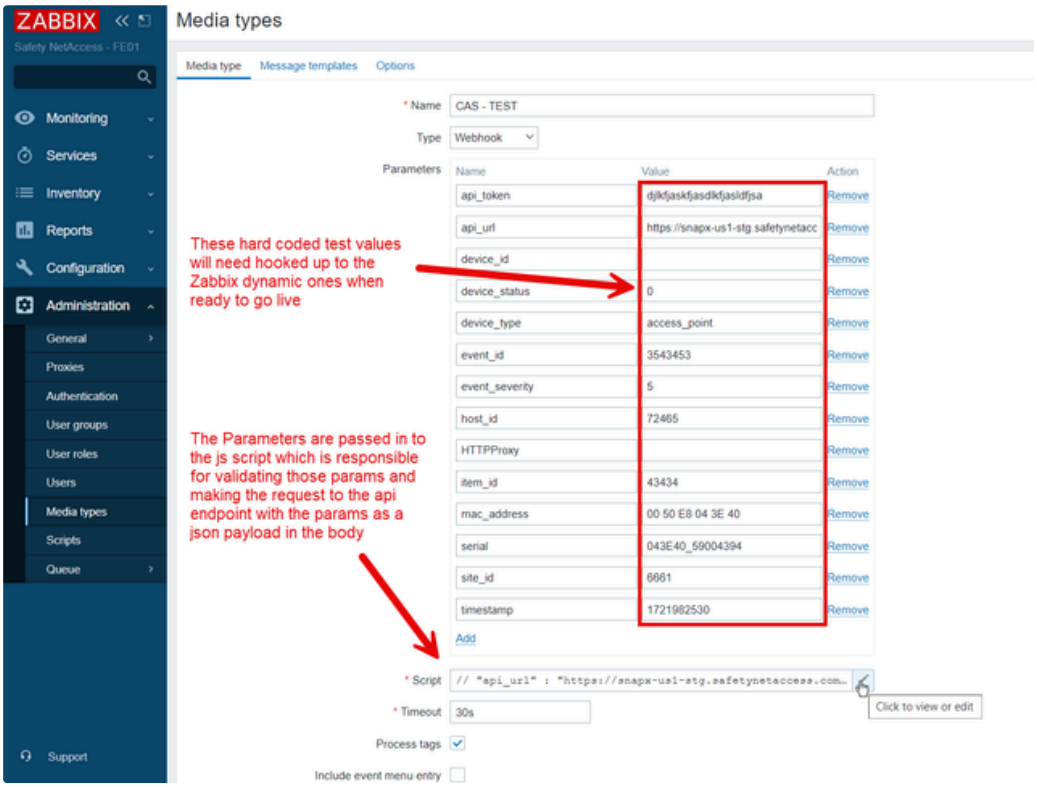
Zabbix Webhook Alerts

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 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

```
{
   "timestamp" : "1722436864",
   "device_id" : "296",
   "device_type" : "network_switch",
   "device_status" : "0",
   "event_severity": "5",
   "event_id" : "2566335",
   "host_id" : "34342",
   "item_id" : "12345",
   "mac_address" : "00 50 E8 04 3E 40",
   "serial" : "043E40_59004394",
   "site_id" : "7993"
}
```

