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:

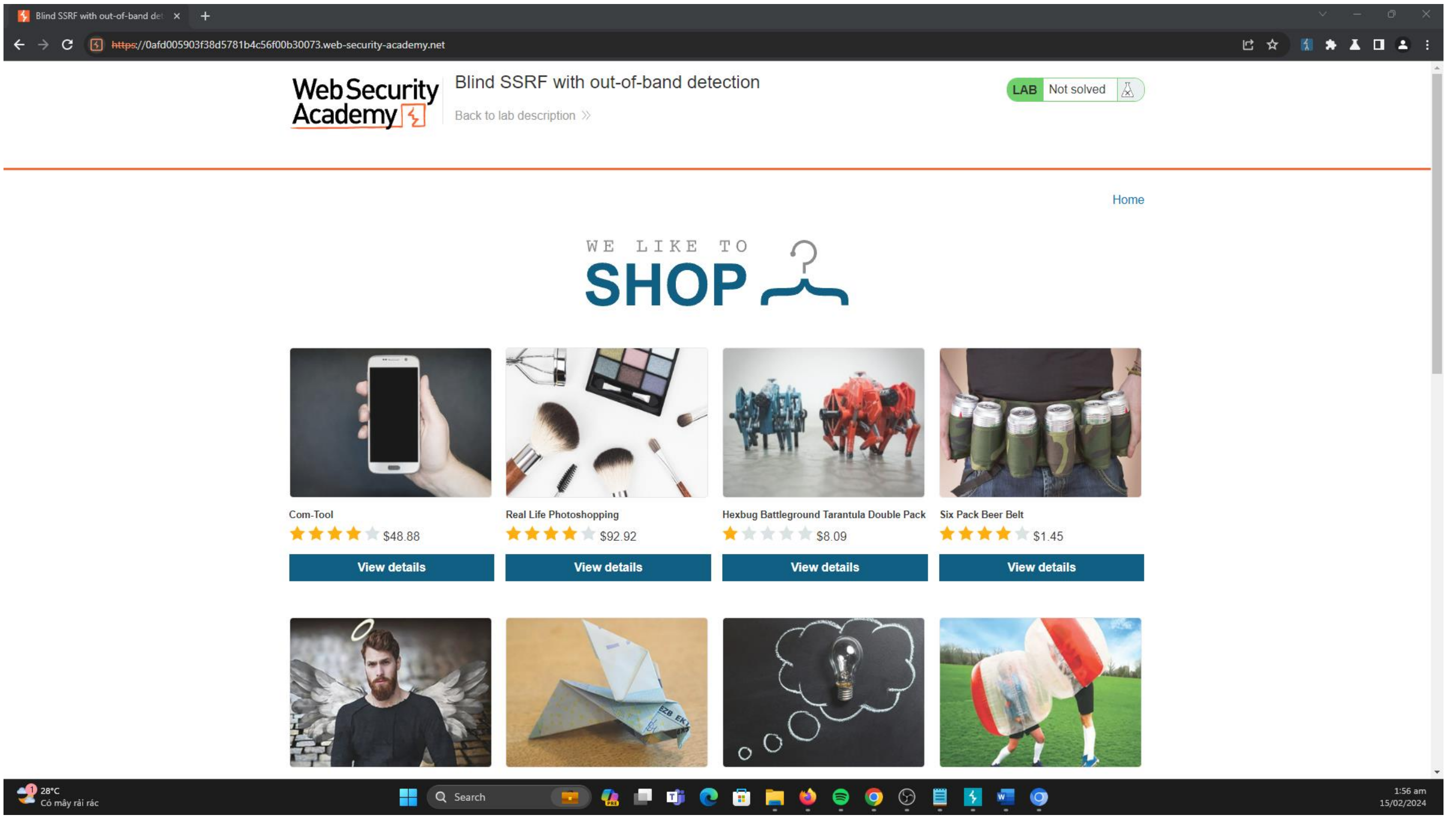
- SSRF: Compared to certain other vulnerabilities, it may be more difficult to exploit and identify.
- Others: Depending on the particular vulnerability and application scenario, difficulty varies.

