TryHackMe – Intro to SSRF

Learn how to exploit Server-Side Request Forgery (SSRF) vulnerabilities, allowing you to access internal server resources.

Topics that we will cover:

- 1 Task 1: What is an SSRF?
- 2 Task 2: SSRF Examples
- 3 Task 3: Finding an SSRF
- 4 Task 4: Defeating Common SSRF Defenses
- 5 Task 5: SSRF Practical

Task 1: What is an SSRF?

SSRF stands for Server-Side Request Forgery. It's a vulnerability that allows a malicious user to cause the webserver to make an additional or edited HTTP request to the resource of the attacker's choosing.

There are two types of SSRF vulnerability:

1. The first is a **regular SSRF** where data is returned to the attacker's screen.
2. The second is a **blind SSRF** vulnerability where an SSRF occurs, but no information is returned to the attacker's screen.

A successful SSRF attack can result in any of the following:

- Access to unauthorised areas.
- Access to customer/organisational data.
- Ability to Scale to internal networks.
- Reveal authentication tokens/credentials.

Question 1: What does SSRF stand for?

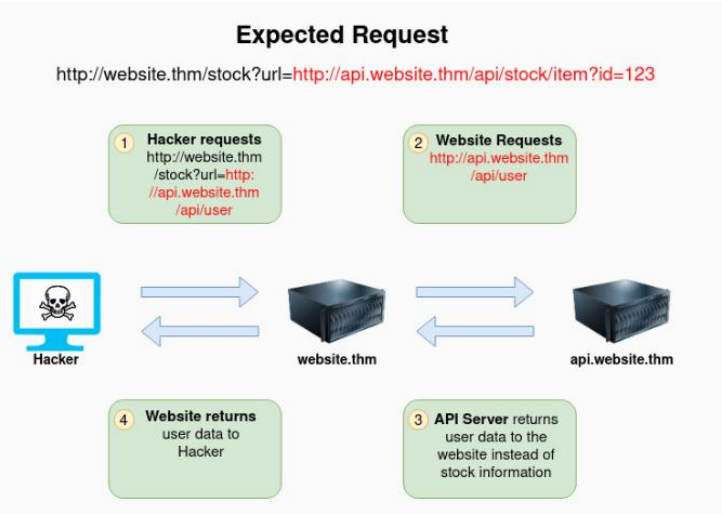
Answer: *Server-Side Request Forgery*

Question 2: As opposed to a regular SSRF, what is the other type?

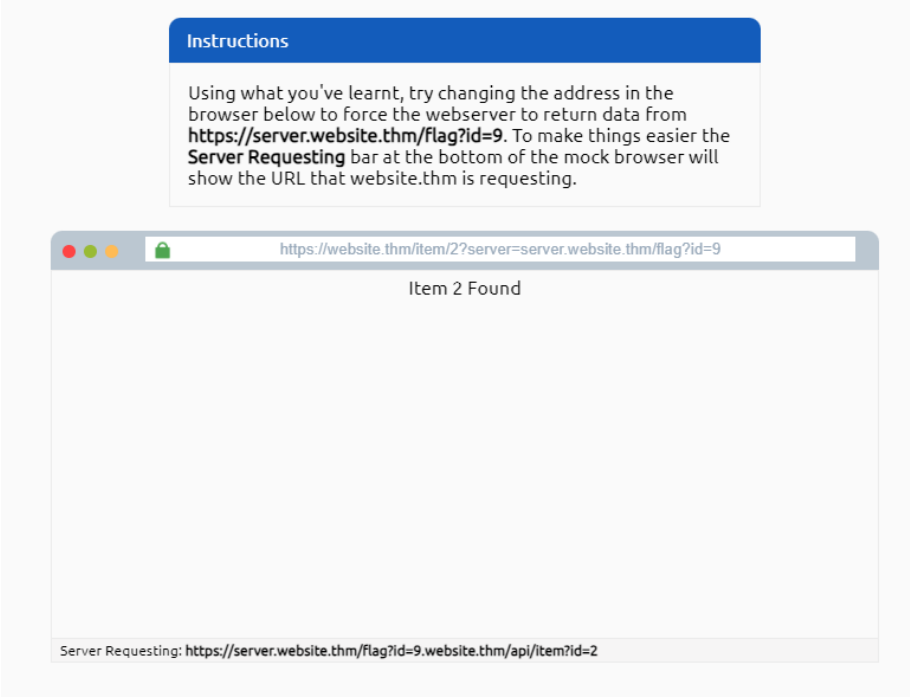
Answer: *blind*

Task 2: SSRF Examples

Click the View Site button, which will take you through some common SSRF examples, how to exploit them and even a simulation to see if you can take advantage of an SSRF vulnerability using what you've learnt.



