

Zabbix Webhook Alerts

- Background
- Setup required going Live
- API Endpoint
- JSON Payload
- Files
- Logic
- Test Setup

Background

This webhook is a method of sending and receiving a NMS alert for a piece of equipment going down on site. The details of the alert will be recorded in snap and a support ticket created in CAS.

Setup required going Live

- On Zabbix side
 - point the API endpoint to the prod endpoint
 - configure Webhook Media Type (CAS - TEST - maybe this would not be name TEST in prod) to use Zabbix dynamic parameters (this is probably a task for David)

The screenshot shows the Zabbix web interface for configuring a media type. The left sidebar contains navigation links: Monitoring, Services, Inventory, Reports, Configuration, and Administration (with sub-links for General, Proxies, Authentication, User groups, User roles, Users, Media types, Scripts, and Queue). The main content area is titled 'Media types' and has tabs for 'Media type', 'Message templates', and 'Options'. The 'Media type' tab is active, showing the configuration for a media type named 'CAS - TEST' of type 'Webhook'. A table lists parameters with their values, and a script field is at the bottom. Red annotations highlight specific areas: a box around the parameter values with a note about hard-coded test values, an arrow pointing to the 'device_status' parameter, and an arrow pointing to the script field with a note about passing parameters to a JavaScript script. A 'Click to view or edit' button is visible next to the script field.

Name	Value	Action
api_token	djkdjaskfjaskdjkfjaskdjsa	Remove
api_url	https://snapx-us1-stg.safetynetacc	Remove
device_id		Remove
device_status	0	Remove
device_type	access_point	Remove
event_id	3543453	Remove
event_severity	5	Remove
host_id	72465	Remove
HTTPProxy		Remove
item_id	43434	Remove
mac_address	00 50 E8 04 3E 40	Remove
serial	043E40_59004394	Remove
site_id	6661	Remove
timestamp	1721982530	Remove

* Script
// "api_url" : "https://snapx-us1-stg.safetynetaccess.com..."

* Timeout
30s

Process tags ☒

Include event menu entry ☐

- the trigger should be setup in Zabbix in order to call the API when an alert is triggered (this should be done by David)
- On Snap side
 - in order to avoid having to configure this site by site, we need to add a way to get the CAS API property_id which is mapped to a snap ID (or vendor_property_id) and store it in the site meta (key: property_id / value: <id in CAS API>); currently there is only one

endpoint but it returns all the properties info and it is quite slow (**Note:** this requirement will go away once we can get the property_id from CAS API - email sent by Terry)

- Need confirmation of the CAS statuses for 'Open' and 'Closed' (16='Closed'?)

API Endpoint

<https://snapx-us1-stg.safetynetaccess.com/api/v1/nms/siteid/eventid/alert>

JSON Payload

When an alert is triggered the JSON payload that hits the endpoint will resemble the following

```
1  {
2    "timestamp" : "1722436864",
3    "device_id" : "296",
4    "device_type" : "network_switch",
5    "device_status" : "0",
6    "event_severity": "5",
7    "event_id" : "2566335",
8    "host_id" : "34342",
9    "item_id" : "12345",
10   "mac_address" : "00 50 E8 04 3E 40",
11   "serial" : "043E40_59004394",
12   "site_id" : "7993"
13 }
```

