Title: Cybersecurity Final Project

Estimated Time: 4 hours

Level of Exercise-Beginner

Pre-requisite: -

- Basic TCP/IP Knowledge
- Basic information security knowledge
- Basic knowledge about packet captures
- Basic operating system knowledge (Windows, Linux)
- PC or Laptop
- Chrome/Firefox Browser
- Must have pre-installed Wireshark tool
- Basic knowledge of password cracking techniques and network scanning.
- Basic understanding of Linux command-line interface
- Familiarity with Kali Linux operating system
- Pre- Installed Hashcat tool in Kali Linux
- Pre- Installed John the ripper tool in Kali Linux
- SQL injection
- Pre-installed Zap
- Basic knowledge of ZAP
- Firefox browser.
- Pre-installed Java Runtime Environment

Note: This project is designed for laptops or desktop computers with a reliable Internet connection only and not on mobile devices.

Objectives:

- Solve real world problems based on different scenarios using Wireshark.
- Understand the basics of password cracking techniques and network scanning.
- Learn to use Hashcat and John the Ripper in Kali Linux for password cracking.
- Learn to interpret and analyze scan results and cracked passwords.
- Identify vulnerable websites by performing SQL injection attack.
- Identify vulnerabilities of a website using ZAP

About this lab:

This capstone project aims to assess your proficiency in utilizing popular cybersecurity analysis tools, including Wireshark, Kali Linux, and ZAP. The project outline consists of several tasks. Firstly, you will employ Wireshark to analyze network traffic and address real-world problems. Next, you will leverage Kali Linux's capabilities, utilizing tools such as Hashcat and John the riper for password cracking. In the next task you will check the vulnerability of a website by manually performing SQL injection attacks. Finally, you will use ZAP to test run on a website and analyze security vulnerabilities of the websites.

Project Tasks: -

Solve real world problems based on different scenarios using Wireshark

Scenario 1: Network Traffic Analysis

A Company wants to monitor traffic originating from or destined to their Router IP Address '192.168.60.1'. After observing the captured packets, the network administrator wants to further drill-down the captures and identify traffic between IP addresses '192.168.60.1' and '192.168.121.128'. Use Wireshark to help identify all traffic to/from gateway and record your observations.

IMP NOTE- Download and open this sample pcap file **TestRun5 - Internal side of Router** from the repository for analysis. [ref: Mixed PCAP file repo with a great deal of BACnet traffic (by Steve Karg)
https://kargs.net/captures/TestRun5%20-%20Internal%20side%20of%20Router.pcap]

Task 1.1: Filter for packets originating from IP address –'192.168.60.1'. Add a quick filter button as 'Filter1' to easily apply or remove this filter from the overall capture.
Take a screenshot of captured packets on Wireshark originating from IP address – '192.168.60.1'.

Task 1.2: Filter for packets reaching IP address-'192.168.60.1'. Provide display filter button as 'Filter2' to easily apply or remove this filter from the overall capture. Take a screenshot of captured packets on Wireshark reaching IP address-'192.168.60.1'.

Task 1.3: Filter for packets that are either originating from or destined to IP address- '192.168.60.1'. Provide display filter button as 'Filter3' to easily apply or remove this filter from the overall capture. Take a screenshot of captured packets on Wireshark where the packets are either originating from or destined to IP address-'192.168.60.1'

Task 1.4: Filter for packets between IP addresses '192.168.60.1' and '192.168.21.100'. Identify and record your observation on the captured date and time, protocol type, number of

packets transmitted/received. Color this filtered traffic in Light Green to visually differentiate the traffic. Take screenshot of captured packets between IP addresses '192.168.60.1' and '192.168.21.100'.

Scenario 2- VoIP Security Analysis

Your organization has implemented the necessary infrastructure for VoIP calling. From past few weeks, employees are complaining of frequent call drops on VoIP network. As a network administrator of this company, you are entrusted with the task of finding out the root cause of this issue. Use Wireshark to help your company identify root cause and record your observations.

Note: Download this sample pcap file **VoIP Calls FINAL.pcapng** (a simple capture file containing SIP call with RTP in G711 codec) from the repository [ref: http://startrinity.com/VoIP/VoipTroubleshootingBook/VoipTroubleshootingBook.aspx].

Task 2.1: List a few commonly faced problems in a VoIP network and identify the issue pointed out in this scenario. What do you think is the cause of this issue?

Task 2.2: How many SIP endpoints do you see in the captured file? How many VoIP calls do you see in the captured files? Take a screenshot to illustrate this.

Task 2.3: Play the audio streams of each call and record your observations – SIP URL, duration of calls and whether audio player is indicating any discrepancies?

Task 2.4: What does a typical SIP Call Flow look like? Take a screenshot of the SIP Call flow on Wireshark tool.

Password Cracking using tools in Kali Linux

Scenario 3: Password Cracking using HashCat and John the ripper tools

XYZ company wants to ensure that there is no unauthorized access to their website as some of the data which they have included is confidential. They have recruited you as a cybersecurity analyst and have shared hashes of some of the passwords with you. You need to apply your skills in cracking the passwords from the given files and analyze the strength and weakness of the files.

Task 3.1: Get the hash files

Here we have two hash files simple_hash.txt and complex_hash.txt which consists of password hashes.