Add the **server.website.thm/flag?id=9** at the end of the URL



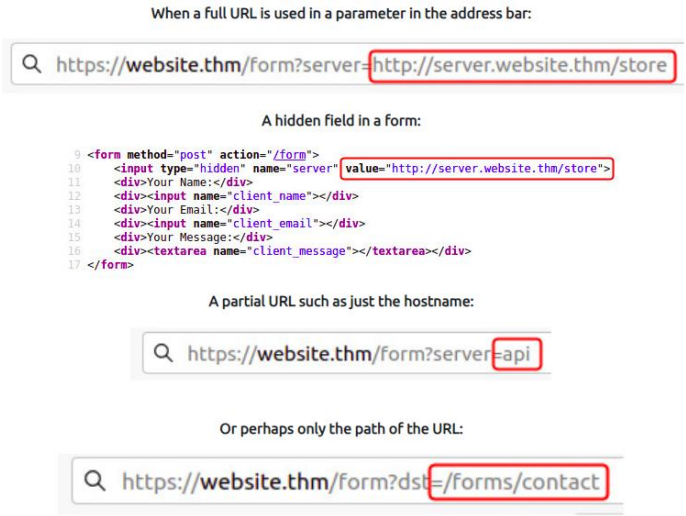
Looks like the server is still asking for "id=2". Add at the end of the URL "**&x=**", to bypass it. You will get the flag. If you append this "**&x=**" at the end of the URL, this will ignore the rest of the URL (which includes the id=2).

Question 3: What is the flag from the SSRF Examples site?

Answer: *THM{SSRF_MASTER}*

Task 3: Finding an SSRF

There are 4 common places to look to find an SSRF:



If working with a blind SSRF where no output is reflected back to you, you'll need to use an external HTTP logging tool to monitor requests such as requestbin.com, your own HTTP server or Burp Suite's Collaborator client.

Question 4: What website can be used to catch HTTP requests from a server?

Answer: *requestbin.com*

Task 4: Defeating Common SSRF Defenses

Deny List = where all requests are accepted apart from resources specified in a list or matching a particular pattern. In a cloud environment, it would be beneficial to block access to the IP address 169.254.169.254, which contains metadata for the deployed cloud server, including possibly sensitive information.

Allow List = where all requests get denied unless they appear on a list or match a particular pattern, such as a rule.

Open Redirect = is an endpoint on the server where the website visitor gets automatically redirected to another website address.

Question 5: What method can be used to bypass strict rules?

Answer: ***Open Redirect***

Question 6: What IP address may contain sensitive data in a cloud environment?

Answer: ***169.254.169.254***

Question 7: What type of list is used to permit only certain input?

Answer: ***Allow List***

Question 8: What type of list is used to stop certain input?

