOŁOSZYN**[2]

[1] *ORCID: 0000-0003-4340-9853. Dr inż., Uniwersytet Radomski, Wydział Informatyki i Matematyki, Katedra Informatyki, ul. Malczewskiego 20A; 26-600 Radom; e-mail: jacek.woloszyn@uthrad.pl*
[2] *BSc, student, Goldsmiths, University of London, 8 Lewisham Way, London SE 14 6NW; e-mail:mwolo001@gold.ac.uk*

data złożenia tekstu do Redakcji DI: 5.04.2024; data wstępnej oceny artykułu: 12.04.2024

USING NMAP AND PYTHON FOR AN AUTOMATED NETWORK SECURITY AUDIT

WYKORZYSTANIE NMAP I PYTHONA DO ZAUTOMATYZOWANEGO AUDYTU BEZPIECZEŃSTWA SIECI

Keywords: Python, Nmap, cybersecurity.
Słowa kluczowe: Python, Nmap, cyberbezpieczeństwo.

Abstract

This article describes how the practical application of Nmap and Python can revolutionize the approach to cybersecurity, offering insight into specific techniques, scripts, and strategies for using these tools to enhance network security. Through in-depth analysis and use cases, this article aims not only to demonstrate the potential of combining these two powerful tools, but also to inspire you to use them to build more secure, resilient environments. The following sections describe the basics of Nmap, Python integrations, and the use case.

Streszczenie

W niniejszym artykule opisano, jak praktyczne zastosowanie Nmap i Pythona może zrewolucjonizować podejście do cyberbezpieczeństwa. Wgląd w konkretne techniki, skrypty i strategie wykorzystania tych narzędzi ma na celu zwiększenie bezpieczeństwa sieciowego. Poprzez dogłębną analizę i przykłady zastosowań owych technologii artykuł ten ma za zadanie nie tylko przedstawić potencjał wynikający z połączenia tych dwóch potężnych narzędzi, ale także zainspirować do ich wykorzystania w celu budowania bardziej zabezpieczonych, odpornych na ataki cyfrowe środowisk. W kolejnych rozdziałach opisano podstawy Nmap, integracje z Pythonem i przypadki użycia.

Introduction

In today's fast-paced digital world, where every organization, regardless of size or industry, faces challenges related to the security of their networks, network mining and auditing tools are becoming essential. Among the many technologies available, Nmap (Network Mapper) stands out as one of the most powerful tools for scanning networks, detecting devices and services running on a network, as well as identifying potential vulnerabilities. Its flexibility and wealth of features make it an invaluable resource for cybersecurity professionals. However, the true potential of Nmap is revealed when used in conjunction with Python, a programming language valued for its simplicity, readability, and powerful libraries that enable the automation of complex tasks.

Nmap's integration with Python opens up new horizons for penetration testing automation, security audits, and network monitoring. Thanks to this synergy, users can create complex scripts that automate routine scanning tasks, analyze the data obtained, and generate detailed reports. The ability to programmatically control the scanning process and analyze the results, without the need for manual intervention, is a great added value, saving time and resources, as well as increasing the precision of the operations performed.

Using Python to extend Nmap's functionality allows you to create personalized solutions tailored to the specific needs of your organization. This can include automatic detection of new devices on the network, identification of changes in service configuration, and even advanced vulnerability analysis using available databases of known security concerns. This allows cybersecurity professionals to not only respond to current threats, but also proactively counter potential attacks before they become a real problem.

1. Nmap Basics

Nmap[1] (Network Mapper) is an advanced and versatile network scanning tool that has become an indispensable part of every cybersecurity professional's arsenal. Its primary function is to discover devices running on the network and identify open ports and running services. Nmap can also be used to detect server software versions, device types, operating systems, and possible weaknesses, making it an invaluable tool in assessing an organization's network security posture.

[1] J. Chirillo, S. Blaul, "Hack Attacks Revealed: A Complete Reference with Custom Security Hacking Toolkit", Wiley, 2002; Fyodor, "Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning", Insecure.Com LLC, 2009.

Features & Capabilities of Nmap

Host discovery

Nmap allows you to quickly scan your network to identify active hosts. This allows users to get an overview of all devices connected to a given network, which is the first step in the security audit process.

