|  |  |
| --- | --- |
| **Lab 16** | |
| **Name** | Dang Hoang Nguyen |
| **Student ID** | SE171946 |

What is Server-Side Request Forgery (SSRF), and how does it differ from other types of web application vulnerabilities?

A online security flaw known as Server-Side Request Forgery (SSRF) enables attackers to trick a server into sending unwanted requests to other systems. It is distinct from other kinds of vulnerabilities in web applications in a few important ways:

Goal:

* SSRF: Targets the server directly by abusing server-side capability.
* Others: Frequently focus on application logic, session data, or user data.

Effect:

* SSRF: May result in data exfiltration, denial-of-service attacks, remote code execution, and exposing of internal resources.
* Others: Could result in unapproved access, account takeover, data breaches, or website vandalism.

Challenge:

* SSRF: Compared to certain other vulnerabilities, it may be more difficult to exploit and identify.
* Others: Depending on the particular vulnerability and application scenario, difficulty varies.

Describe a methodology for exploiting an SSRF vulnerability in a web application. What are the key steps and techniques involved in crafting and sending malicious requests to exploit SSRF vulnerabilities?

1. **Identify SSRF Vulnerability**: Find endpoints vulnerable to SSRF.
2. **Understand Functionality**: Know how the app processes user-supplied URLs.
3. **Craft Malicious Requests**: Manipulate URLs, abuse protocols, exploit DNS rebinding, or request smuggling.
4. **Test Payloads**: Verify crafted payloads work.
5. **Exploit Internal Services**: Access sensitive files, query metadata, or exploit intra-app communication.
6. **Exploit External Resources**: Perform port scanning, access administrative panels, or attack other targets.
7. **Cover Tracks**: Modify headers, use proxies, or delete logs.
8. **Report and Mitigation**: Report responsibly, provide details, and recommend mitigations.
9. **Continuously Monitor**: Regularly scan for SSRF vulnerabilities.
10. **Stay Updated**: Keep up with new SSRF exploitation techniques.

**Lab: Blind SSRF with out-of-band detection**

Every time a user hits a product page, this lab program uses another piece of software to automatically get the URL from the Referer header field. Thus, by entering the Referer URL field that we control, we may OOB SSRF. Put Burp Collaborator to use.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Because this time we are dealing with Blind SSRF we should setup Burp Collaborator to receive DNS response

We have successfully performed Blind SSRF attack(POC only) with the help of Burp Collaborator and we have received a DNS/HTTP Responses

A screenshot of a computer

Description automatically generated

A close up of a line

Description automatically generated