Files

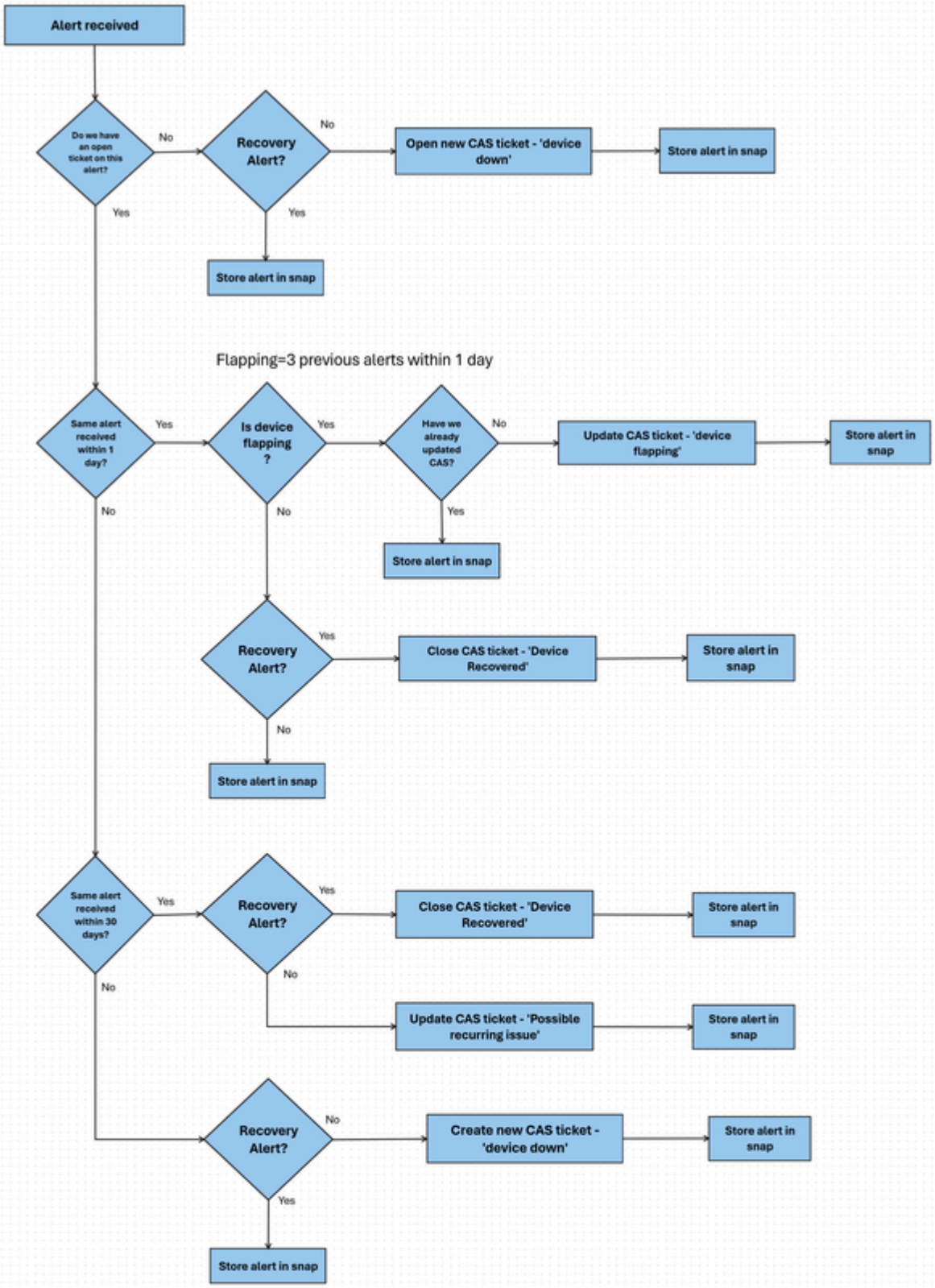
The code that receives and processes the webhook alert data can be found here:

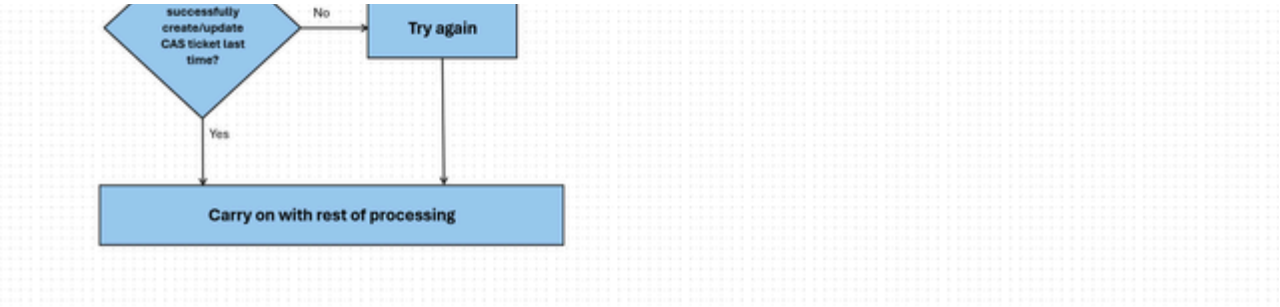
app/Api/Controller/NmsMiddlewareController.php