Port Scanning

One of the main features of Nmap is the ability to scan TCP and UDP ports to identify open ports that can serve as potential entrances for attackers. Nmap offers a variety of scanning techniques, including SYN (semi-open) scanning, TCP connect (full connection open), UDP scanning, and more, allowing you to tailor the process to your specific needs and limitations.

Service and version discovery

The tool can also identify what services are running on open ports, along with information about software versions. This feature is essential for identifying potentially outdated or vulnerable software that requires updates or additional security.

Detecting Operating Systems

Nmap can perform advanced scans to determine what operating systems are being used by identified hosts, which provides valuable information about potential attack vectors.

Nmap Scripting Engine (NSE)

One of Nmap's most powerful features is its NSE scripting engine, which allows you to run pre-built or custom scripts for a variety of tasks, such as advanced vulnerability detection, task automation, or gathering additional network information.

Usage examples

A basic network scan to detect active hosts can be performed using the command below

```
nmap -sn 192.168.1.0/24
```

This command uses a "ping" scan (-sn option) to quickly identify devices on the local network with addresses between 192.168.1.1 and 192.168.1.254.

To scan a specific host for open TCP ports, use the

```
nmap -p 1-65535 192.168.1.1
```

This command examines all possible ports (from 1 to 65535) on a device with an IP address of 192.168.1.1.

Advanced scanning to identify services and their versions can be performed with the command

```
nmap -sV 192.168.1.1
```

where the -sV option activates version detection of services running on open ports of the selected host.

Nmap provides a tooling foundation for cybersecurity professionals, enabling them to deeply analyze and understand the structure and potential weaknesses within the scanned networks. Thanks to the versatility of the features it offers, users can tailor scanning to their specific requirements and expectations, making Nmap an indispensable tool in the process of identifying threats and managing cyber risks.

Knowledge and skillful use of Nmap allows for a comprehensive assessment of the security of IT infrastructure, from discovering devices and services operating in the network, through identification of open ports, to advanced analyses of software versions and potential vulnerabilities[2]. Integration with the Nmap Scripting Engine opens up additional capabilities, enabling custom scanning tasks and automating threat detection and response processes.

The practical application of Nmap in combination with the Python programming language significantly expands the possibilities of network analysis and automation of security tasks, which is especially valuable in a rapidly changing cyber environment. By writing Python scripts that integrate Nmap functions, specialists can build their own customized security monitoring tools, which significantly increases the efficiency and effectiveness of defending against potential attacks.

Nmap is a key component in the arsenal of cybersecurity tools, offering not only advanced network scanning and analysis capabilities, but also the ability to customize and automate security processes. Its flexibility and potential to

[2] D. Kennedy, J. O'Gorman, D. Kearns, M. Aharoni, "Metasploit: The Penetration Tester's Guide", No Starch Press, 2011; J. Erickson, "Hacking: The Art of Exploitation, 2nd Edition", No Starch Press, 2008.

integrate with other tools and programming languages, such as Python, underscore the value of Nmap as a must-have tool for anyone professionally involved in network security.

2. Nmap Integration with Python

The integration of Nmap with Python opens up new horizons of efficiency and opportunities for cybersecurity professionals. By leveraging Python, a language of great power and flexibility, users can significantly extend the functionality of Nmap by automating complex scanning, data analysis, and reporting processes. This synergistic relationship allows for the creation of custom, highly automated security solutions that are able to meet the specific requirements and challenges of each organization.

Scan Automation

A fundamental aspect of integrating Nmap with Python is the ability to automate network scanning processes. Using Python libraries such as python-nmap, professionals can programmatically perform Nmap scans, process the results, and interpret them in an automated manner. This provides great opportunities for continuous network security monitoring, allowing you to identify new devices on your network, changes in service configuration, and even potential vulnerabilities in real-time.

Processing and analysis of results

Integration with Python also enables advanced processing and analysis of scan results. With powerful data processing libraries like Pandas, professionals can easily manipulate, filter, and analyze data, which is crucial for effectively identifying threats and vulnerabilities in the network. Automating this analysis allows you to react quickly to potential threats, minimizing the time needed to take corrective action.