Files

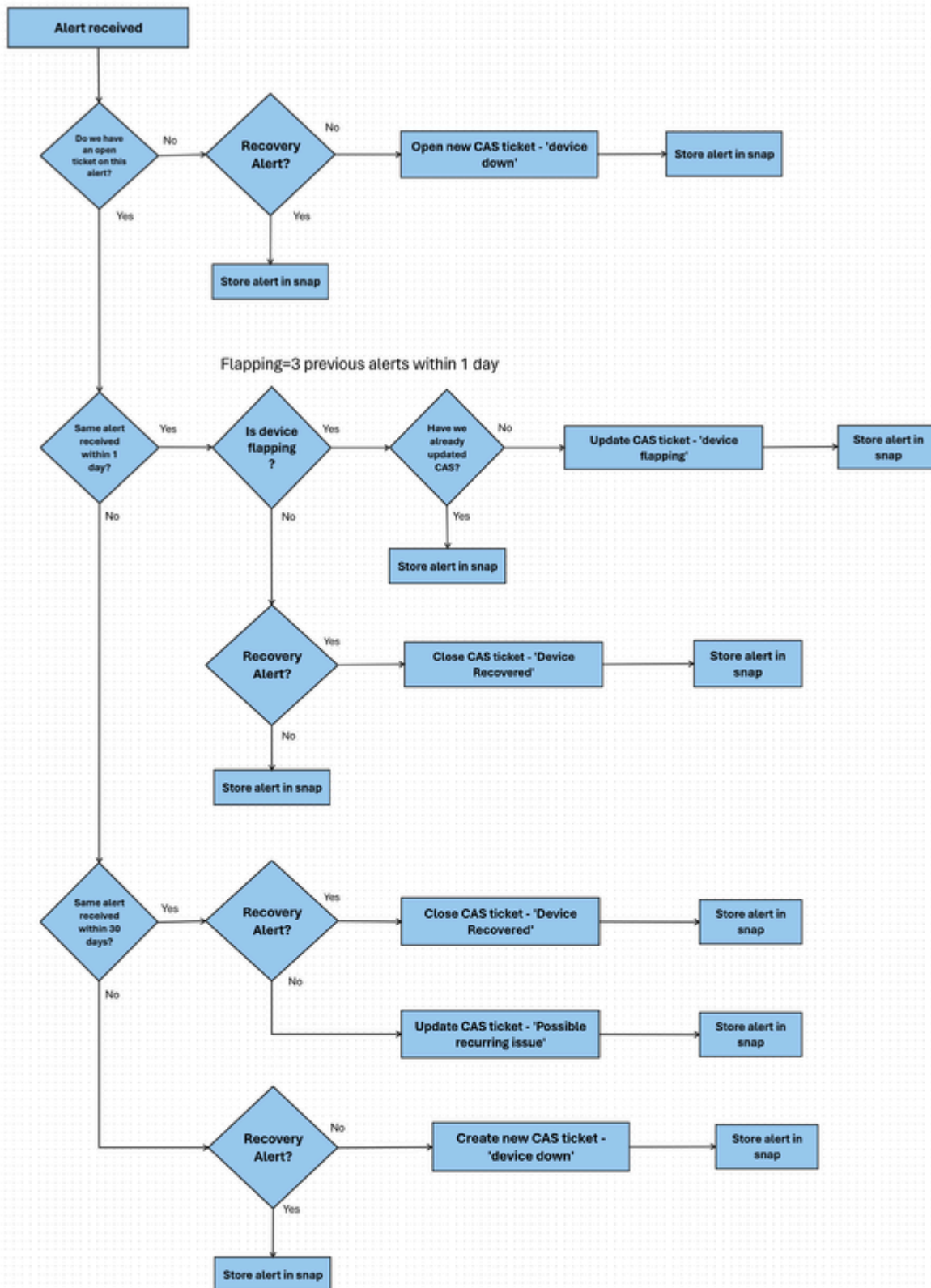
The code that receives and processes the webhook alert data can be found here:

app/Api/Controller/NmsMiddlewareController.php

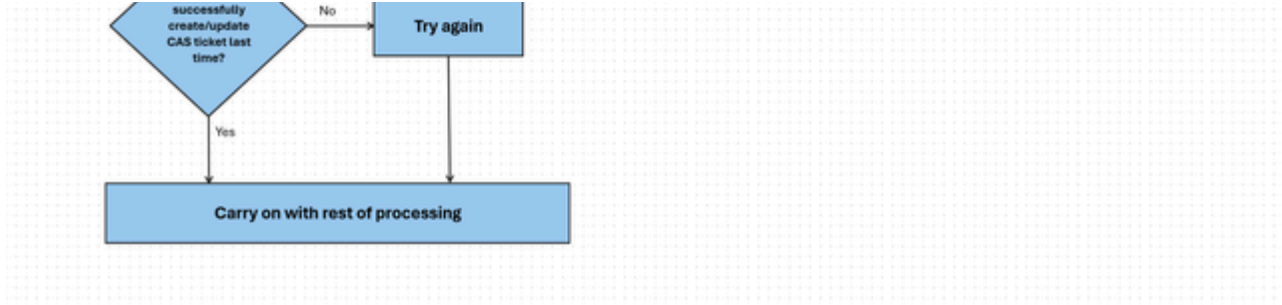
The Javascript that sends the request can be found further down this page