1. Download the hash file simple_hash.txt from here. Take the screenshot of the bash command used to download the file.
2. Display the contents of the file **simple_hash.txt** using suitable bash command. Take the screenshot of the command used.
3. Download the hash file **complex_hash.txt** from here. Take the screenshot of the bash command used to download the file.
4. Display the contents of the file **complex_hash.txt** using suitable bash command. Take the screenshot of the command used.
5. Identify which hash algorithm is used for encryption.

Task 3.2 Cracking passwords using a wordlist and Hashcat's built-in rules.

1. Locate the built-in **rockyou.tx**t wordlist file using a suitable command. Take a screenshot of the command used.
2. Decrypt the hashes in the files **simple_hash.txt and Complex_hash.txt** using suitable hashcat commands. Take screenshots of the commands used.

Task 3.3 Analyzing cracked passwords and their implications.

1. Document the cracked passwords and their corresponding plaintext results.
2. Analyze the strength and weaknesses of the cracked passwords based on the complexity of both the hash files.
3. Take a screenshot of the outcomes of the decryption.

Task 3.4: Get the hash files

1. Download the hash file **hash1.txt** from here. Take the screenshot of the bash command used to download the file.
2. Display the contents of the file **hash1.txt** using suitable bash command. Take the screenshot of the command used.
3. Download the hash file **hash2.txt** from here. Take the screenshot of the bash command used to download the file.
4. Display the contents of file **hash2.txt** using suitable bash command. Take the screenshot of the command used.
5. Identify which hash algorithms are used for encryption for both the files.

Task 3.5: Using different modes and techniques in John the Ripper

1. Perform a basic dictionary attack using a built-in wordlist password in john and run the suitable command for **hash1.txt** with suitable hash format. Take a screenshot of the command used.
2. Perform an incremental mode attack, run the suitable command for **hash2.txt** with suitable hash format. Take a screenshot of the command used.

Task 3.6: Analyzing cracked passwords and assessing their strength.

1. Document the cracked passwords and their corresponding plaintext results.
2. Analyze the strength and weaknesses of the cracked passwords based on the complexity of both the hash files.
3. Take a screenshot of the outcomes of the decryption.

SQL Injection Attack on a test website

Scenario 4: Manual SQL injection

Meet Sarah, a cybersecurity analyst working for a reputable financial institution that provides online banking services. Sarah's role is crucial in safeguarding the bank's digital infrastructure and protecting sensitive customer information from cyber threats.

One morning, as Sarah settles into her office, she notices an alert from the bank's security monitoring system. The alert indicates a potential cyber-attack targeting the online banking platform. Sarah's training and experience kick in, and she quickly springs into action.

Company name: - Altoro Mutual

The company has its own website, https://demo.testfire.net/, where attackers are trying to attack and hack the data.

Task-4.1: Explore the Website with Admin Credentials

Objective: Log in to the website using the provided admin credentials.

1. Open the website in a web browser.
2. Locate the login section on the homepage.
3. Enter the following credentials:
    - *Username: admin*
    - *Password: admin*
4. Click on the "Login" button.
5. Take a screenshot of the successfully logged-in status.

Task-4.2: Transfer 1000 Euros from Corporate Account to Checkings

Objective: Perform a transfer of 1000 Euros from the corporate account (800000) to the checking account (800000).

1. Access the account management section of the website.
2. Locate the transfer funds feature.
3. Enter the required details:
    - *Source Account: 800000 (corporate account)*
    - *Destination Account: 800000 (checking account)*

- *Amount: 1000 Euros*

4. Initiate the transfer.
5. Take a screenshot of the transfer status, confirming the successful transfer.

Task-4.3: Identify and Discuss SQL Injection Vulnerabilities

Objective: Identify potential security risks associated with the website's login page and discuss SQL injection vulnerabilities.

1. Discuss the potential security risks associated with the website, focusing on SQL injection vulnerabilities.
    a. List some potential security risks associated with SQL injection vulnerabilities in a website
    b. Suggest a solution to mitigate the risks associated with SQL injection vulnerabilities
2. Provide an example username and password that could exploit the identified vulnerabilities on the login page of the demo website. What is the equivalent SQL query with these inputs and draw inferences from this query?
3. Login to the demo website with these credentials and check whether you can access the website. Take a screenshot of the same.

Identify vulnerabilities of a website using ZAP

Scenario 5: Running pen-tests on a website using ZAP

Jane is a cybersecurity analyst working for an e-commerce portal. Her role includes running pen tests and scans on the site and identifying any vulnerabilities which can be used by attackers. She needs to simulate techniques used by attackers with the goal of breaking into the system.

As these techniques simulate actual attacks on the system, before running the tests on the live company site, Jane wants to test-run ZAP on http://itsecgames.com. Commonly known as bWAPP, or a buggy web application, this is a free and open-source web application which is deliberately made insecure.

Task 5.1: Running an Automated Scan with traditional spider

1: Briefly explain the concept of Active scanning and document the steps used to perform automated scanning with traditional spider in ZAP.

2: Run an automated scan on the web application http://itsecgames.com. and take a screenshot showing the results of the scan.

Task 5.2: Running a scan with Ajax Spider

1: Document the steps used to perform automated scanning with Ajax spider.

2: Execute an automated scan using the Ajax Spider and take a screenshot showing the results.

Task 5.3: See Explored Pages

1: Display the tree view of the explored pages after running an automated scan on the web application with traditional spider. Take a screenshot of the explored pages.

2: Display the tree view of the explored pages after running an automated scan on the web application with ajax spider. Take a screenshot of the explored pages