Extension of functionality

With Python, Nmap users can also create their own custom scripts and tools that extend the functionality of basic scanning. This can include integrating with other security tools, automating complex test scenarios, or even creating user interfaces for more interactive analytics. The ability to integrate Nmap with the broad ecosystem of tools and technologies available in Python greatly increases the value of both tools, enabling the creation of end-to-end security solutions.

Usage examples

Consider a scenario where an organization wants to perform regular scans[3] of its network for new devices and potential vulnerabilities. Using Python, you can write a script that automatically runs Nmap scans, analyzes the results for known vulnerabilities (using databases such as NVD – National Vulnerability Database), and generates reports that are then sent to the security team.

```
nmap import
Import pandas as PD

nm = nmap. PortScanner()

nm.scan('192.168.1.0/24', arguments='-sV')

# Processing results
hosts_list = [(x, nm[x]['status']['state'], nm[x].all_protocols(), nm[x]['hostnames']) for x in nm.all_hosts()]
df = pd. DataFrame(hosts_list, columns=['Host', 'Status', 'Protocols', 'Hostnames'])

print(df)
```

Listing 1. An example of a script to scan itself and create a simple report in DataFrame format.

Source: Author's own elaboration

Consider a more advanced scenario where an organization wants to monitor its network environment for specific vulnerabilities that have recently been disclosed. Using Python[1], you can create a script that not only runs Nmap scans for open ports, but also uses the Nmap Scripting Engine (NSE) to perform a deeper analysis of potential vulnerabilities and automatically generates alerts when they are detected.

Example of a Python script using NSE to check for ransomware vulnerability on Server Message Block (SMB) servers:

```
nmap import
def scan_vulnerabilities(target):
```

[3] J. Beale, A. Baker, B. Caswell, "Snort: IDS and IPS Toolkit", Syngress, 2007; S. Greenblatt, "Cybersecurity: The Beginner's Guide: A comprehensive guide to getting started in cybersecurity", Packt Publishing, 2019.

```
    # Create a Nmap scanner object
    nm = nmap. PortScanner()

    nm.scan(target, arguments='-p 445 --script=smb-vuln-*')

    # Cycle through the results and view vulnerability information
    for host in nm.all_hosts():
        print(f'Host: {host} ({nm[host].hostname()})')
        print(f'State: {nm[host].state()}')
        for proto in nm[host].all_protocols():
            print(f'Protocol: {proto}')
            lport = nm[host][proto].keys()
            for port in lport:
                print(f'Port: {port}\tState: {nm[host][proto][port]["state"]}')
                for script, output in nm[host][proto][port]['script'].items():
                    print(f'{script}: {output}')

scan_vulnerabilities('192.168.1.0/24')
```

Listing 2. An example of a script that finds vulnerabilities.

Source: Author's own elaboration

This script automatically scans all devices on a specific network for open port 445, used by the SMB protocol, which is often used by ransomware attacks such as WannaCry. Then, using the smb-vuln-* NSE scripts, the script analyzes potential SMB-related vulnerabilities that could be exploited by attackers. As a result, the script generates a detailed report on the port's status, potential vulnerabilities, and suggested remediation actions for each identified host.

Such an automated process not only significantly improves the work of security teams, but also ensures that new threats are quickly identified and responded to, which is crucial in a dynamic cyber environment. Nmap's integration with Python enables you to create a powerful security monitoring tool tailored to each organization's specific needs and challenges.

3. Use Cases

Nmap's integration with Python opens up a broad spectrum of possibilities for cybersecurity professionals, offering an advanced and automated approach to network security management. The practical use of these tools enables the implementation of a number of key tasks, from monitoring and auditing, to

advanced penetration testing and incident response. The following are some examples of practical applications of Nmap integration with Python that illustrate its potential in various cybersecurity scenarios.

Network Monitoring & New Device Discovery

One of the primary uses of Nmap and Python integration is continuous network monitoring to detect newly added devices. Automatic network scanning using Python scripts using Nmap can provide up-to-date knowledge of all devices connected to the network, which is crucial for maintaining security and managing changes in the network environment. Such a system can also automatically notify the security team of unauthorized devices, potentially indicating infiltration attempts or unsecured hardware.

