NahamStore: CSRF

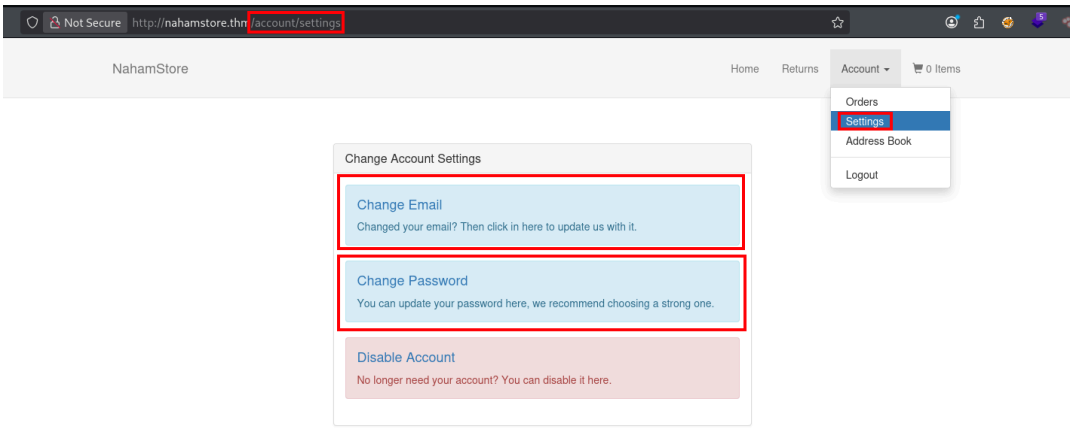
Introduction

Cross-Site Request Forgery (CSRF) is a web vulnerability that allows an attacker ***to force an authenticated user to perform unwanted actions within a web application without the user's knowledge***. This occurs when the application fails to properly verify whether a sensitive request originated legitimately from the user.

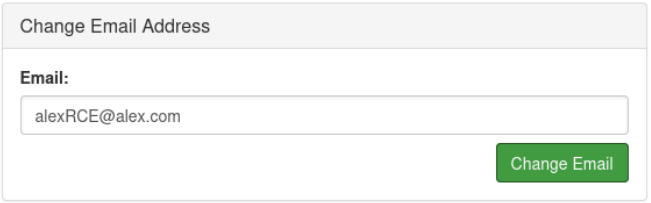
This type of vulnerability typically affects critical functionalities such as administrative actions, sensitive transactions, *password changes*, or *profile data updates*.

Looking for CSRF

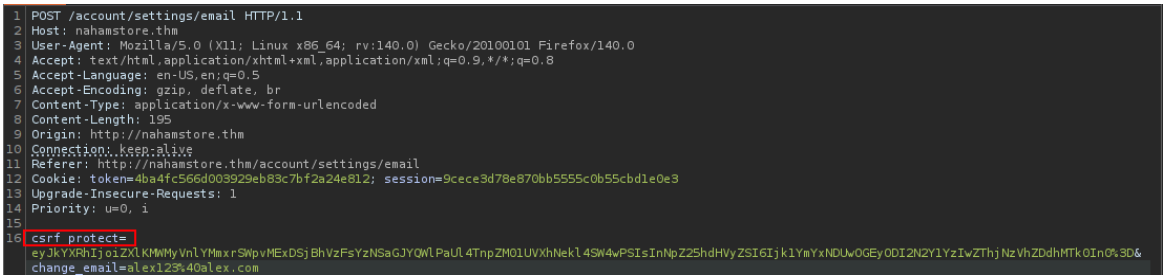
We'll look for a potential CSRF vulnerability in the /account/settings section while logged in, where we'll see "***Change Email***" and "***Change Password***".



For this example, we can choose either one. Let's start with "Change Email".



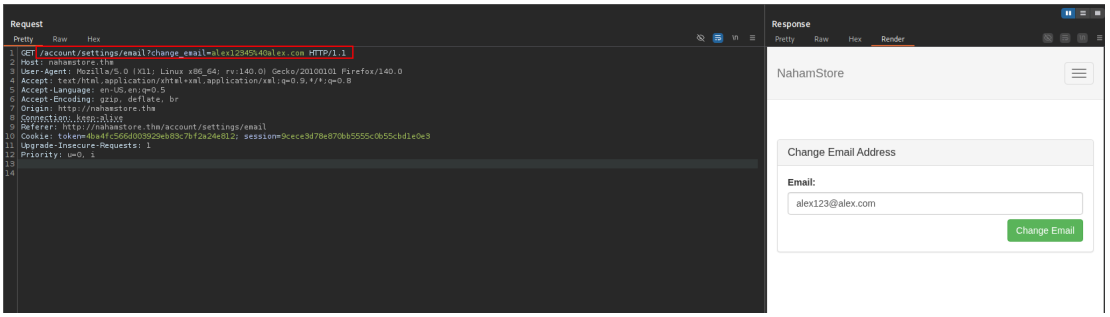
To do this, we will interpret the request with Burp Suite and send it to the repeater.