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

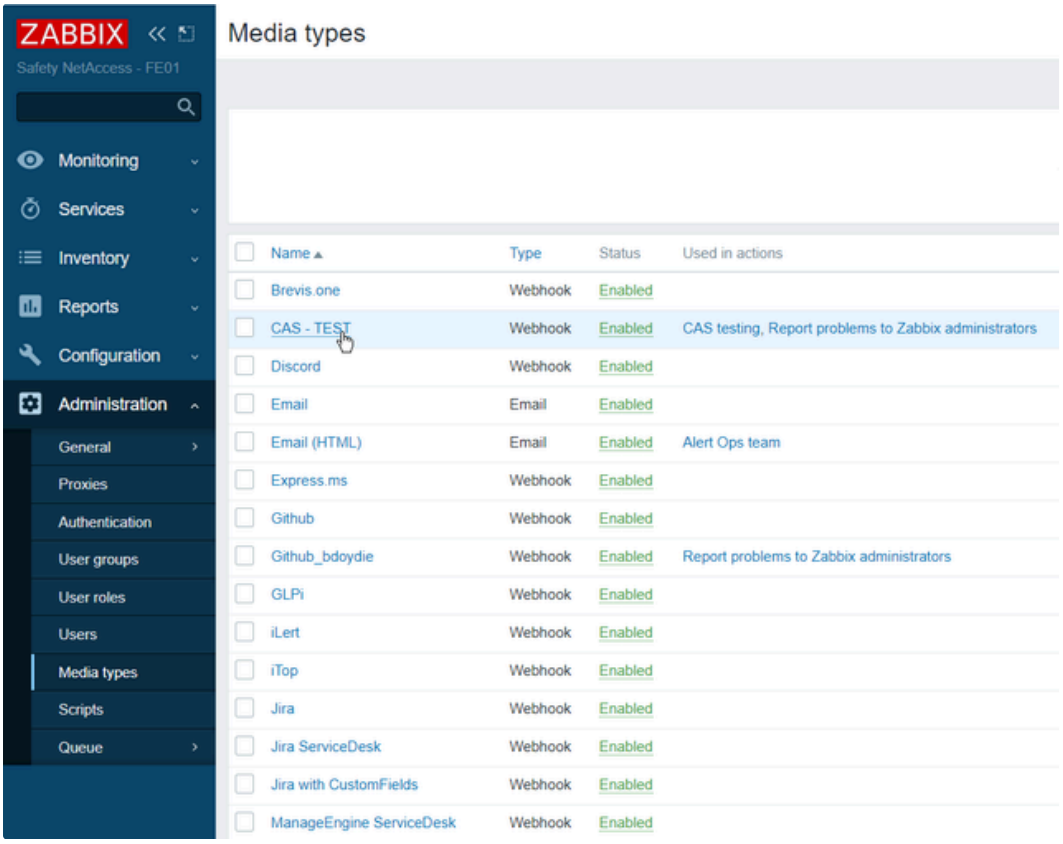

Alert received
Do we have an open ticket on this alert?
No
Recovery Alert?
No
Open new CAS ticket - 'device down'
Store alert in snap
Yes
Yes
Store alert in snap
Flapping=3 previous alerts within 1 day
Same alert received within 1 day?
Yes
Is device flapping ?
Yes
Have we already updated CAS?
No
Update CAS ticket - 'device flapping'
Store alert in snap
No
No
Yes
Store alert in snap
Recovery Alert?
Yes
Close CAS ticket - 'Device Recovered'
Store alert in snap
No
Store alert in snap
Same alert received within 30 days?
Yes
Recovery Alert?
Yes
Close CAS ticket - 'Device Recovered'
Store alert in snap
No
No
Update CAS ticket - 'Possible recurring issue'
Store alert in snap
Recovery Alert?
No
Create new CAS ticket - 'device down'
Store alert in snap
Yes
Store alert in snap
Alert received
Embeded logic: If cas was unreachable last time an attempt was made to create or update a ticket, try again.
Did we