Logic

The alert is processed at the endpoint using the following logic



Embeded logic: If cas was unreachable last time an attempt was made to create or update a ticket, try again.



Test Setup

On the Zabbix side a webhook alert can be simulated by following these steps, this was used for testing the functionality e2e in staging:

1. Open the Zabbix control panel and navigate to Administration->Media Types->CAS - TEST

ZABBIX

Safety NetAccess - FE01

Monitoring

Services

Inventory

Reports

Configuration

Administration

General

Proxies

Authentication

User groups

User roles

Users

Media types

Scripts

Queue

Media types

<input type="checkbox"/>	Name ▲	Type	Status	Used in actions
<input type="checkbox"/>	Brevis.one	Webhook	Enabled	
<input type="checkbox"/>	CAS - TEST	Webhook	Enabled	CAS testing, Report problems to Zabbix administrators
<input type="checkbox"/>	Discord	Webhook	Enabled	
<input type="checkbox"/>	Email	Email	Enabled	
<input type="checkbox"/>	Email (HTML)	Email	Enabled	Alert Ops team
<input type="checkbox"/>	Express.ms	Webhook	Enabled	
<input type="checkbox"/>	Github	Webhook	Enabled	
<input type="checkbox"/>	Github_bdoymie	Webhook	Enabled	Report problems to Zabbix administrators
<input type="checkbox"/>	GLPi	Webhook	Enabled	
<input type="checkbox"/>	iLert	Webhook	Enabled	
<input type="checkbox"/>	iTop	Webhook	Enabled	
<input type="checkbox"/>	Jira	Webhook	Enabled	
<input type="checkbox"/>	Jira ServiceDesk	Webhook	Enabled	
<input type="checkbox"/>	Jira with CustomFields	Webhook	Enabled	
<input type="checkbox"/>	ManageEngine ServiceDesk	Webhook	Enabled	

ZABBIX << Media types

Safety NetAccess - FE01

Media type Message templates Options

* Name CAS - TEST

Type Webhook

Parameters

Name	Value	Action
api_token	djdkaskfjaskdfjaskdfjsa	Remove
api_url	https://snapx-us1-stg.safetynetacc	Remove
device_id		Remove
device_status	0	Remove
device_type	access_point	Remove
event_id	3543453	Remove
event_severity	5	Remove
host_id	72465	Remove
HTTPProxy		Remove
item_id	43434	Remove
mac_address	00 50 E8 04 3E 40	Remove
serial	043E40_59004394	Remove
site_id	8861	Remove
timestamp	1721982530	Remove

Add

* Script // "api_url" : "https://snapx-us1-stg.safetynetaccess.com..."

* Timeout 30s

Process tags ☒

Include event menu entry ☐

Click to view or edit

These hard coded test values will need hooked up to the Zabbix dynamic ones when ready to go live

The Parameters are passed in to the js script which is responsible for validating those params and making the request to the api endpoint with the params as a json payload in the body

2. The Javascript responsible for validating the params and making the request to the api endpoint

```

1 // "api_url" : "https://snapx-us1-stg.safetynetaccess.com/api/v1/nms"
2
3 var EventTicket = {
4
5     setParams: function (params) {
6         if (typeof params !== 'object') {
7             return;
8         }
9         EventTicket.params = params;
10    },
11
12
13    urlSetFormat: function (params) {
14
15        var url = params.api_url;
16        if (typeof url === 'string' && !url.endsWith('/')) {
17            url += '/';
18        }
19        url += params.site_id + '/' + params.event_id + '/alert';
20        if (url.indexOf('http://') === -1 && url.indexOf('https://') === -1) {
21            url = 'https://' + url;
22        }
23        return url;
24    },
25
26
27    getData: function(params) {
28
29        var keysToInclude = [
30            'timestamp',
31            'site_id',
32            'event_id',