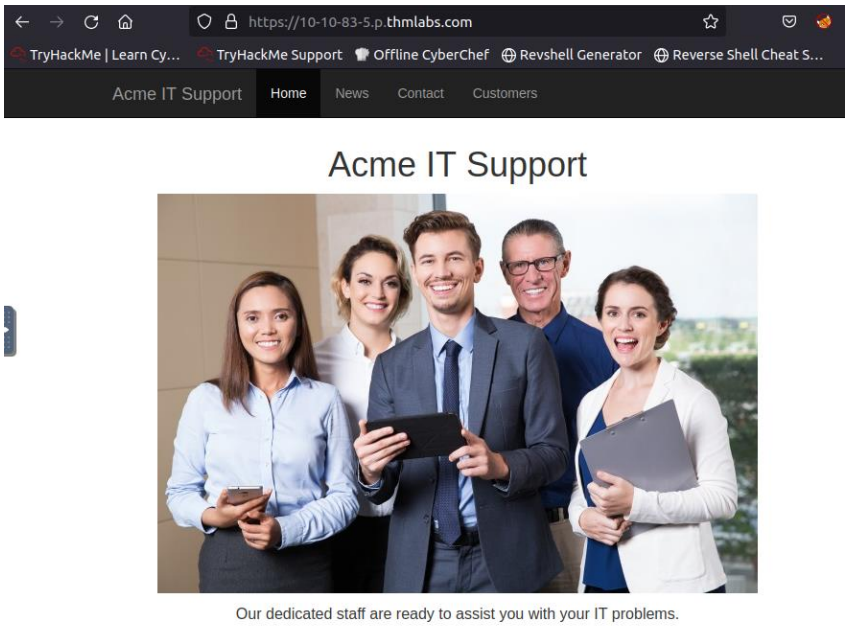
Answer: ***Deny List***

Task 5: SSRF Practical

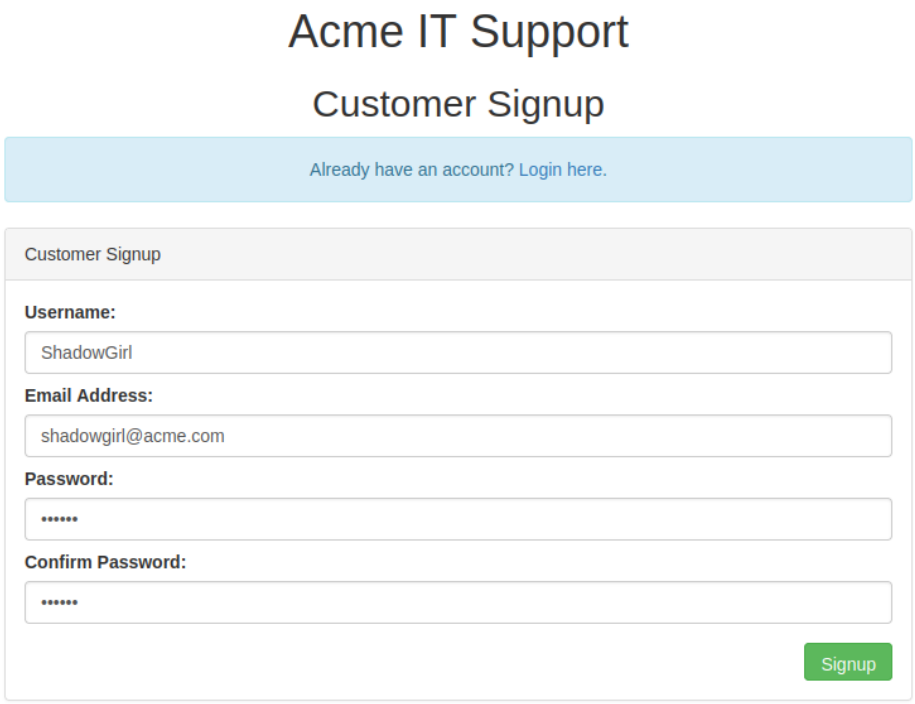
We've come across two new endpoints during a content discovery exercise against the **Acme IT Support** website. The first one is **/private**, which gives us an error message explaining that the contents cannot be viewed from our IP address. The second is a new version of the customer account page at **/customers/new-account-page** with a new feature allowing customers to choose an avatar for their account.

Begin by clicking the **Start Machine** button to launch the **Acme IT Support** website. Once running, visit it at the URL https://LAB_WEB_URL.p.thmlabs.com and then follow the instructions in the room to get the flag.

