Talos Vulnerability Report

TALOS-2020-1206

OpenClinic GA Web portal SQL injection vulnerability in 'manageServiceStocks.jsp' page

APRIL 13, 2021

CVE NUMBER

CVE-2020-27232

Summary

An exploitable SQL injection vulnerability exists in ‘manageServiceStocks.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.

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

```
POST /openclinic/main.do?Page=pharmacy/manageServiceStocks.jsp&ts=1603981647838 HTTP/1.1
Referer: http://[IP]:10080/openclinic/main.do?Page=pharmacy/manageServiceStocks.jsp&ts=1603981646619
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.102 Safari/537.36
Edge/18.18362
Accept-Encoding: gzip, deflate
Content-Length: 750
Host: [IP]:10080
Cookie: JSESSIONID=C3AEAECEA33285FCCE9E02171E8B3FA1
Connection: close

FindStockName=&FindServiceUid=



SQLINJECTION>&FindServiceName=&FindBegin=&FindEnd=&FindManagerUid=&FindManagerName=&FindDefaultSupplierUid=&FindDefault
SupplierNameNametEnd=EditManagerUid=4&EditManagerName=test&AuthorizedUserIdAdd=&AuthorizedUserNameAdd=&EditAuthorized
Users=&ReceivingUserIdAdd=&ReceicingUserNameAdd=&EditReceivingUsers=&DispensingUserIdAdd=&DispensingUserNameAdd=&Edit   t
DispensingUsers=&ValidationUserIdAdd=&ValidationUserNameAdd=&EditValidtionUsers=&EditDefaultSupplierUid=is &EditOrderPeriodIn
Months=1&EditNosync=1&EditHidden=1&EditValidateOutgoing=1&Action=find&EditStockUid=-1&DisplaySearchFields=false&DisplayActive
ServiceStocks=
```

The above vulnerability is triggered due to dynamic use of `FindServiceUid` paremter when the `find` action is invoked as seen below:

```
 //--- FIND ------------------------------------------------------------------------------------
if(sAction.startsWith("find")){
    displayActiveServiceStocks = false;
    displayEditFields = false;
    displayFoundRecords = true;

    if(sAction.equals("findShowOverview")){
        displaySearchFields = true;
    }

    // get data from form
    sFindStockName          = checkString(request.getParameter("FindStockName"));
    sFindBegin              = checkString(request.getParameter("FindBegin"));
    sFindEnd                = checkString(request.getParameter("FindEnd"));
    sFindManagerUid         = checkString(request.getParameter("FindManagerUid"));
    sFindServiceUid         = checkString(request.getParameter("FindServiceUid"));
    sFindDefaultSupplierUid = checkString(request.getParameter("FindDefaultSupplierUid"));

    Vector serviceStocks = ServiceStock.find(sFindStockName,sFindServiceUid,sFindBegin,sFindEnd,
                                             sFindManagerUid,sFindDefaultSupplierUid,"OC_STOCK_NAME","ASC");
    stocksHtml =
objectsToHtml(serviceStocks,sWebLanguage,activeUser,request.getParameter("showhidden"),bHasActivePrescriptions,request.getParameter("showuna
uthorized"));
    foundStockCount = serviceStocks.size();
```

In the function call to `ServiceStock.find`, an SQL query is created and, eventually, executed when the function below is called in the `be.openclinic.pharmacy.SeviceStock` class:

```
public static Vector find(String sFindStockName, String sFindServiceUid, String sFindBegin, String sFindEnd, String sFindManagerUid,
String sFindDefaultSupplierUid, String sSortCol, String sSortDir)
  {
        Vector foundObjects = new Vector();
        PreparedStatement ps = null;
        ResultSet rs = null;

        Connection oc_conn = MedwanQuery.getInstance().getOpenclinicConnection();
        try {
          String sSelect = "SELECT OC_STOCK_SERVERID, OC_STOCK_OBJECTID FROM OC_SERVICESTOCKS";

          if ((sFindStockName.length() > 0) || (sFindServiceUid.length() > 0) || (sFindBegin.length() > 0) ||
                (sFindEnd.length() > 0) || (sFindManagerUid.length() > 0) || (sFindDefaultSupplierUid.length() > 0)) {
                sSelect = sSelect + " WHERE ";
                if (sFindServiceUid.length() > 0)
                {
                  Vector childIds = Service.getChildIds(sFindServiceUid);
                  childIds.add(sFindServiceUid);

                  String sChildIds = ScreenHelper.tokenizeVector(childIds, ",", "'");
                  if (sChildIds.length() > 0) {
                        sSelect = sSelect + "OC_STOCK_SERVICEUID IN (" + sChildIds + ") AND ";
                  }
                  else {
                        sSelect = sSelect + "OC_STOCK_SERVICEUID IN ('') 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.

VULNERABILITY REPORTS

PREVIOUS REPORT TALOS-2020-1207

NEXT REPORT TALOS-2020-1205