Ting Yan

tyan37

Target 1 Epilogue

Vulnerability

The vulnerable code is in account.php

```
if ($_POST['response'] != $expected) {
        notify('CSRF attempt prevented!'.$teststr.'--'.$_POST['response'].' !=
'.$expected, -1);
```

Reason: this line will notify the attack the expected value. So we can hardcode response the same value as expected value, then we can bypass the CSRF protection.

How to prevent XSRF:

In our target, we can delete the `$_POST['response']` in the `notify function()`, so that the attack can not hardcode the response. After reading related souce online, it mentions a better solution. Sending a unique identifier with each request, along with a unique user "sessionID" added into it, to defeat against CSRF. Those identifier are long and randomly generated. So it is impossible for the attacker to know the unique identifier.

Target 2 Epilogue

Vulnerability

The vulnerable code is in index.php, the input is not escaped before it's rendered. The attacker can execute arbitrary javascript in the login input field and steal login credentials.

How to prevent XSS:

Using character and string escaping can help prevent XSS. Escaping is to make sure that every part of a string is interpreted as a string primitive, not as a control character or code. By escaping the `<script>` tags, we prevented the script from executing.

Target 3 Epilogue

Vulnerability

In `auth.php login()function`, even though it has sqli_filter($username) function to escape username, but it not sufficient. Attacker can still bypass it and add injection in the query:

```
$sql = “SELECT user_id, name eid FROM users WHERE eid = ‘$username’ AND
password = ‘$hash’”;
```

By adding “----” after the valid username, everything after the -- character will be discarded in the query, so attacker does not need password to login.

How to prevent SQL injection:

1. We need to employ comprehensive data sanitization, for example, username should be filtered to allow only the characters and numbers.
2. Avoid constructing SQL queries with user input. In the query above, we should not use `$username` in the query, but use `$escaped_username`.