## Lab 24-01: Automating Pentesting With Python

### Scenario

A cybersecurity team at a tech firm leverages Python to automate penetration testing tasks in Kali Linux. They develop Python scripts to streamline vulnerability scanning, exploit execution, and reconnaissance processes. By automating these tasks, the team accelerates their testing workflow, identifies vulnerabilities more efficiently, and strengthens the security posture of their systems. Additionally, they schedule regular scans and tests using cron jobs to ensure continuous monitoring and proactive threat mitigation. Through automation, the team optimizes its resources and enhances its ability to detect and respond to potential security threats effectively.

### Solution

To automate penetration testing with Python in Kali Linux, the cybersecurity team develops scripts for vulnerability scanning, exploit execution, and surveillance. They streamline tasks using Python libraries such as requests, socket, and nmap. Scheduling regular scans with cron jobs ensures continuous monitoring. Additionally, the team maintains and updates its scripts to adapt to evolving threats and improve efficiency over time. Through automation, they optimize their testing workflow and enhance their ability to detect and mitigate security vulnerabilities effectively.

|  |
| --- |
| 1. Open your VScode in Kali Linux and write the attached code. The code is briefly described here.   Import Statements: The script imports necessary modules such as socket, sys, threading, time, and nmap.  Usage Information: It provides usage information if the command-line arguments are not provided correctly.  Command-line Arguments: The script expects three command-line arguments: target IP address or hostname, initial port number, and final port number to scan.  Scanning Target: It attempts to resolve the target IP address using a socket.gethostbyname() function. If the resolution fails, it prints an error message and exits.  Port Scanning Function: The scan\_port() function is defined to scan individual ports. It creates a socket and attempts to connect to the specified port on the target. If the connection is successful (connection is not non-zero), it prints that the port is open and tries to get the associated service name using the nmap library. Then, it calls the exploit\_port() function.  Getting Service Name: The get\_service\_name() function utilizes the nmap library to get the service name associated with a port. It scans the target IP address for the specified port and retrieves the service name.  Exploiting Port: The exploit\_port() function is a hypothetical function that simulates exploiting the open port. It is a placeholder for your actual exploit code.  Main Scanning Logic: The script iterates over the range of ports to scan using a loop. For each port, it creates a separate thread (to perform scanning concurrently) and starts the thread. It limits the number of concurrent threads to avoid overwhelming the system. After starting threads for each batch of ports, all threads are waited for to finish using the join() method.  Timing: The script calculates the time taken for port scanning and prints it at the end.         1. Save this code anywhere in Kali Linux with any name.      1. Now, select the folder by using the **cd** command.      1. Now, select the python file folder.      1. Use the command python3 filename.py TARGET IP/URL START\_PORT END\_PORT.      1. Now, the scanning will start.       7. The above scanning is done by **nmap**. |