```

```

33         'device_id',
34         'device_status',
35         'event_severity',
36         'device_type',
37         'host_id',
38         'item_id',
39         'mac_address',
40         'serial'
41     ];
42
43     var data = {};
44     for (var key in params) {
45         if(keysToInclude.indexOf(key) > -1) {
46             data[key] = params[key];
47         }
48     }
49     return data;
50 },
51
52
53     request: function (url, data, params) {
54
55         var response,
56         request = new HttpRequest();
57         request.addHeader('Content-type: application/json');
58         request.addHeader('Accept: application/json');
59         request.addHeader('Authorization: token ' + params.api_token);
60
61         if (typeof data !== 'undefined') {
62             data = JSON.stringify(data);
63         }
64
65         Zabbix.log(4, '[ EventTicket Webhook ] Sending request: ' + url + ((typeof data === 'string') ? ('\n'
+ data) : ''));
66         response = request.post(url, data);
67         Zabbix.log(4, '[ EventTicket Webhook ] Received response with status code: ' + request.getStatus() +
'\n' + response);
68
69         if (response !== null) {
70             try {
71                 response = JSON.parse(response);
72             }
73             catch (error) {
74                 Zabbix.log(4, '[ EventTicket Webhook ] Failed to parse response from: ' + url);
75                 response = null;
76             }
77         }
78
79         if (typeof response !== 'object') {
80             throw 'Failed to process response received from ' + url + '. Check debug log for more
information.';
81         }
82
83         if (request.getStatus() < 200 || request.getStatus() >= 300) {
84             var message = 'Request failed with status code ' + request.getStatus();
85             if (response.message) {
86                 message += ': ' + response.message;
87             }

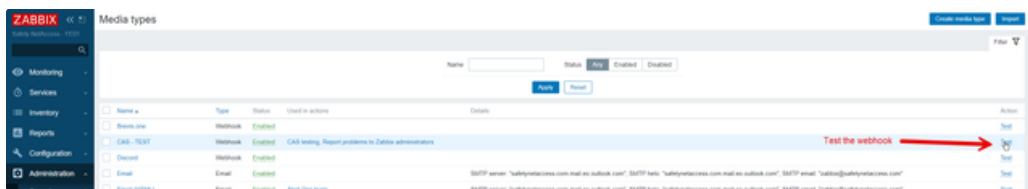
```

```

88         throw message + ' Check debug log for more information.';
89     }
90     return response;
91 }
92 };
93
94
95 try {
96     var params = JSON.parse(value),
97         ticket = {},
98         required_params = [
99             'api_url',
100             'api_token',
101             'timestamp',
102             'site_id',
103             'event_id',
104             'item_id',
105             'host_id',
106             'mac_address',
107             'device_status',
108             'event_severity',
109             'device_type',
110         ];
111
112     Object.keys(params).forEach(function (key) {
113         if (required_params.indexOf(key) !== -1 && params[key] === '') {
114             throw 'Parameter "' + key + '" can\'t be empty.';
115         }
116     });
117
118     url = EventTicket.urlSetFormat(params);
119
120     // url = params.api_url // TEST
121
122     data = EventTicket.getData(params);
123
124     var response = EventTicket.request(url, data, params);
125
126
127     Zabbix.log(4, '[ EventTicket Webhook ] Result: ' + JSON.stringify(response));
128     return JSON.stringify(response);
129 }
130 catch (error) {
131     Zabbix.log(4, '[ EventTicket Webhook ] ERROR: ' + error);
132     throw 'Sending failed: ' + error;
133 }

```

3. To test the webhook functionality



Test media type "CAS - TEST"

Media type test successful.