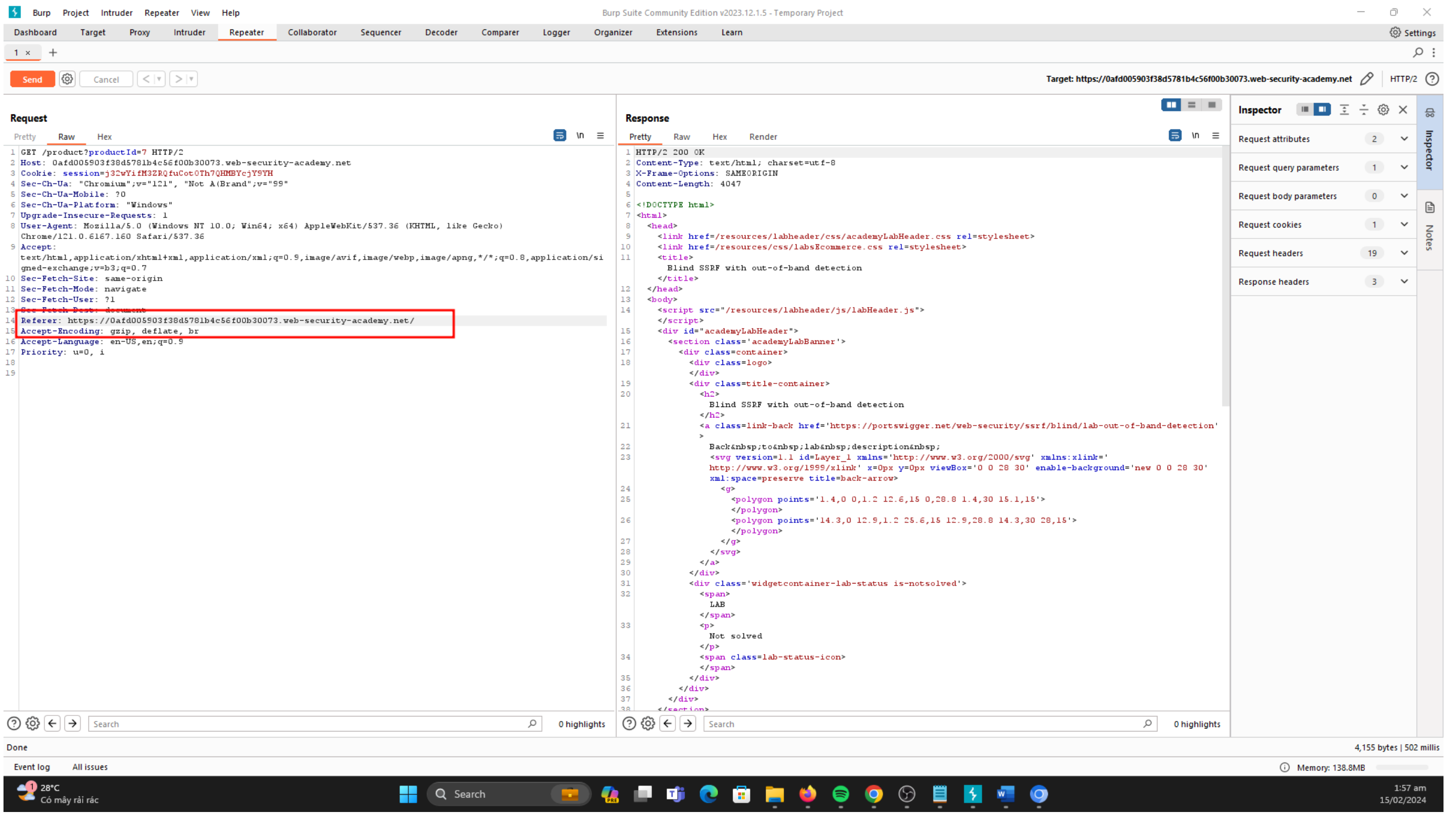
Describe a methodology for exploiting an SSRF vulnerability in a web application. What are the key steps and techniques involved in crafting and sending malicious requests to exploit SSRF vulnerabilities?

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.





Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2023.12.1.5 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Settings
Send
Cancel
Target: https://0afd005903f38d5781b4c56f00b30073.web-security-academy.net
HTTP/2
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
Inspector
Request attributes 2
Request query parameters 1
Request body parameters 0
Request cookies 1
Request headers 19
Response headers 3
Notes
Search
0 highlights
Done
4,155 bytes | 502 millis
Event log
All issues
Memory: 138.8MB
28°C
Có mây rải rác
1:57 am
15/02/2024

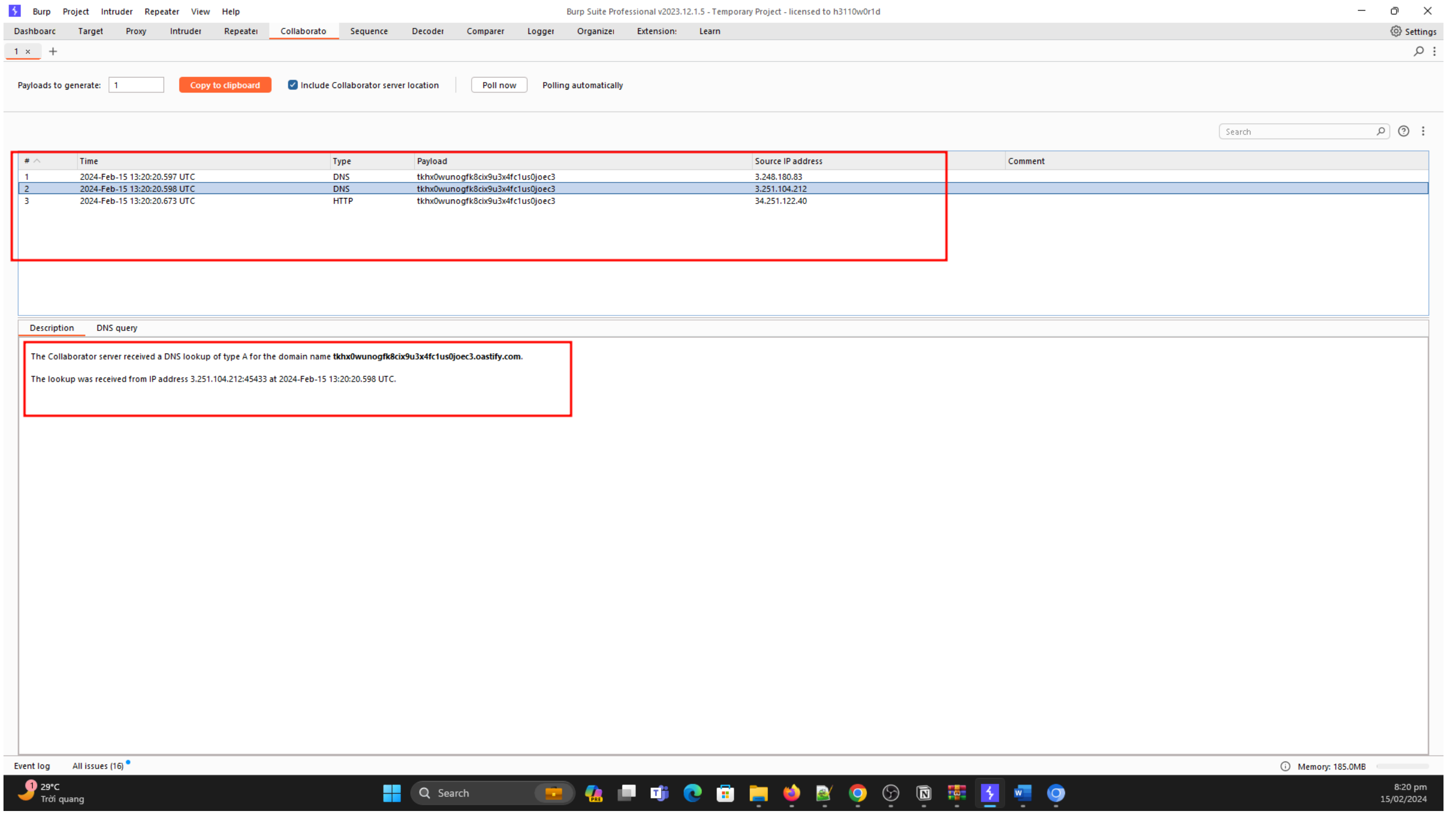
```
GET /product?productId=7 HTTP/2
Host: 0afd005903f38d5781b4c56f00b30073.web-security-academy.net
Cookie: session=j32wYifM3ZRQfuCot0Th7QHMBYcjY9YH
Sec-Ch-Ua: "Chromium";v="121", "Not A(Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/121.0.6167.160 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/si
gned-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://0afd005903f38d5781b4c56f00b30073.web-security-academy.net/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i
```

```
HTTP/2 200 OK
Content-Type: text/html; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 4047

<!DOCTYPE html>
<html>
  <head>
    <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet>
    <link href=/resources/css/labsEcommerce.css rel=stylesheet>
    <title>
      Blind SSRF with out-of-band detection
    </title>
  </head>
  <body>
    <script src="/resources/labheader/js/labHeader.js">
    </script>
    <div id="academyLabHeader">
      <section class='academyLabBanner'>
        <div class=container>
          <div class=logo>
          </div>
          <div class=title-container>
            <h2>
              Blind SSRF with out-of-band detection
            </h2>
            <a class=link-back href='https://portswigger.net/web-security/ssrf/blind/lab-out-of-band-detection'
            >
              Back&nbsp;to&nbsp;lab&nbsp;description&nbsp;
              <svg version=1.1 id=Layer_1 xmlns='http://www.w3.org/2000/svg' xmlns:xlink='
              http://www.w3.org/1999/xlink' x=0px y=0px viewBox='0 0 28 30' enable-background='new 0 0 28 30'
              xml:space=preserve title=back-arrow>
                <g>
                  <polygon points='1.4,0 0,1.2 12.6,15 0,28.8 1.4,30 15.1,15'>
                  </polygon>
                  <polygon points='14.3,0 12.9,1.2 25.6,15 12.9,28.8 14.3,30 28,15'>
                  </polygon>
                </g>
              </svg>
            </a>
          </div>
          <div class='widgetcontainer-lab-status is-notsolved'>
            <span>
              LAB
            </span>
            <p>
              Not solved
            </p>
            <span class=lab-status-icon>
            </span>
          </div>
        </div>
      </div>
    </section>
```