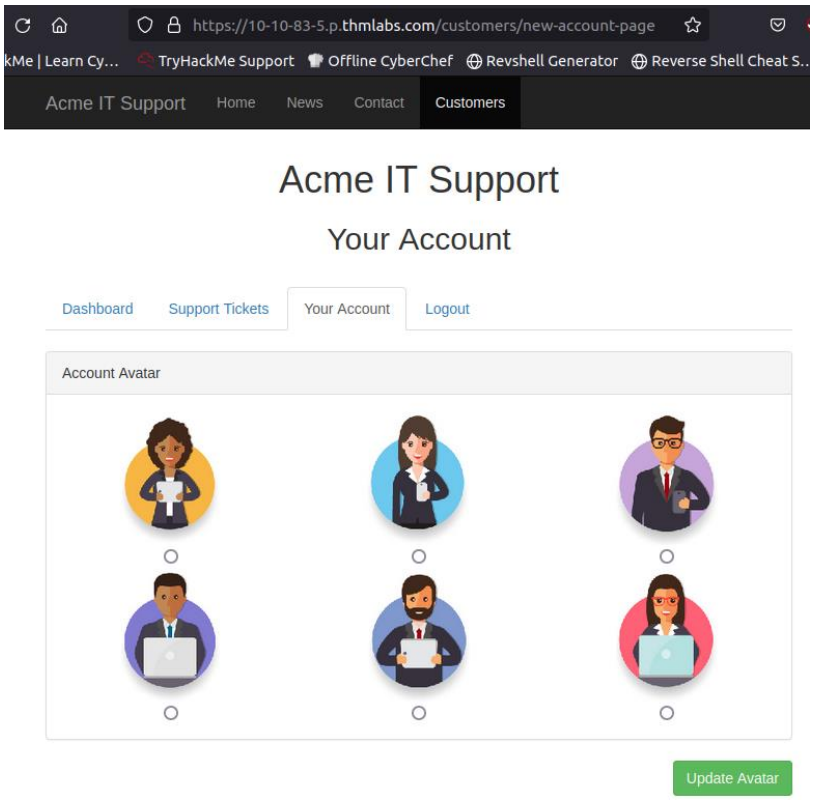
This is how the page looks like:



First, create a customer account and sign in.



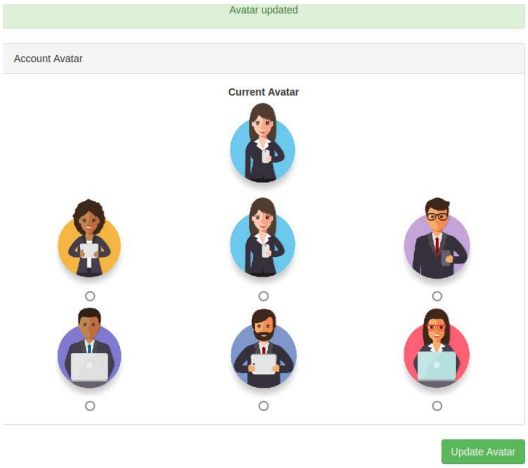
Once you've signed in, visit https://10-10-83-5.p.thmlabs.com/customers/new-account-page to view the new avatar selection feature. This is displayed:



Click on the Page Source and find the path to the image (of the avatar).

```
<div class="row text-center">
    <div class="col-xs-4">
        <div class="avatar-image" style="background-image: url('/assets/avatars/1.png')"></div>
        <input type="radio" name="avatar" value="assets/avatars/1.png">
    </div>
    <div class="col-xs-4">
        <div class="avatar-image" style="background-image: url('/assets/avatars/2.png')"></div>
        <input type="radio" name="avatar" value="assets/avatars/2.png">
    </div>
    <div class="col-xs-4">
        <div class="avatar-image" style="background-image: url('/assets/avatars/3.png')"></div>
        <input type="radio" name="avatar" value="assets/avatars/3.png">
    </div>
    <div class="col-xs-4">
        <div class="avatar-image" style="background-image: url('/assets/avatars/4.png')"></div>
        <input type="radio" name="avatar" value="assets/avatars/4.png">
    </div>
    <div class="col-xs-4">
        <div class="avatar-image" style="background-image: url('/assets/avatars/5.png')"></div>
        <input type="radio" name="avatar" value="assets/avatars/5.png">
    </div>
    <div class="col-xs-4">
        <div class="avatar-image" style="background-image: url('/assets/avatars/6.png')"></div>
        <input type="radio" name="avatar" value="assets/avatars/6.png">
    </div>
```

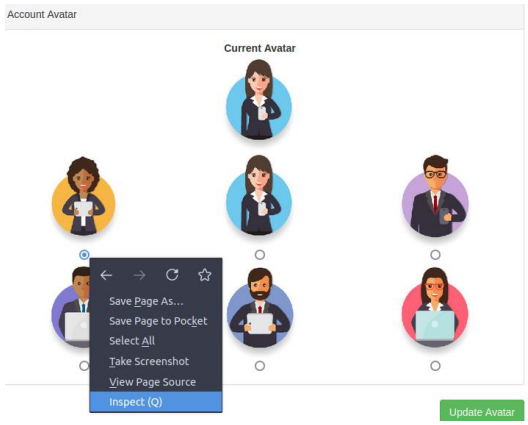
Return to the page with the avatars and select your avatar. Click “Update Avatar”.