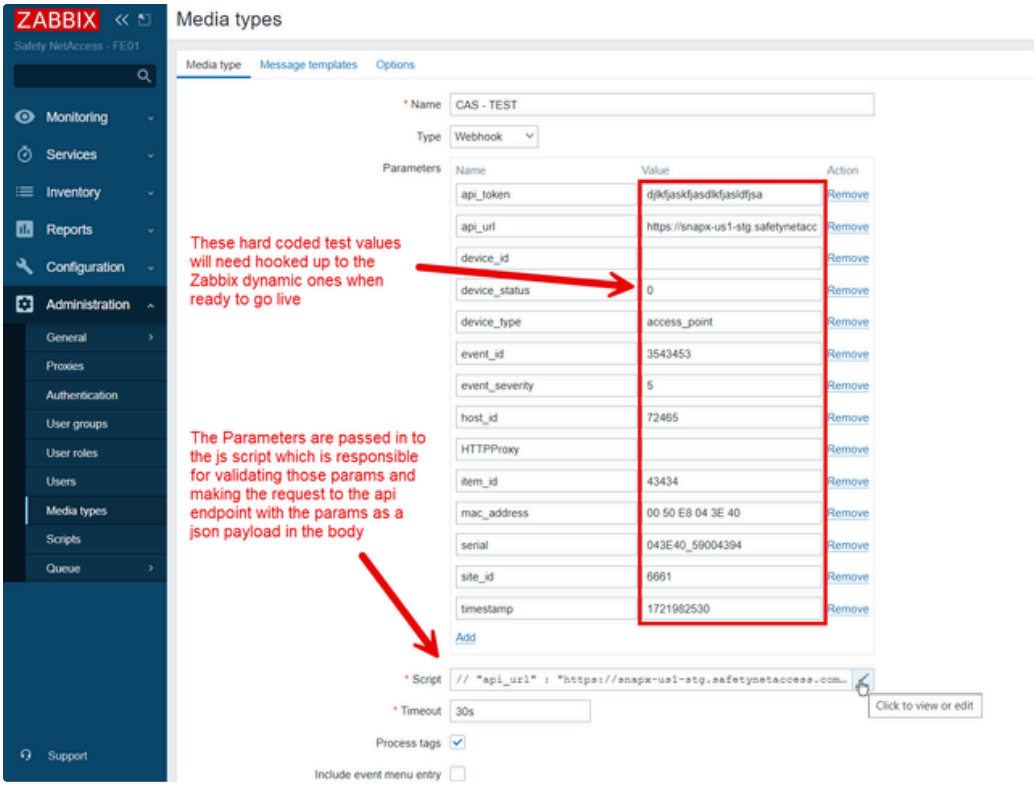


Test Setup

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





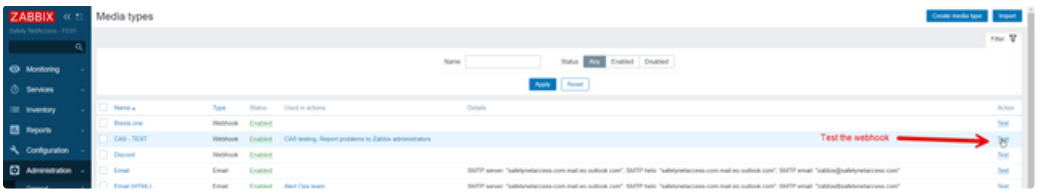
2. The Javascript responsible for validating the params and making the request to the api endpoint

