Talos Vulnerability Report

TALOS-2020-1080

OS4Ed openSIS Password Reset Multiple SQL injection vulnerabilities

AUGUST 31, 2020

CVE NUMBER

CVE-2020-6137, CVE-2020-6138, CVE-2020-6139, CVE-2020-6140

Summary

Multiple SQL injection vulnerabilities exist in the password reset functionality of OS4Ed openSIS 7.3. A specially crafted HTTP request can lead to SQL injection. An attacker can send an HTTP request to trigger this vulnerability.

Tested Versions

OS4Ed openSIS 7.3

Product URLs

https://opensis.com/

CVSSv3 Score

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

CWE

CWE-89 - Improper Neutralization of Special Elements used in an SQL Command ('SQL 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.

Multiple SQL injections exist in the password reset functionality of openSIS 7.3. A successful attack could allow an attacker to access information such as usernames and password hashes as well as other data stored in the database.

CVE-2020-6137 - `password_stf_email` parameter

The `password_stf_email` parameter in the password reset page `/opensis/ResetUserInfo.php` is vulnerable to SQL injection.

Below is an example post that will trigger the vulnerability:

```
POST /opensis/ResetUserInfo.php HTTP/1.1
Host: [IP]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:74.0) Gecko/20100101 Firefox/74.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 167
Origin: http://[IP]
DNT: 1
Connection: close
Referer: http://[IP]/opensis/ForgotPass.php
Upgrade-Insecure-Requests: 1

pass_user_type=pass_parent&pass_type_form=password&password_stn_id=&uname=za&month_password_dob=&day_password_dob=&year_password_dob=&pass_e
mail=&password_stf_email=aa[SQL INJECTION]
```

The vulnerable code for this parameter is also at line 313:

```
301     if ($_REQUEST['pass_user_type'] == 'pass_parent') {
302         if ($_REQUEST['uname'] == '') {
303             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Username.</b></font>';
304             echo'<script>window.location.href="ForgotPass.php"</script>';
305         }
306         if ($_REQUEST['password_stf_email'] == '') {
307             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Email Address.</b></font>';
308             echo'<script>window.location.href="ForgotPass.php"</script>';
309         }
310 
311         if ($_REQUEST['password_stf_email'] != '' && $_REQUEST['uname'] != '') {
312 
313             $par_info = DBGet(DBQuery('SELECT p.* FROM people p,login_authentication la  WHERE la.USER_ID=p.STAFF_ID AND la.USERNAME=\''
. $_REQUEST['uname'] . '\' AND p.EMAIL=\'' . $_REQUEST['password_stf_em    ail'] . '\' AND la.PROFILE_ID = 4'));
314 
315             if ($par_info[1]['STAFF_ID'] == '') {
316                 $_SESSION['err_msg'] = '<font color="red" ><b>Incorrect login credential.</b></font>';
317                 echo'<script>window.location.href="ForgotPass.php"</script>';
318             } else {
319                 $flag = 'par_pass';
320             }
321         }
322     }
```

CVE-2020-6138 - `uname` parameter

The `uname` parameter in the password reset page `/opensis/ResetUserInfo.php` is vulnerable to SQL injection. Below is an example post that will trigger the vulnerability:

```
POST /opensis/ResetUserInfo.php HTTP/1.1
Host: [IP]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:74.0) Gecko/20100101 Firefox/74.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 167
Origin: http://[IP]
DNT: 1
Connection: close
Referer: http://[IP]/opensis/ForgotPass.php
Upgrade-Insecure-Requests: 1

pass_user_type=pass_parent&pass_type_form=password&password_stn_id=&uname=za[SQL
INJECTION]&month_password_dob=&day_password_dob=&year_password_dob=&pass_email=&password_stf_email=aa
```

The vulnerable code for `opensis/ResetUserInfophp` is at line 313 is due to a lack of input sanitation leading:

```
301     if ($_REQUEST['pass_user_type'] == 'pass_parent') {
302         if ($_REQUEST['uname'] == '') {
303             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Username.</b></font>';
304             echo'<script>window.location.href="ForgotPass.php"</script>';
305         }
306         if ($_REQUEST['password_stf_email'] == '') {
307             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Email Address.</b></font>';
308             echo'<script>window.location.href="ForgotPass.php"</script>';
309         }
310 
311         if ($_REQUEST['password_stf_email'] != '' && $_REQUEST['uname'] != '') {
312 
313             $par_info = DBGet(DBQuery('SELECT p.* FROM people p,login_authentication la  WHERE la.USER_ID=p.STAFF_ID AND la.USERNAME=\''
. $_REQUEST['uname'] . '\' AND p.EMAIL=\'' . $_REQUEST['password_stf_em    ail'] . '\' AND la.PROFILE_ID = 4'));
314 
315             if ($par_info[1]['STAFF_ID'] == '') {
316                 $_SESSION['err_msg'] = '<font color="red" ><b>Incorrect login credential.</b></font>';
317                 echo'<script>window.location.href="ForgotPass.php"</script>';
318             } else {
319                 $flag = 'par_pass';
320             }
321         }
322     }
```

CVE-2020-6139 - `username_stf_email` parameter

The `username_stf_email` parameter in the password reset page `/opensis/ResetUserInfo.php` is vulnerable to SQL injection.