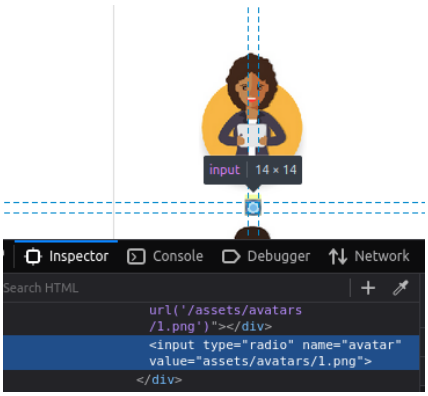
Reload the Page Source to see the changes. The image content is base64 encoded as per the screenshot below:

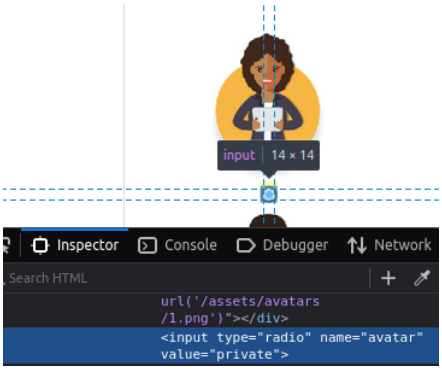
```
<div class="col-xs-6 col-xs-offset-3">
    <div><strong>Current Avatar</strong></div>
    <div class="avatar-image" style="background-image: url(
</div>
```

Now let's try making the request again but changing the avatar value to private in hopes that the server will access the resource and get past the IP address block. To do this, firstly, right-click on one of the radio buttons on the avatar form and select **Inspect**:

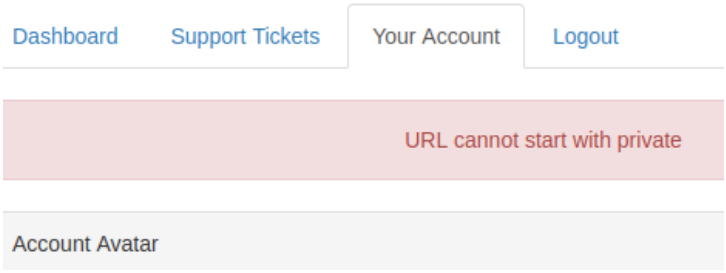


Edit the value of the radio button to private.





Click “Update Avatar”. An error appears because of the deny list in place.

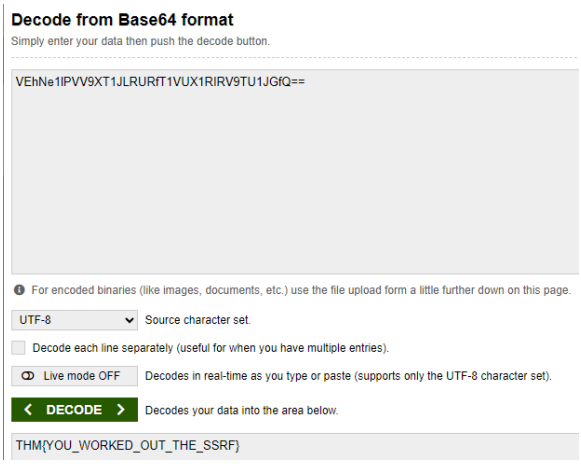


To bypass it, replace the **“/private”** with **“x/../private”**. Return to the Page Source. This is displayed:

```
<div class="col-xs-6 col-xs-offset-3">
    <div><strong>Current Avatar</strong></div>
    <div class="avatar-image" style="background-image: url(
</div>
```

Viewing the page source of the avatar form, you'll see the currently set avatar now contains the contents from the /private directory in base64 encoding, decode this content and it will reveal the flag.

To decode it, I used [Base64 Decode and Encode - Online] .



Question 8: What is the flag from the /private directory?

Answer: *THM{YOU_WORKED_OUT_THE_SSRF}*

Happy Hacking! 😸

> *Thanks and Regards,*
>
> *ShadowGirl* 🦹😉