ZROC102

zumaroc

Module 3: Testing Common Web Application Vulnerabilities (6) – SSRF

SSRF (SERVER-SIDE REQUEST FORGERY)

What is SSRF?

Server-side request forgery (also known as **SSRF**) is a web security vulnerability that allows an attacker to force the server-side application to make HTTP requests to an arbitrary domain of the attacker's choosing.

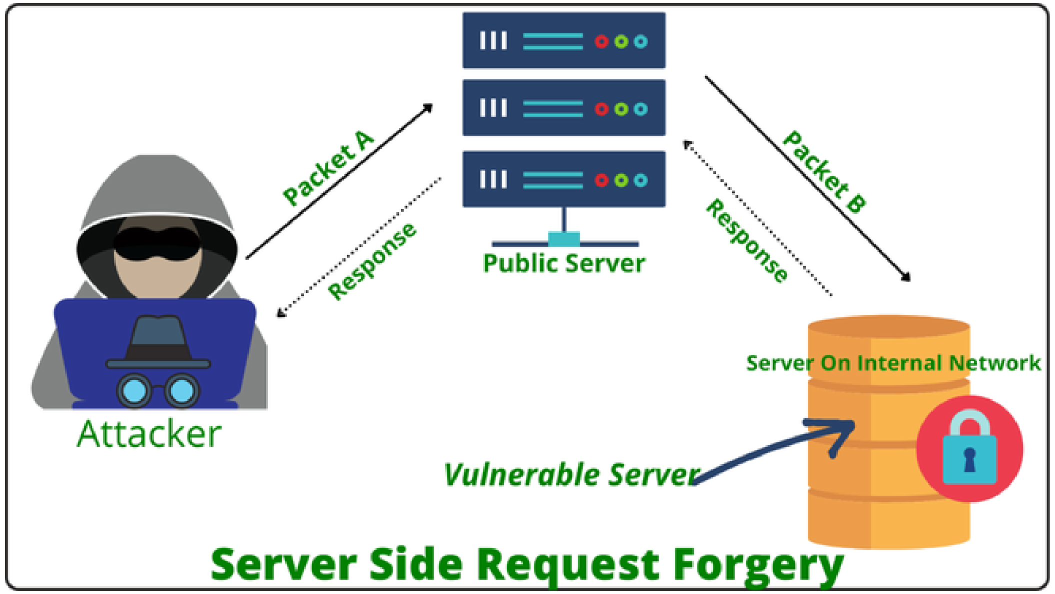
Modern web applications have a complex structure, and it’s common to have multiple services that constitute a web application. For example, you could have a service just to take care of payments, a service for backend and admin actions, and a client-facing service.

In a typical SSRF attack, an attacker might cause the server to make a connection to internal-only services within an organization's infrastructure. While in other cases, they may be able to force the server to connect to arbitrary external systems, potentially leaking sensitive data such as authorization credentials.

What is its impact?

A malicious actor can retrieve the content of arbitrary files on the system, which leads to sensitive information exposure (passwords, source code, confidential data, etc.).

SSRF ATTACK



The attacker sends a crafted **Packet A** to the publicly available server, and to fulfill the users query, the **public server** sends request to the back-end server with **Packet B** , as this request coming from the public server, the **internal network** would trust **Packet B** coming from back-end server and accept the packet and send response. This is possible because the attacker made a request on behalf of other servers.

TYPES OF SSRF

❖ Blind SSRF:

In a Blind SSRF, attackers are not able to control the data of **packet B** (shown in fig) that are sent to the application in a trusted internal network. Here, the attacker can control the IP address and ports of server. To exploit this type of SSRF we must feed it the **URL** followed by the **colon** and **port number**, by observing responses and error messages from the server we can find the open and close ports of server. We have tried this procedure for the different ports to check their status.

- ✓ **Example:**

```
http://site.com:22
http://site.com:80
```

❖ Limited Response / Partial SSRF :

In this type of SSRF, we get limited response from the server like title of the page or get access to resources but can't see the data. We can control only certain parts of **packet B** (shown in fig) that arrives in the internal application of this type of vulnerability & can be used to read local system files such as /etc/config, /etc/hosts, etc/passwd and many others. By using file:// protocol we can read file on the system. In some cases, XXE injection, DDoS, these type of vulnerabilities may be useful to exploit Partial SSRF Vulnerability.

```
file:///etc/passwd
file:///etc/config
```

TYPES OF SSRF (CONTD.)

