A logo for a software company

Description automatically generated

***Hijacking using Kali Linux***

***Introduction***

Session hijacking is a critical attack method in cybersecurity that allows an attacker to intercept and take over a user’s session by capturing session cookies or tokens. This technique leverages the vulnerabilities in unsecured sessions, where attackers can exploit weak or non-encrypted communication channels. This report outlines the methodology for performing session hijacking using tools available in Kali Linux and demonstrates how attackers can intercept and hijack a user’s session on a network. The report also includes steps for observing, detecting, and mitigating this type of attack.

***Purpose***

The purpose of this lab is to demonstrate how session hijacking can be performed to highlight the vulnerabilities in unsecured network environments. Additionally, the lab aims to provide students with insights into detection and defense techniques against session hijacking, fostering a better understanding of proactive security measures.

***Scope***

*The lab consists of three main sections:*

***Session Hijacking Setup****: Configuring the environment to intercept and manipulate session traffic.*

***Session Hijacking with Ettercap and Wireshark****: Performing an active session hijacking attack using ARP spoofing to capture and inject session cookies.*

***Detecting and Mitigating Session Hijacking:*** *Using monitoring tools to detect session hijacking attempts and discussing best practices for securing network sessions.*

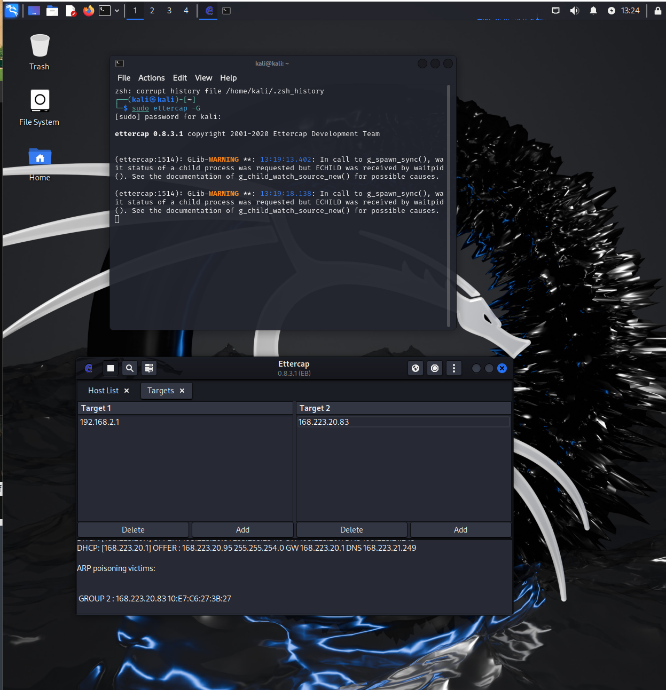
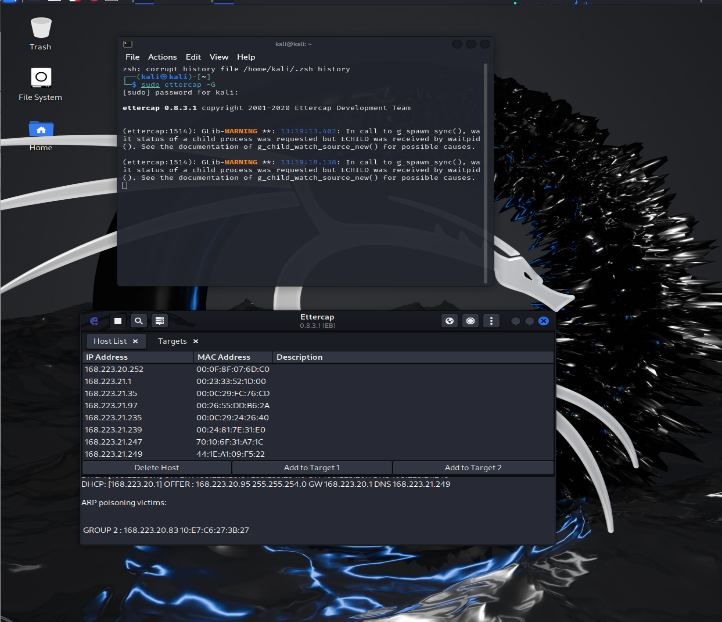
***Overview***

This lab explores the vulnerabilities in unsecured sessions, specifically the risks posed by ARP spoofing and session hijacking. In Part 1, the lab setup is established to allow for interception of traffic within a local network. Part 2 focuses on using Ettercap and Wireshark to capture session data, allowing attackers to take over an existing session. Part 3 presents methods to detect and mitigate these attacks, emphasizing the importance of encryption and network monitoring.

**Results**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

1. **What vulnerabilities in the session management process made this attack possible?**

Vulnerabilities in session management that enabled this attack include transmitting session cookies over HTTP, which leaves them unencrypted and susceptible to interception. Additionally, weak or predictable session IDs allow attackers to brute-force or guess them, increasing the likelihood of a successful hijacking. A lack of proper session expiration also prolongs the attack window, as sessions remain open longer, while not regenerating session IDs upon login or critical actions enables session fixation attacks, where attackers force a user to log in using a session ID controlled by the attacker.

1. **How would the use of HTTPS prevent session hijacking in this scenario?**

The use of HTTPS prevents session hijacking by encrypting all data exchanged between the client and server, including session cookies, which makes them unreadable to attackers intercepting network traffic. HTTPS also ensures server authenticity through certificates, helping protect against man-in-the-middle (MITM) attacks. By enforcing secure and HttpOnly attributes for cookies, HTTPS further restricts cookie access, limiting transmission to secure connections and preventing JavaScript access.

1. **What other defensive mechanisms could be implemented to prevent session hijacking attacks?**

Additional defenses against session hijacking include implementing session timeouts and forcing re-authentication after periods of inactivity to limit the hijack window. Regenerating session IDs upon login and after significant actions reduces fixation risks, while two-factor authentication adds an extra security layer, making unauthorized access more difficult. Verifying IP addresses and user agents per session strengthens user validation, and setting cookies with the SameSite attribute prevents them from being sent with cross-site requests, reducing the risk of cross-site request forgery (CSRF).

1. **How does ARP poisoning work, and why is it effective in man-in-the-middle attacks?**

ARP poisoning works by sending falsified ARP messages on a network, associating the attacker’s MAC address with the IP address of a legitimate host, like a router. As a result, other devices on the network unknowingly send their data through the attacker, who can intercept, alter, or redirect this traffic. It’s effective in man-in-the-middle attacks because it allows the attacker to gain access to sensitive information, such as session cookies and credentials, which can be used for further attacks like session hijacking. ARP’s lack of authentication or security controls makes it particularly vulnerable to this form of manipulation.