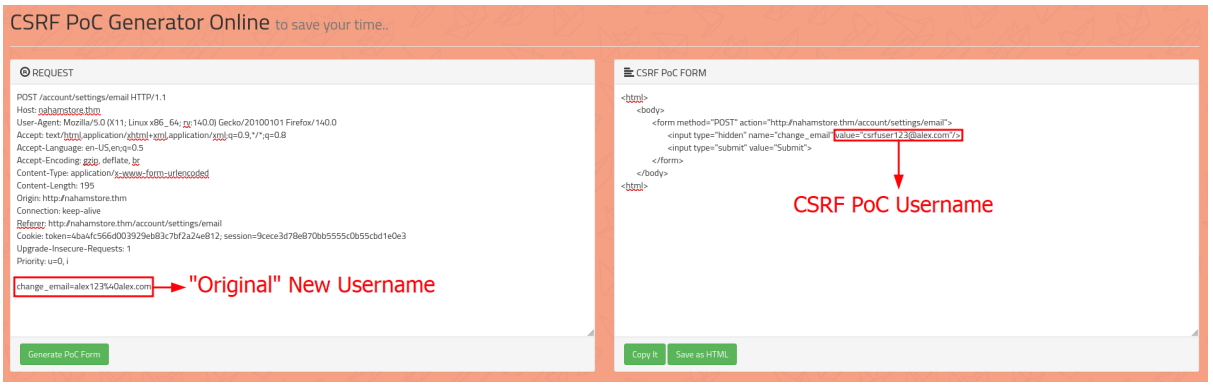
```
POST /account/settings/email HTTP/1.1
Host: nahamstore.thm
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 195
Origin: http://nahamstore.thm
Connection: keep-alive
Referer: http://nahamstore.thm/account/settings/email
Cookie: token=4ba4fc566d003929eb83c7bf2a24e812; session=9cece3d78e870bb5555c0b55cbd1e0e3
Upgrade-Insecure-Requests: 1
Priority: u=0, i

csrf_protect=
eyJkYXRhIjoiZXlKMWMyVnlYMmxrSWpvMExDSjBhVzFsYzNSaGJYQWlPaUl4TnpZM01UVXhNekl4SW4wPSIsInNpZ25hdHVyZSI6Ijk1YmYxNDUwOGEyODI2N2Y1YzIwZThjNzVhZDdhMTk0In0%3D&
change_email=alex123%40alex.com
```

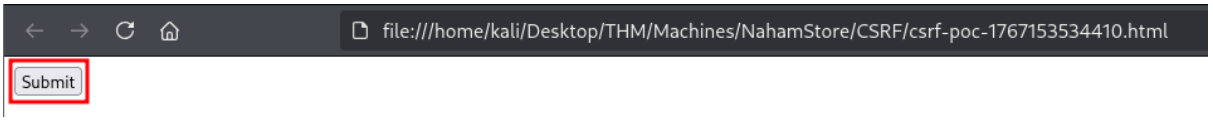
As we can see, there is a section called *"csrf_protect"* which is supposed to protect this section from this type of attack. What we'll do is change the "request method" in the repeater, along with everything related to "csrf_protect," and use a different email address. This can also be done by adding the parameters after the POST method to the URL.



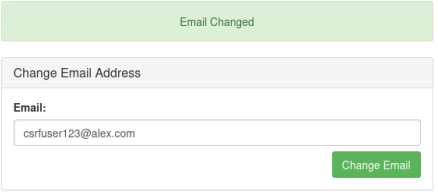
However, we see that this did not work out. However, we see that this doesn't work. So, to save time, we're going to use an online resource that will allow us to create a CSRF Proof of Concept (PoC) based on the intercepted website content to test if this section is vulnerable. That resource is https://tools.nakanosec.com/csrf/



So let's save it as HTML. If running this file results in the *username being changed to the one we specified in the Proof of Concept*, then it will confirm that this parameter is vulnerable to CSRF.