- **Full Response SSRF :**
  In Full SSRF we have complete control over the **Packet B** (shown in fig). Now we can access the services running of the internal network and find the vulnerabilities in internal network. In this type of SSRF we can use the protocols like **file://, dict://, http://, gopher://,** etc. Here, we have large scope of creating different request and exploiting the internal network if any vulnerabilities are present. Full SSRF vulnerability may cause the application to crash through buffer overflow, by sending large strings in the request which then causes buffer overflow to occur.

  ✓ **Example:**

```
http://192.168.1.8/BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
```

HOW TO TEST FOR SSRF

- ✓ Always make sure that you are making request to back-end server on the behalf of public server. These back-end systems often have non-routable private IP addresses or are restricted to certain hosts.

- ✓ To fetch the data from server also try “http://localhost/xyz/” with the “http://127.0.0.1/xyz”.

- ✓ Server may have the firewall protection always try to bypass the firewall if possible.

BLOCKS ENCOUNTERED DURING SSRF TESTING

- ❑ **Whitelisting:** Server only allows the few domain name to be used in the request , server has a white-list of domain if domain name from that list matches with domain name from request, then only accept the request otherwise server decline the request.

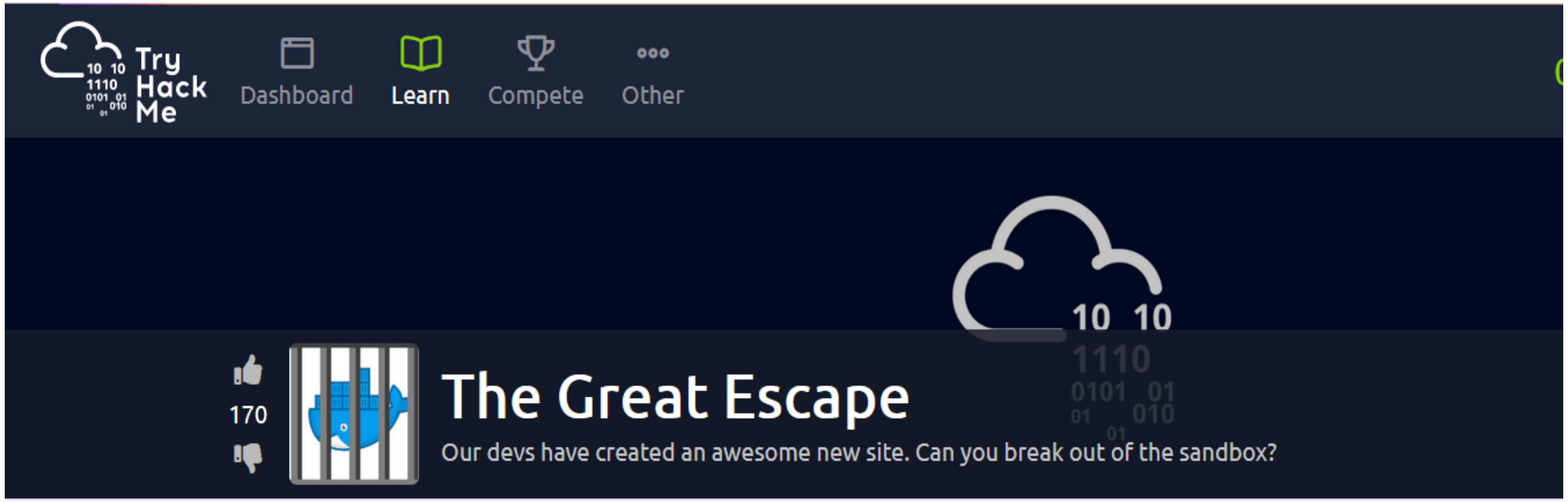
- ❑ **Blacklisting:** Server discards all requests containing IP addresses, domain names, keywords from the blacklists of the server.

- ❑ **Restricted content:** Server allows to access only particular number of files to user; it allows only the few file extension types for public access.

SSRF MITIGATION/PREVENTION BY OWASP

https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html

TRYHACKME (SSRF) - PRACTICAL



The Great Escape

Our devs have created an awesome new site. Can you break out of the sandbox?

Practical (Hands-on)

URL:

https://tryhackme.com/room/thegreatescape

SOURCES

SSRF
https://portswigger.net/web-security/ssrf
https://blog.sqreen.com/ssrf-explained/
https://www.geeksforgeeks.org/server-side-request-forgery-ssrf-in-depth/

SSRF Mitigations
https://blog.sqreen.com/ssrf-explained/