Below is an example post that will trigger the vulnerability:

```
POST /opensis/ResetUserInfo.php HTTP/1.1
Host: [IP]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:74.0) Gecko/20100101 Firefox/74.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 162
Origin: http://[IP]
DNT: 1
Connection: close
Referer: http://[IP]/opensis/ForgotPass.php
Upgrade-Insecure-Requests: 1

user_type_form=username&uname_user_type=uname_staff&username_stn_id=1&pass=12&month_username_dob=1&day_username_dob=1&year_username_dob=1994
&username_stf_email=1[SQL INJECTION
```

The vulnerable code in this case is at line 340:

```
324 if ($_REQUEST['user_type_form'] == 'username') {
325     if ($_REQUEST['uname_user_type'] == 'uname_student') {
326         if ($_REQUEST['username_stn_id'] == '') {
327             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Student Id.</b></font>';
328             echo'<script>window.location.href="ForgotPass.php"</script>';
329         }
330         if ($_REQUEST['pass'] == '') {
331             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Password.</b></font>';
332             echo'<script>window.location.href="ForgotPass.php"</script>';
333         }
334         if ($_REQUEST['month_username_dob'] == '' || $_REQUEST['day_username_dob'] == '' || $_REQUEST['year_username_dob'] == '') {
335             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Birthday Properly.</b></font>';
336             echo'<script>window.location.href="ForgotPass.php"</script>';
337         }
338
339         if ($_REQUEST['username_stn_id'] != '' && $_REQUEST['pass'] != '' && $_REQUEST['month_username_dob'] != '' &&
$_REQUEST['day_username_dob'] != '' && $_REQUEST['year_username_dob'] != '') {
340             $stu_dob = $_REQUEST['year_username_dob'] . '-' . $_REQUEST['month_username_dob'] . '-' . $_REQUEST['day_username_dob'];
341             $stu_info = DBGet(DBQuery('SELECT s.* FROM students s,login_authentication la  WHERE la.USER_ID=s.STUDENT_ID AND
la.PASSWORD=\'' . md5($_REQUEST['pass']) . '\' AND s.BIRTHDATE=\'' . date('Y-m-d',     strtotime($stu_dob)) . '\' AND s.STUDENT_ID=' .
$_REQUEST['username_stn_id'] . ''));
342
343             if ($stu_info[1]['STUDENT_ID'] == '') {
344                 $_SESSION['err_msg'] = '<font color="red" ><b>Incorrect login credential.</b></font>';
345                 echo'<script>window.location.href="ForgotPass.php"</script>';
346             } else {
347                 $get_uname = DBGet(DBQuery('SELECT USERNAME FROM login_authentication WHERE USER_ID=' . $_REQUEST['username_stn_id'] . '
AND PROFILE_ID=3'));
348                 $_SESSION['fill_username'] = $get_uname[1]['USERNAME'];
349                 echo'<script>window.location.href="index.php"</script>';
350             }
351         }
352     }
```

CVE-2020-6140 - `username_stn_id` parameter

The `password_stf_email` parameter in the password reset page `/opensis/ResetUserInfo.php` is vulnerable to SQL injection.

Below is an example post that will trigger the vulnerability:

```
POST /opensis/ResetUserInfo.php HTTP/1.1
Host: [IP]
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:74.0) Gecko/20100101 Firefox/74.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 143
Origin: http://[IP]
DNT: 1
Connection: close
Referer: http://[IP]/opensis/ForgotPass.php
Upgrade-Insecure-Requests: 1

user_type_form=username&uname_user_type=uname_student&username_stn_id=1[SQL
INJECTION]&pass=12&month_username_dob=1&day_username_dob=1&year_username_dob=1994
```

The vulnerable code for this issue is at line 365:

```
353     if ($_REQUEST['uname_user_type'] == 'uname_staff') {
354
355         if ($_REQUEST['pass'] == '') {
356             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Password.</b></font>';
357             echo'<script>window.location.href="ForgotPass.php"</script>';
358         }
359         if ($_REQUEST['username_stf_email'] == '') {
360             $_SESSION['err_msg'] = '<font color="red"><b>Please Enter Email Address.</b></font>';
361             echo'<script>window.location.href="ForgotPass.php"</script>';
362         }
363
364         if ($_REQUEST['username_stf_email'] != '' && $_REQUEST['pass'] != '') {
365             $stf_info = DBGet(DBQuery('SELECT s.* FROM staff s,login_authentication la WHERE la.USER_ID=s.STAFF_ID AND la.PASSWORD=\'' .
md5($_REQUEST['pass']) . '\' AND s.EMAIL=\'' . $_REQUEST['username_stf_email'] . '\''));
366
367             if ($stf_info[1]['STAFF_ID'] == '') {
368                 $_SESSION['err_msg'] = '<font color="red" ><b>Incorrect login credential.</b></font>';
369                 echo'<script>window.location.href="ForgotPass.php"</script>';
370             } else {
371                 $get_uname = DBGet(DBQuery('SELECT USERNAME FROM login_authentication WHERE USER_ID=' . $stf_info[1]['STAFF_ID'] . ' AND
PROFILE_ID=' . $stf_info[1]['PROFILE_ID']));
372                 $_SESSION['fill_username'] = $get_uname[1]['USERNAME'];
373                 echo'<script>window.location.href="index.php"</script>';
374             }
375         }
376     }
```

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 Yuri Kramarz of Cisco Talos.

VULNERABILITY REPORTS

PREVIOUS REPORT TALOS-2020-1079

NEXT REPORT TALOS-2020-1095