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



Payloads to generate: 1

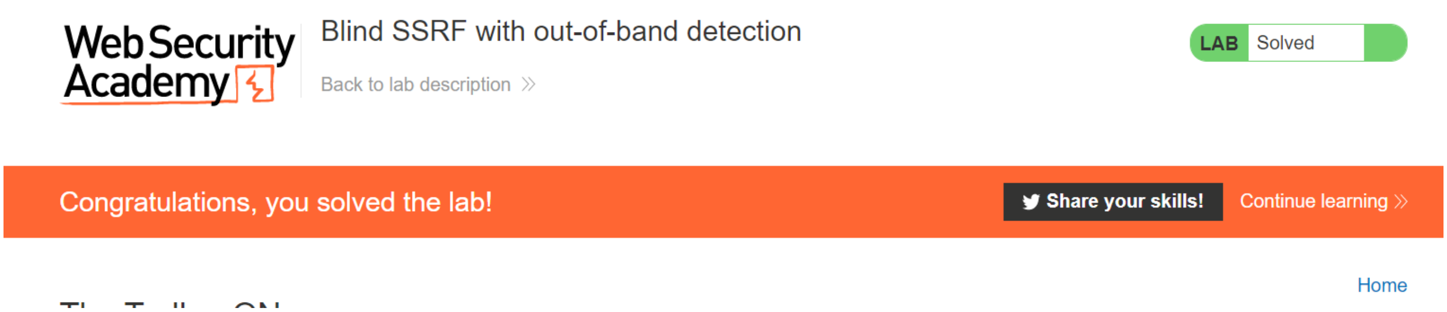
☑ Include Collaborator server location

Polling automatically

#	Time	Type	Payload	Source IP address	Comment
1	2024-Feb-15 13:20:20.597 UTC	DNS	tkhx0wunogfk8cix9u3x4fc1us0joec3	3.248.180.83	
2	2024-Feb-15 13:20:20.598 UTC	DNS	tkhx0wunogfk8cix9u3x4fc1us0joec3	3.251.104.212	
3	2024-Feb-15 13:20:20.673 UTC	HTTP	tkhx0wunogfk8cix9u3x4fc1us0joec3	34.251.122.40	

The Collaborator server received a DNS lookup of type A for the domain name **tkhx0wunogfk8cix9u3x4fc1us0joec3.oastify.com**.

The lookup was received from IP address 3.251.104.212:45433 at 2024-Feb-15 13:20:20.598 UTC.



Blind SSRF with out-of-band detection

Congratulations, you solved the lab!