Metasploit Bonus
Date: 04/29/2024

For the bonus assignment, I'll create a new Metasploit exercise focusing on a Server-Side Request Forgery (SSRF) attack using the Metasploit framework. This exercise will help you understand how SSRF attacks can be utilized to manipulate server-side applications into executing unintended actions, such as accessing internal services. Here's a step-by-step guide

Metasploit Exercise: Server-Side Request Forgery (SSRF) Attack

Objective: Demonstrate the exploitation of SSRF vulnerabilities using Metasploit to show how attackers can force a server to make requests to unintended locations, potentially accessing sensitive information or interacting with internal systems.

Tools and Setup Required:

- Kali Linux with Metasploit installed.
- Vulnerable application server (victim) simulating a web service with SSRF vulnerability.
- Internal application accessible only from the victim server.

Step-by-Step Instructions:

1. **Environment Setup:**
   - Configure the vulnerable application on a server, ensuring it can make outbound HTTP requests.
   - Set up an internal web server that will be the target of the SSRF attack. This server should host sensitive data or services not directly accessible from the public internet.

2. **Identify SSRF Vulnerability:**

- Analyze the application to find endpoints that take URLs as input and fetch data from them. These are potential SSRF points.

3. **Exploit Setup:**

- On your Kali system, start Metasploit:

```
msfconsole
```

- Use an appropriate SSRF module or script. For this exercise, let's simulate an SSRF module:

```
msf > use auxiliary/scanner/http/ssrf_inject
msf > set RHOSTS victim_server_ip
msf > set RPORT 80
msf > set TARGETURI /vulnerable_endpoint
msf > set SSRF_URI http://internal_server_ip/secret_data
msf > set VERBOSE true
```

4. **Execute the Attack:**

- Run the exploit to make the vulnerable server issue a request to the internal server:

```
msf > run
```

- Monitor the output. If successful, the server's response should include data from the internal service, demonstrating the SSRF vulnerability.

5. **Post-Exploitation Analysis:**

- Analyze the data retrieved from the internal server.
- Discuss potential impacts, such as unauthorized access to sensitive information or interaction with internal APIs.

6. **Cleanup and Reflection:**
- Remove any temporary configurations used during the exercise.
- Reflect on how to mitigate such vulnerabilities, focusing on input validation and restricting outgoing requests from servers.

Explanation:

During this exercise, you manipulated a server into making an HTTP request to a restricted internal service. This illustrates how SSRF can be used to bypass firewalls and access controls, allowing attackers to interact with internal infrastructure that should not be exposed to the internet.

To Write a Technical Report:

- Write up a detailed report of the exercise, including your setup, steps taken, screenshots of the exploit in action, and your analysis of the outcome.
- Include your reflections on potential security measures to prevent SSRF vulnerabilities.

This exercise offers practical experience with SSRF attacks and underscores the importance of secure coding practices and server configurations to prevent such vulnerabilities.