Talos Vulnerability Report

TALOS-2020-1202

OpenClinic GA web portal SQL injection vulnerability in 'statistics/quickFile.jsp' page

APRIL 13, 2021

CVE NUMBER

CVE-2020-27226

Summary

An exploitable SQL injection vulnerability exists in ‘quickFile.jsp’ page of OpenClinic GA 5.173.3. A specially crafted HTTP request can lead to SQL injection. An attacker can make an authenticated HTTP request to trigger this vulnerability.

Tested Versions

OpenClinic GA 5.173.3

Product URLs

https://sourceforge.net/projects/open-clinic/

CVSSv3 Score

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

CWE

CWE-89 - Improper Neutralization of Special Elements used in an SQL Command (‘SQL Injection’)

Details

OpenClinic GA is an open source fully integrated hospital management solution.

The `PatientUID` parameter in `quickFile.jsp` page is vulnerable to authenticated SQL injection. The following request would trigger the vulnerability:

```
POST /openclinic/popup.jsp?Page=statistics/quickFile.jsp&PopupHeight=600&PopupWidth=1000 HTTP/1.1
Host: [IP]:10080
Content-Length: 160
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://[IP]:10080
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-
exchange;v=b3;q=0.9
Referer: http://[IP]:10080/openclinic/popup.jsp?Page=statistics/quickFile.jsp&PopupWidth=1000&PopupHeight=600
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: JSESSIONID=64BC25CBE4CE5E171D2A859C058194BA
Connection: close

PatientUID=
     <SQLINJECTION>&patientLastname=&patientFirstname=&patientDateOfBirth=&patientGender=&findPatient=Find&newPatient=&activePatientUID=
```

The above vulnerablity appears due to a lack of filtering applied in the `statistics/quickFile.jsp` source file and underlying `net.admin.AdminPerson` Java class when input parameters are used to create a search query for specific patients as seen below:

```
if(isFindPatient || isFindEncounter) {
    // kijken of de ingevoerde patiënt wel bestaat
    java.util.List persons = new java.util.Vector();
    if(!isNewPatient && (sPatientLastname.trim().length() > 0 || sPatientFirstname.trim().length() > 0 ||
sPatientDateOfBirth.trim().length() > 0 || sPatientUID.trim().length() > 0)) {
        if(isFindEncounter) {
                Connection ad_conn = MedwanQuery.getInstance().getAdminConnection();
            persons.add(AdminPerson.getAdminPerson(ad_conn, sPatientUID));
            ad_conn.close();
        }
        else {
            persons = AdminPerson.getAllPatients("", "", "", sPatientLastname, sPatientFirstname, sPatientDateOfBirth, sPatientUID,"");
        }
```

After call to the `getAllPatients`' function is made, the SQL query is created and executed as seen below:

```
  public static List getAllPatients(String simmatnew, String sArchiveFileCode, String snatreg, String sName, String sFirstname, String
sDateOfBirth, String sPersonID, String sDistrict) {
        PreparedStatement ps = null;
        ResultSet rs = null;

        List lResultList = new ArrayList();

        String sSQLSelect = " SELECT DISTINCT a.searchname, a.personid, a.immatnew, a.natreg, a.lastname, a.firstname, a.gender,
a.dateofbirth, a.pension";
        String sSQLFrom = " FROM AdminView a";
        String sSQLWhere = " 1=1 AND";

        if (simmatnew.trim().length() > 0)
        {
          sSQLWhere = sSQLWhere + " immatnew like '" + simmatnew + "%' AND";
        }

        Connection oc_conn = MedwanQuery.getInstance().getOpenclinicConnection();

        if (sArchiveFileCode.trim().length() > 0) {
          String lowerArchiverFileCode = ScreenHelper.getConfigParam("lowerCompare", "archiveFileCode", oc_conn);
          sSQLWhere = sSQLWhere + " " + lowerArchiverFileCode + " LIKE '" + sArchiveFileCode.toLowerCase() + "' AND";
        }

        if (snatreg.trim().length() > 0) {
          sSQLWhere = sSQLWhere + " natreg like '" + snatreg + "%' AND";
        }

        if (sPersonID.trim().length() > 0) {
          sSQLWhere = sSQLWhere + " a.personid = " + sPersonID + " AND";
        }

        if (sDistrict.trim().length() > 0) {
          sSQLFrom = sSQLFrom + ", AdminPrivate p";
          sSQLWhere = sSQLWhere + " p.personid = a.personid AND district = '" + sDistrict + "' AND";
        }
```

Timeline

2020-11-19 - Initial contact
2020-12-07 - 2nd contact; copy of advisories issued and vendor acknowledged receipt
2021-02-01 - 60 day follow up; no response
2021-03-09 - 90 day follow up; no response
2021-03-22 - Final notice
2021-04-13 - Public disclosure

CREDIT

Discovered by Yuri Kramarz of Cisco Talos.