Contents

[Introduction 1](#_Toc188618171)

[Features 1](#_Toc188618172)

[General Features 1](#_Toc188618173)

[Module-Specific Features 1](#_Toc188618174)

[Async Port Scanner 1](#_Toc188618175)

[SSH Brute-Forcer 1](#_Toc188618176)

[Modules Overview 2](#_Toc188618177)

[Async Port Scanner 2](#_Toc188618178)

[SSH Brute-Forcer 2](#_Toc188618179)

[How to Execute 2](#_Toc188618180)

[Output Examples 3](#_Toc188618181)

[SSH Brute-Forcer 3](#_Toc188618182)

[Log File Examples 4](#_Toc188618183)

[Requirements Documentation 4](#_Toc188618184)

[Functional Requirements 4](#_Toc188618185)

[Non-Functional Requirements 4](#_Toc188618186)

# Introduction

The **Penetration Testing Toolkit** is a modular Python-based tool designed for network and system security assessment. It includes:

1. An **Async Port Scanner** for identifying open ports, services, and banners.
2. An **SSH Brute-Forcer** for testing password strength on SSH services.

This toolkit is efficient, user-friendly, and includes detailed logging and reporting capabilities.

# Features

## General Features

* **Modular Design**: Independent modules that work together seamlessly.
* **Detailed Reporting**: Outputs results to CSV files for analysis.
* **User-Friendly**: CLI interface and GUI options.

## Module-Specific Features

### Async Port Scanner

* **Asynchronous Scanning**: Uses asyncio for high-performance concurrent scanning.
* **Service Identification**: Detects services using socket.getservbyport.
* **Banner Grabbing**: Fetches banners from open ports.
* **IP Range Support**: Scans individual IPs or ranges.
* **GUI**: Includes a tkinter-based interface.

### SSH Brute-Forcer

* **Asynchronous Brute-Forcing**: Tests multiple passwords concurrently.
* **Password List Support**: Customizable input of password wordlists.
* **Service-Specific Targeting**: Focused on SSH (port 22).

## Modules Overview

### Async Port Scanner

**Objective**: Efficiently scan open ports, detect running services, and grab banners.

**Key Methods**:

* scan\_port(port): Scans a single port and fetches its banner.
* run\_scan(): Scans all ports in the specified range concurrently.
* save\_results(filename): Saves results to a CSV file.
* print\_results(): Displays results in the terminal.

### SSH Brute-Forcer

**Objective**: Test the strength of SSH credentials using a provided wordlist.

**Key Methods**:

* try\_password(password): Attempts SSH login with a given password.
* run\_bruteforce(): Concurrently tests multiple passwords.

### How to Execute

**Prerequisites**

1. Install Python 3.8 or higher.
2. Install required libraries:
3. pip install asyncio paramiko ipaddress

**Steps**

1. Clone the repository:
2. git clone https://github.com/username/penetration-toolkit.git
3. cd penetration-toolkit
4. Run the main toolkit script:
5. python toolkit.py
6. Choose an option from the menu:
   * **1**: Async Port Scanner.
   * **2**: SSH Brute-Forcer.
   * **3**: Exit.

### Output Examples

**Async Port Scanner**

**Input**

* **Target**: 192.168.1.1
* **Port Range**: 1-1000
* **Timeout**: 2 seconds

**Output (Terminal)**

Open Ports:

Port 22 (ssh): OpenSSH 8.2p1 Ubuntu

Port 80 (http): Apache HTTP Server

**Output (CSV)**

Port,Service,Banner

22,ssh,OpenSSH 8.2p1 Ubuntu

80,http,Apache HTTP Server

### SSH Brute-Forcer

**Input**

* **Target**: 192.168.1.1
* **Username**: admin
* **Password List**: passwords.txt

**Output**

[FAILED] Incorrect password: 12345

[FAILED] Incorrect password: admin123

[SUCCESS] Password found: letmein

### Log File Examples

**Port Scanner (port\_scanner.log)**

2025-01-22 12:34:56 - INFO - Open port found: 22 (ssh) - OpenSSH 8.2p1 Ubuntu

2025-01-22 12:34:56 - WARNING - Failed to connect to port 23

2025-01-22 12:34:56 - INFO - Results saved to scan\_results\_192.168.1.1.csv

## Requirements Documentation

### Functional Requirements

1. **Port Scanning**: Identify open ports, services, and banners.
2. **SSH Brute-Forcing**: Test SSH credentials using a custom wordlist.
3. **Asynchronous Execution**: Handle concurrent tasks efficiently.
4. **Output Formats**: Save results in a readable CSV format.

### Non-Functional Requirements

1. **Performance**: Complete scans within reasonable time frames.
2. **Scalability**: Support large IP ranges and port ranges.
3. **Usability**: Provide a clear CLI and optional GUI for ease of use.
4. **Portability**: Run on major platforms (Windows, Linux).