Security Audit & Vulnerability Assessment

With the help of Python scripts, Nmap can be used to conduct regular security audits, identifying open ports, running services, and potential vulnerabilities. The integration of these tools allows you to create detailed reports on the state of network security, pinpointing weaknesses that need attention. Automating this process allows for regular security reviews, minimizing the risk of overlooking critical vulnerabilities.

Penetration Testing Automation

Nmap's integration with Python can also be used to automate penetration testing. By using Python scripts to control Nmap scanning, penetration testers can efficiently map the network, identify potential attack vectors, and automatically test known vulnerabilities. This method enables comprehensive security testing that is both effective and time-efficient.

Incident Response and Threat Intelligence

If a potential security incident is detected, the integration of Nmap and Python can be used to quickly diagnose the situation and take appropriate action. Automated scripts can perform detailed scans to identify the source of the threat, assess the extent of the incident's impact on the network, and speed up the response process. Such a quick and automated response is crucial in minimizing potential damage.

Integration with safety management systems

Nmap's integration with Python allows you to easily connect to other security management systems and analysis tools. Data from scans can be exported to SIEM (Security Information and Event Management) systems,

vulnerability management tools, or security automation platforms, providing a holistic approach to the topic under consideration.

4. Advanced Techniques

Advanced techniques for using Nmap combined with Python allow for the creation of a powerful set of cybersecurity tools that can be used not only for ongoing network monitoring and auditing, but also for deep security analysis and the development of complex defense strategies. By taking advantage of the advanced capabilities of both tools, security professionals are able to more effectively counteract cyber threats, identify weaknesses in the IT infrastructure, and automate incident response processes. The following are some advanced techniques that demonstrate the potential of integrating Nmap with Python.

Automatic vulnerability identification and classification

Using Nmap scripting (NSE – Nmap Scripting Engine) and advanced Python libraries for data analysis, you can automatically identify and classify potential vulnerabilities in the network. NSE scripts provide a rich set of tools for testing various aspects of security, from detecting specific vulnerabilities to testing weak passwords to identifying insecure service configurations. By using Python to process Nmap scan results, specialists can create automated pipelines that not only detect vulnerabilities, but also assign them priorities based on defined criteria, such as the degree of threat, ease of exploitation, or impact on infrastructure.

Intelligent Network Scanning with Machine Learning

Python's integration with Nmap opens the door to the use of machine learning (ML) algorithms to analyze patterns in network traffic and scan results. For example, ML models can be trained on historical scan data to predict potential threats or identify anomalies in network traffic that may indicate unauthorized activity. This intelligent network scanning allows you to proactively detect threats before they can cause damage, as well as optimize scanning processes by focusing on areas with higher risk.

Dynamic Network Topology Mapping

Advanced Python scripts, using the results of Nmap scans, can be used to dynamically map network topologies. This technique allows you to visualize the structure of your network, devices, and connections, which is invaluable for security analysis and infrastructure change planning and expansion. Automatic

network map generation makes it easier to understand the complexity of your IT environment, identify critical points, and plan defensive measures.

Automated Incident Response

Combining Nmap with Python also makes it possible to develop advanced automated incident response systems. For example, Python scripts can listen for alerts generated by Nmap (or other integrated security tools) and automatically run predefined response procedures, such as isolating infected devices, updating settings.

5. Best practices and recommendations

Nmap's integration with Python is a powerful tool in any cybersecurity professional's arsenal, enabling efficient network scanning, vulnerability analysis, task automation, and advanced defense strategies. However, in order to fully exploit the potential of combining these technologies, it is worth following some best practices and recommendations. The following are some key tips to help you use Nmap and Python effectively and securely in your cybersecurity practice.

Understanding the tool and using it responsibly

Before we start using Nmap and Python to scan networks, it is essential to have a deep understanding[4] of the features and capabilities of these tools. Be sure to carefully review the Nmap documentation and Python educational resources to make the most of their potential while avoiding unwanted effects such as network or device disruption.

Scanning networks, especially those that are not part of our organization, may be illegal or unethical without the explicit permission of the owner. You should always make sure that you have the proper authority to carry out scans to avoid legal and ethical consequences.

