The project was successfully retested.

php-cron-requests-events main

Code Analysis

Overview History Settings

Created Thu 12th Jan 2023
Snapshot taken by snyk.io a few seconds ago

| Retest now

IMPORTED BY

E Евгений

PROJECT OWNER

Add a project owner

ANALYSIS SUMMARY

4 analyzed files (31%) Repo breakdown

Search...

19 of 19 issues

Group by **none** Sort by **highest severity**

H Command Injection

SCORE
822

SNYK CODE CWE-78

```
        if(
            is_callable("shell_exec") &&
            $wget !== ''
        ){
            if($protocol ===  'https') shell_exec($wget . ' -T 1 --no-check-
```

Unsanitized input from *an HTTP header flows* into *shell_exec*, where it is used to build a shell command. This may result in a Command Injection vulnerability.

cron.php

16 steps in 1 file

Ignore

Full details

H Command Injection

SNYK CODE CWE-78

SCORE
822

```
            is_callable("shell_exec") &&
            $wget !== ''
        ){
            if($protocol ===  'https') shell_exec($wget . ' -T 1 --no-check-c
            else shell_exec($wget . ' -T 1 --delete-after -q "' . $cron_url .
```

Unsanitized input from *an HTTP header flows* into *shell_exec*, where it is used to build a shell command. This may result in a Command Injection vulnerability.

cron.php

16 steps in **1** file

Ignore Full details

H Command Injection

SNYK CODE CWE-78

SCORE
822

```
        } elseif(
            is_callable("shell_exec") &&
            $curl !== ''
        ){
            if($protocol ===  'https') shell_exec($curl . ' -I -k --connect-t
```

Unsanitized input from *an HTTP header flows* into *shell_exec*, where it is used to build a shell command. This may result in a Command Injection vulnerability.

cron.php

16 steps in **1** file

Ignore Full details

H Command Injection

SNYK CODE CWE-78

SCORE
822

```
            is_callable("shell_exec") &&
            $curl !== ''
        ){
            if($protocol ===  'https') shell_exec($curl . ' -I -k --connect-t
```

Unsanitized input from *an HTTP header flows* into *shell_exec*, where it is used to build a shell command. This may result in a Command Injection vulnerability.

cron.php

16 steps in **1** file

Ignore Full details

H Cross-site Scripting (XSS)

SNYK CODE CWE-79

SCORE
806

```
        $protocol= 'https';
        $host= "localhost";
        $document_root= dirname(__FILE__) . DIRECTORY_SEPARATOR; // site

        echo "Request: " . $protocol . '://' . $host . "/" . basename(__F
```

Unsanitized input from *an HTTP header flows* into *the echo statement*, where it is used to render an HTML page returned to the user. This may result in a Cross-Site Scripting attack (XSS).

cron.php

16 steps in **1** file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

H Path Traversal

SNYK CODE CWE-23

SCORE
806

```
            if($protocol ===  'https') shell_exec($curl . ' -I -k --connect-t
            else shell_exec($curl . ' -I --connect-timeout 1 "' . $cron_url
        } else {
            @fclose(
                @fopen(
```

> Unsanitized input from *an HTTP header flows* into *fopen*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to open arbitrary files.
>
> cron.php — **14** steps in **1** file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```

                }
            }

            $boot= unserialize(queue_address_pop(4096, 0, '', "init_boot_fra
```

> Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.
>
> cron.php — **16** steps in **1** file

Ignore | Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```

            $boot= unserialize(queue_address_pop(4096, 0, '', "init_boot_fra
            if(!is_array($boot)) return; // file read error
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

16 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```
// examples use adressed mode
if(is_array($boot) && count($boot['handlers']) === 1): // first h
    // example 1, get first element
    $value= unserialize(queue_address_pop($frame_size, $index_dat
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

16 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```
    //usleep(2000); // test load, micro delay


    // example 2, get last - 10 element, and get first frame in
    $value= unserialize(queue_address_pop($frame_size, $index_dat
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

16 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```

                    // example 3, linear read
                    for($i= 100; $i < 800; $i++){ // execution time:  0.03701186
                        $value= unserialize(queue_address_pop($frame_size, $index
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

16 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```


                    // example 4, replace frames in file
                    for($i= 10; $i < 500; $i++){ // execution time:  0.076093912
                        $value= unserialize(queue_address_pop($frame_size, $index
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

16 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE
603

```

                    // example 5, random access
                    shuffle($index_data);
                    for($i= 0; $i < 10; $i++){// execution time: 0.0353598594665
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

16 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE

603

```
        // execution time: 0.051764011383057 end - start, 1000 cycles
        while(true){ // example: loop from the end
            $frame= queue_address_pop($frame_size,  PHP_INT_MAX, '', "cou
            $value= unserialize($frame);
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

17 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SNYK CODE CWE-502

SCORE

603

```

            $cron_resource= fopen($cron_dat_file, "r+");
            if(flock($cron_resource, LOCK_EX | LOCK_NB)) {
                $stat= fstat($cron_resource);
                $cs= unserialize(fread($cron_resource, $stat['size']));
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php

8 steps in **1** file

Ignore

Full details

M Deserialization of Untrusted Data

SCORE
603

SNYK CODE CWE-502

```

$cron_resource= fopen(CRON_DAT_FILE, "r+");
if(flock($cron_resource, LOCK_EX | LOCK_NB)) {
    $stat= fstat($cron_resource);
    $cs= unserialize(@fread($cron_resource, $stat['size']));
```

Unsanitized input from *data from a remote resource flows* into *unserialize*, where it is used to deserialize an object. This may result in an Unsafe Deserialization vulnerability.

cron.php 9 steps in 1 file

Ignore Full details

M Use of Password Hash With Insufficient Computational Effort

SCORE
566

SNYK CODE CWE-916

```
    if(isset($init[$job_process_id])) return;
    $init[$job_process_id]= true;

    if(isset($cron_session[$job_process_id]['md5'])) {
        if($cron_session[$job_process_id]['md5'] !== md5(serialize($job)
```

MD5 hash (used in *md5*) is insecure. Consider changing it to a secure hashing algorithm.

cron.php 1 step in 1 file

Ignore Full details

M Use of Password Hash With Insufficient Computational Effort

SCORE
566

SNYK CODE CWE-916

```
    $init[$job_process_id]= true;

    if(isset($cron_session[$job_process_id]['md5'])) {
        if($cron_session[$job_process_id]['md5'] !== md5(serialize($job)
            $cron_session[$job_process_id]['md5']= md5(serialize($job));
```

MD5 hash (used in *md5*) is insecure. Consider changing it to a secure hashing algorithm.

cron.php

1 step in **1** file

Ignore

Full details

M Use of Password Hash With Insufficient Computational Effort

SCORE
566

SNYK CODE CWE-916

```
        $cron_session[$job_process_id]['last_update']= 0;
        $cron_session[$job_process_id]['complete']= false;
    }
} else {
    $cron_session[$job_process_id]['md5']= md5(serialize($job));
```

MD5 hash (used in *md5*) is insecure. Consider changing it to a secure hashing algorithm.

cron.php

1 step in **1** file

Ignore

Full details