api_token: djlkfjaskfjasdlkfjasldfjsa

api_url: https://snapx-us1-stg.safetynetaccess.com/api/v1/nms

device_id:

device_status: 0

device_type: access_point

event_id: 3543453

event_severity: 5

host_id: 72465

HTTPProxy:

item_id: 43434

mac_address: 00 50 E8 04 3E 40

serial: 043E40_59004394

site_id: 6661

timestamp: 1722415774

Response: { "success": true }

Response type: JSON

[Open log](#)

View the logs generated by the script

Test Cancel

4. Expected log entries on success

Media type test log

```
00:00:00.000 [Debug] [ EventTicket Webhook ] Sending request: https://snapx-us1-stg.safetynetaccess.com/api/v1/nms/6661/3543453/index ("device_
00:00:00.056 [Debug] [ EventTicket Webhook ] Received response with status code: 200 ("success":true)
00:00:00.057 [Debug] [ EventTicket Webhook ] Result: {"success":true}
```

Time elapsed: 57ms

Ok

5. Example of an unsuccessful response

Response: { "success": false, "error": "A current timestamp is required in webhook data" }

6. The nms_alerts table will hold a record of all the received alerts and the actions taken on them (e.g. tickets created/updated in CAS)

Result: 1 CAS Ticket ID

Enter a SQL expression. Filter results, e.g. columns: name=10

#	id	site_code	site_name	device_status	ticket_status	device_type	device_flapping	action	notes	event_date
1	13785658	554303	42434	0	Opened	access_point	[N/A]	Ticket Created on CAS	Access_point ID:1048928 is DOWN	2024-07-01 12:32:41:000
2	13785658	554303	42434	0	Opened	access_point	[N/A]		Access_point ID:1048928 is DOWN, Possible Flapping Device/Port, an alert has already been received within 1 day	2024-07-01 12:32:41:000
3	13785658	554303	42434	0	Opened	access_point	[N/A]		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
4	13785658	554303	42434	0	Updated	access_point	1	Ticket updated on CAS	Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
5	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
6	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
7	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
8	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
9	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
10	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
11	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
12	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000
13	13785658	554303	42434	0	Updated	access_point	1		Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day	2024-07-01 12:32:41:000

7. An example of a CAS ticket created when an alert received and then updated when multiple alerts received on the same device indicating a flapping device

```

1  {
2      "ticket_id": "13785658",
3      "ticket_date_time": "7/30/2024 4:32:50",
4      "site_code": "6661",
5      "hotel_name": "SNA Training Video",
6      "guest_user_name": "",
7      "guest_name": "",
8      "last_name": "",
9      "room": "",
10     "contact_no": "",
11     "email": "",
12     "issue": "SNAP Alert",
13     "solution": "SNAP Alert",
14     "os": "N/A",
15     "server_status": "Down",
16     "isp_status": "Down",
17     "created_by": "Snap API",
18     "assigned_to": "SNA Tier 2",
19     "ip_address": "",
20     "mac_address": "",
21     "priority": "Low",
22     "status": "Resolved (Close)",
23     "recording": "No recording available yet.",
24     "media_type": "",
25     "severity_level": "",
26     "notes": [
27         {
28             "created_by": "Snap API",
29             "assigned_to": "SNA_ONMS",
30             "call_type": "No Call",
31             "caller_type": "Other",
32             "start_time": "7/30/2024 4:32:50",
33             "end_time": "7/30/2024 4:32:50",
34             "total_time": "00:00:00",
35             "content": "Access_point ID:1048928 is DOWN",
36             "call_date_time": "",
37             "call_duration": "",
38             "call_time_to_answer": "",
39             "call_ani": "",
40             "call_queue_name": ""
41         },
42         {
43             "created_by": "SNA Tier 2",
44             "assigned_to": "SNA Tier 2",
45             "call_type": "No Call",
46             "caller_type": "Other",
47             "start_time": "7/30/2024 4:33:05",

```

```
48         "end_time": "7/30/2024 4:33:05",
49         "total_time": "00:00:00",
50         "content": "Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been
received within 1 day",
51         "call_date_time": "",
52         "call_duration": "",
53         "call_time_to_answer": "",
54         "call_ani": "",
55         "call_queue_name": ""
56     },
57 ]
58 }
```