—

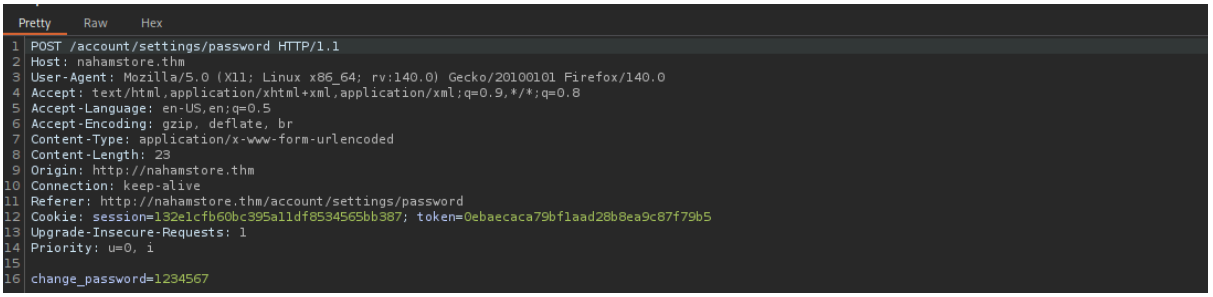


We have confirmed that the parameter for changing the email address is vulnerable to CSRF.

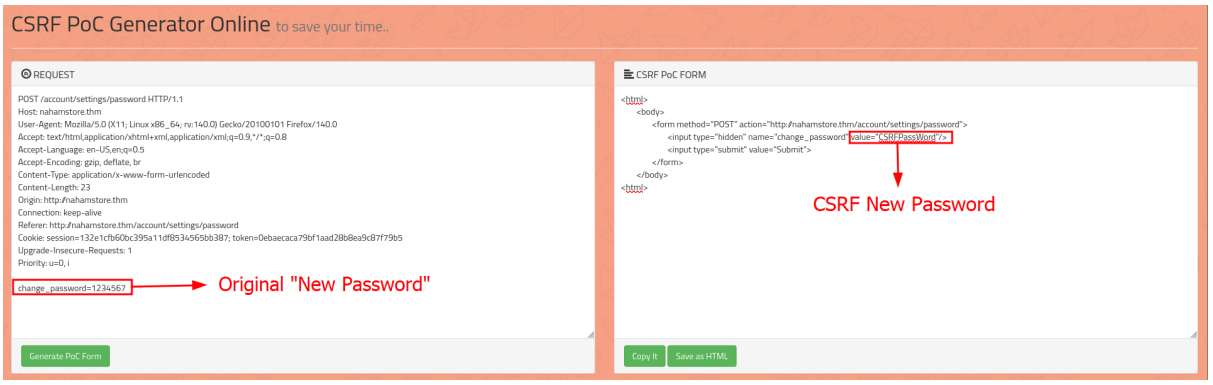
Second CSRF

For the second CSRF, we go to the ***"Change Password"*** section.

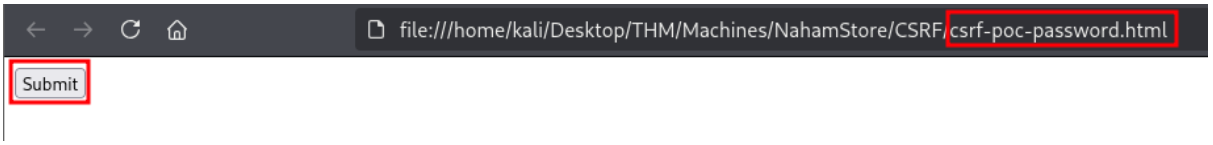
Here we follow the same methodology, that is, we will capture the request with Burp Suite and run it through the CSRF PoC from https://tools.nakanosec.com/csrf/ .



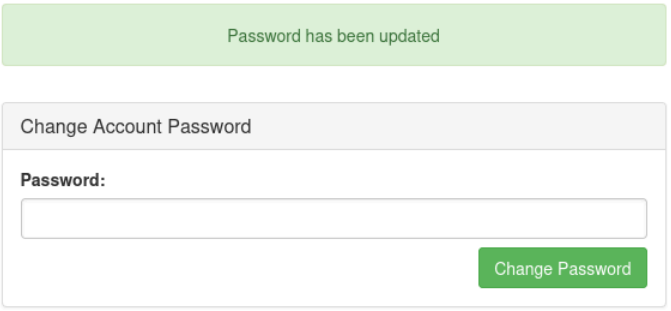
When intercepting the request, we can see something very interesting: ***there is no "csrf_protect"*** as there was in the *"Change Email"* section. This means we don't have to "bypass" anything and can go straight to the CSRF. And this is what we will do:



Once again, we save the content in HTML and run it.



Lets see the result:



With this, we have confirmed the second CSRF.

Conclusion.

With this, we have identified the CSRF vulnerabilities present in NahamStore. Next, we will focus on one of the most common vulnerabilities in websites today: *IDORs*.