Talos Vulnerability Report

TALOS-2020-1083

OS4Ed openSIS install remote code execution vulnerability

AUGUST 31, 2020

CVE NUMBER

CVE-2020-6143, CVE-2020-6144

Summary

A remote code execution vulnerability exists in the install functionality of OS4Ed openSIS 7.4. A specially crafted HTTP request can lead to remote code execution. An attacker can send an HTTP request to trigger this vulnerability.

Tested Versions

OS4Ed openSIS 7.4

Product URLs

https://opensis.com/

CVSSv3 Score

10.0 - CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

CWE

CWE-96 - Improper Neutralization of Directives in Statically Saved Code ('Static Code Injection')

Details

openSIS is a student information system and school management system. It is available in commercial and open-source versions. It allows schools to create schedules and track attendance, grades and transcripts.

Opensis performs its install via the web, after the user has downloaded and unpacked the software into a web directory, they must go to http://www.example.com/opensis/install. No authentication is required for that URL as it is supposed to only be used for a first install. However, the install directory is never removed, allowing anyone to access this location and overwrite the current installation. The installer asks for a hostname, port, username and password for a mysql server. It then verifies that it can connect and then asks for the database name that it should create. All this information is written into `Data.php` in the opensis directory and is included by every page that opensis works with. It subsequently lets the user create a username and password that they can use to log into the site and provides the user with a link to their newly configured opensis installation.

`install/Step5.php` writes the `Data.php` file at lines 113 and below:

```
} elseif ($_SESSION['mod'] != 'upgrade') {
    $myFile = "../Data.php";
    $fh = fopen($myFile, 'w');

    if ($fh == TRUE) {
        $string .= "<" . "?php \n";
        $string .= "$" . "DatabaseType = 'mysqli'; \n";
        $string .= "$" . "DatabaseServer = '" . $_SESSION['server'] . "'; \n";
        $string .= "$" . "DatabaseUsername = '" . $_SESSION['username'] . "'; \n";
        $string .= "$" . "DatabasePassword = '" . $_SESSION['password'] . "'; \n";
        $string .= "$" . "DatabaseName = '" . $_SESSION['db'] . "'; \n";
        $string .= "$" . "DatabasePort = '" . $_SESSION['port'] . "'; \n";
        $string .="?" . ">";

        fwrite($fh, $string);
```

Since the information is used to verify if a connection to the database can be made, server and port can't be misused. The variable db is also sanitized because it's used as a database name. However if an attacker runs their own mysql server instance they can simply start that with the `skip-grant-tables` option, which allows all users to log in regardless of username or password, which leads to vulnerabilities in this configuration.

CVE-2020-6143 - password variable

The password variable which is set at line 122 in `install/Step5.php` allows for injection of PHP code into the Data.php file that it writes.

An example string that can be entered is:

```
'; phpinfo();//
```

This will result in `phpinfo()` being executed when surfing to the newly "installed" version at http://www.example.com/opensis

CVE-2020-6144 - username variable

The username variable which is set at line 121 in `install/Step5.php` allows for injection of PHP code into the Data.php file that it writes.

An example string that can be entered is:

```
'; phpinfo();//
```

This will result in `phpinfo()` being executed when surfing to the newly “installed” version at http://www.example.com/opensis

Timeline

2020-06-02 - Vendor Disclosure
2020-08-13 - Vendor provided patch to Talos for testing
2020-08-17 - Talos confirmed patch resolved issue
2020-08-31 - Public Release

CREDIT

Discovered by Yves Younan of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT TALOS-2020-1081

NEXT REPORT TALOS-2020-1082