CSRF

- using burp pro will be better .
- csrf is all about changing the email of the user using a link once the link is opened the email is changed for what i forged (also applies to passwords).
- current password is a solution for this because it validates the current and the enter the new password to change it.
- to stop the csrf servers uses csrf token that validates the request token
- csrf token are generated in the back-end using timestamp or using the user cookies or randomly generated but it will be a one time only use or using the user id

How the developers try to stop csrf :

1. csrf tokens : developers make a hidden parameter called csrf tokens where it have a unique value for each user and that are randomly generated by the server

> !! **we try to brake this by intercepting the request to get the csrf token and change or copy it to try and access the account our selves and get an account take over**
> 2. same site : is used by most browsers by configuring it to (**strict or lax**)

```
- strict : the client browser won't send the cookies during cross site
requests
- lax : tells the client browser to send cookies only in requests when user
click a link to navigate to the site
```

3. referrer header : developers also user referrer headers to filter the server requests that are not from the original website ,**however it can be bypassed by changing it to local host or the original website url**

Scenario

1. the current password can be deleted in the request and if the server doesn't validate it we can change the password without current password.
2. change the request method is a way of delivering a csrf
3. if the session is invalid for the use we can use another and change its value in the request will be the same thing
4. the token may be one time use only so by refreshing the session we will get a new token but make sure to drop the refreshed session to not use the new token and then change the old token with the new one and we will be able to change the password or the email.
   CRLF : to make it we do it by injecting headers in the request as we want
5. to achieve a csrf sometimes we need to chain a CRLF by using an injection for the cookies
6. if the website allows state changing requests with the GET http request

   > **!! if the website allows to change password using a GET http request**

7. some websites requires the third party sites to send authenticated requests *in this case we can change the same site parameter to none* allowing to send cookies of the users to implement the csrf.
8. any user who is not using a browser that implements a same site policy is vulnerable to csrf.

How to Hunt for csrf :

1. visit the target site and start searching for any **state changing events** that includes all the changing password or emails or usernames or permissions any thing that change the user statues and then collect all of them in a list to visit later
2. we go through each and every link of the last step using burp suite checking if there is any csrf protection on them , and use the search to find any csrf keywords
3. to confirm our finding we make a script that we are going to execute ourselves

```html
<html>
    <form method="post" action="target infected url" id="csrf-form">
    <input type="text" name="new-passowrd" value="u have been hacked">
    <input type="submit" value="submit">
    </form>
    <script>
    document.getElementById("csrf-form").submit;
    </script>
</html>
```

then we run this using the signed in browser for the target and check if our password has been changed or not

4. we need to check the referrer header where if the server filters request that aren't from the same referrer header we try to change the address to just move on to the next endpoint