Accurate Scan Planning and Configuration

Performing network scans, especially in large and complex environments, requires careful planning. It is necessary to determine which scanning targets are the most important, whether we focus on specific ports, devices, or look for

[4] K. Henry, M. Simon, "CompTIA Security+ Certification Guide: Master IT security essentials and exam topics for CompTIA Security+ SY0-501 certification", Packt Publishing, 2018; J. Long, "Google Hacking for Penetration Testers, Volume 1", Syngress, 2005.

specific vulnerabilities. By carefully planning and configuring your Python scripts, you can minimize the load on your network and focus on the most important aspects from a security perspective.

Automation with care

Automating scanning and analysis processes is one of the main advantages of integrating Nmap with Python. However, over-automation without proper supervision can lead to important information being overlooked or false positives being generated. It's important to strike a balance between automation and manual analysis of the results, ensuring that each find is thoroughly vetted by a specialist.

Continuous education and update

The world of cybersecurity is fast-paced, and attack methods and defense tools are constantly evolving. Therefore, it is crucial to continuously improve your skills and knowledge of the new capabilities of Nmap, Python, and general trends in cybersecurity. Regular software and script updates are essential to ensure protection against the latest threats and take advantage of the latest defense techniques.

Taking care of Data Privacy and Security

In the process of scanning and analysing network data, particularly sensitive information can be discovered or collected. It's important to follow data protection best practices, such as encrypting data, restricting access to scan results, and applying data minimization policies.

Summary

The integration of Nmap with Python opens up new opportunities for cybersecurity professionals to effectively manage and secure their network infrastructure. Nmap, being one of the most advanced network scanning tools, offers not only the ability to detect devices and services running on a given network, but also allows you to identify open ports and potential vulnerabilities. Its flexibility and depth of analysis make it an invaluable tool in the arsenal of anyone involved in digital security. On the other hand, Python – with its simplicity of syntax, powerful libraries and versatility – is an ideal language for automating tasks, analyzing data, and creating complex cybersecurity solutions.

The combination of these two tools allows you to create automated scanning processes that can run continuously or run on a scheduled schedule, providing an up-to-date view of your network security status. This automation is crucial,

especially in large organizations where it is virtually impossible to manually monitor all aspects of the network. Examples of use include real-time monitoring of network status for new or unauthorized devices, identifying changes in service configuration that could introduce new vulnerabilities, and conducting regular security audits.

Advanced techniques such as dynamic network topology mapping, intelligent scanning using machine learning, and automated incident response open up new possibilities for protecting against cyber threats. By analyzing network traffic patterns and detecting anomalies, these systems can pinpoint attack attempts in near real-time, enabling rapid response and minimizing potential damage.

However, the use of such powerful tools requires not only technical knowledge, but also responsibility and adherence to best practices. Responsible scanning, taking into account ethical and legal principles, is the foundation for the safe and effective use of Nmap and Python. It is also important to keep in mind the protection of data privacy and security when working with scan results, as well as the need for continuous education and updating of knowledge in order to be able to effectively counteract dynamically changing threats in cyberspace.

Nmap's integration with Python is a key tool in any cybersecurity professional's arsenal, enabling deep analysis and understanding of network infrastructure and ensuring that it is protected from ever-evolving threats. The use of these tools allows you to build advanced monitoring, analysis and response systems, which are necessary to ensure a high level of cybersecurity in any organization.

Literature

Althoff C., "The Self-Taught Programmer: The Definitive Guide to Programming Professionally", Independently published, 2017.

Beale J., Baker A., Caswell B., "Snort: IDS and IPS Toolkit", Syngress, 2007.

Chirillo J., Blaul S., "Hack Attacks Revealed: A Complete Reference with Custom Security Hacking Toolkit", Wiley, 2002.

Erickson J., "Hacking: The Art of Exploitation, 2nd Edition", No Starch Press, 2008.

Fyodor, "Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning", Insecure.Com LLC, 2009.

Greenblatt S., "Cybersecurity: The Beginner's Guide: A comprehensive guide to getting started in cybersecurity", Packt Publishing, 2019.

Henry K., Simon M., "CompTIA Security+ Certification Guide: Master IT security essentials and exam topics for CompTIA Security+ SY0-501 certification", Packt Publishing, 2018.

Kennedy D., O'Gorman J., Kearns D., Aharoni M., "Metasploit: The Penetration Tester's Guide", No Starch Press, 2011.

Long J., "Google Hacking for Penetration Testers, Volume 1", Syngress, 2005.