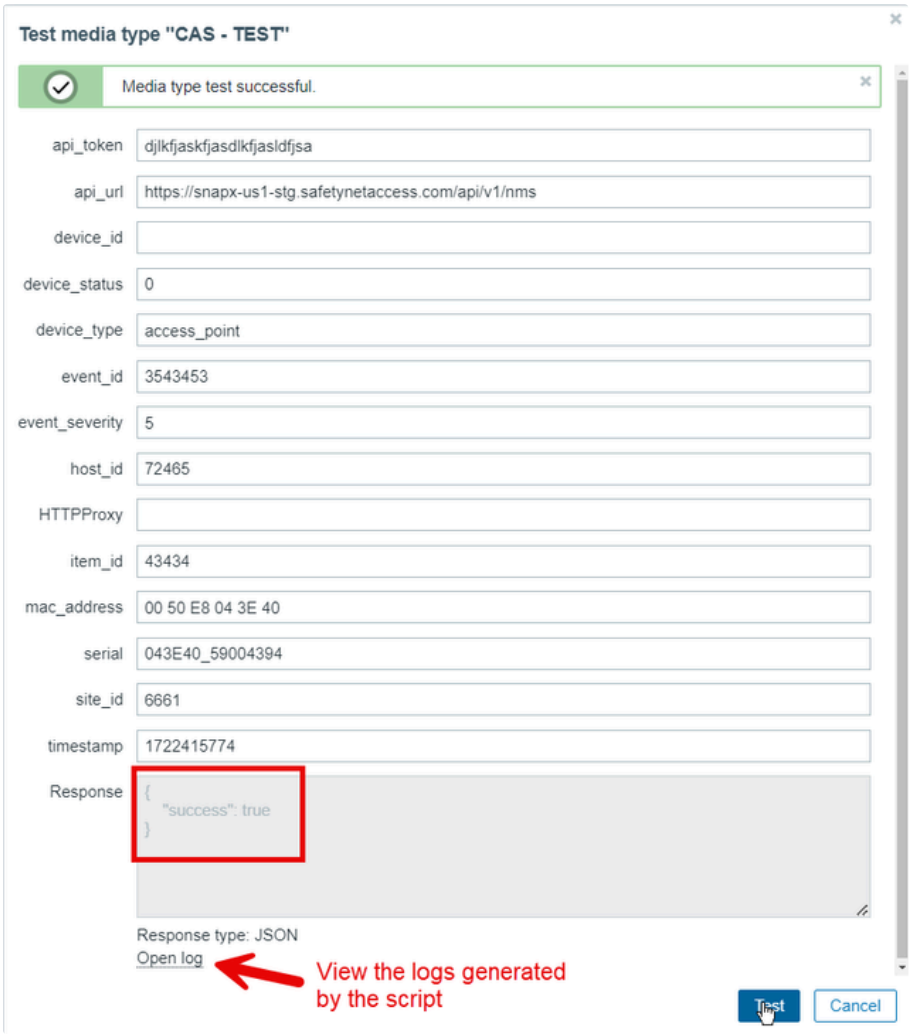
```
// "api_url" : "https://snapx-us1-stg.safetynetaccess.com/api/v1/nms"

var EventTicket = {

    setParams: function (params) {
        if (typeof params !== 'object') {
            return;
        }
        EventTicket.params = params;
    },


    urlSetFormat: function (params) {

        var url = params.api_url;
        if (typeof url === 'string' && !url.endsWith('/')) {
            url += '/';
        }
        url += params.site_id + '/' + params.event_id + '/alert';
        if (url.indexOf('http://') === -1 && url.indexOf('https://') === -1) {
            url = 'https://' + url;
        }
        return url;
    },


    getData: function(params) {

        var keysToInclude = [
            'timestamp',
            'site_id',
            'event_id',
```

```
            'device_id',
            'device_status',
            'event_severity',
            'device_type',
            'host_id',
            'item_id',
            'mac_address',
            'serial'
        ];

        var data = {};
        for (var key in params) {
             if(keysToInclude.indexOf(key) > -1) {
                data[key] = params[key];
            }
        }
        return data;
    },


    request: function (url, data, params) {

        var response,
        request = new HttpRequest();
        request.addHeader('Content-type: application/json');
        request.addHeader('Accept: application/json');
        request.addHeader('Authorization: token ' + params.api_token);

        if (typeof data !== 'undefined') {
            data = JSON.stringify(data);
        }

        Zabbix.log(4, '[ EventTicket Webhook ] Sending request: ' + url + ((typeof data === 'string') ? ('\n' + data) : ''));
        response = request.post(url, data);
        Zabbix.log(4, '[ EventTicket Webhook ] Received response with status code: ' + request.getStatus() + '\n' + response);

        if (response !== null) {
            try {
                response = JSON.parse(response);
            }
            catch (error) {
                Zabbix.log(4, '[ EventTicket Webhook ] Failed to parse response from: ' + url);
                response = null;
            }
        }

        if (typeof response !== 'object') {
            throw 'Failed to process response received from ' + url + '. Check debug log for more information.';
        }

        if (request.getStatus() < 200 || request.getStatus() >= 300) {
            var message = 'Request failed with status code ' + request.getStatus();
            if (response.message) {
                message += ': ' + response.message;
            }
```

```
            throw message + ' Check debug log for more information.';
        }
        return response;
    }
};


try {
    var params = JSON.parse(value),
        ticket = {},
        required_params = [
            'api_url',
            'api_token',
            'timestamp',
            'site_id',
            'event_id',
            'item_id',
            'host_id',
            'mac_address',
            'device_status',
            'event_severity',
            'device_type',
        ];

        Object.keys(params).forEach(function (key) {
            if (required_params.indexOf(key) !== -1 && params[key] === '') {
                throw 'Parameter "' + key + '" can\'t be empty.';
            }
        });

         url = EventTicket.urlSetFormat(params);

        // url = params.api_url // TEST

        data = EventTicket.getData(params);

        var response = EventTicket.request(url, data, params);


    Zabbix.log(4, '[ EventTicket Webhook ] Result: ' + JSON.stringify(response));
    return JSON.stringify(response);
}
catch (error) {
    Zabbix.log(4, '[ EventTicket Webhook ] ERROR: ' + error);
    throw 'Sending failed: ' + error;
}
```

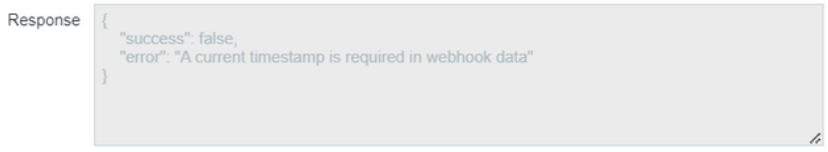
3. To test the webhook functionality



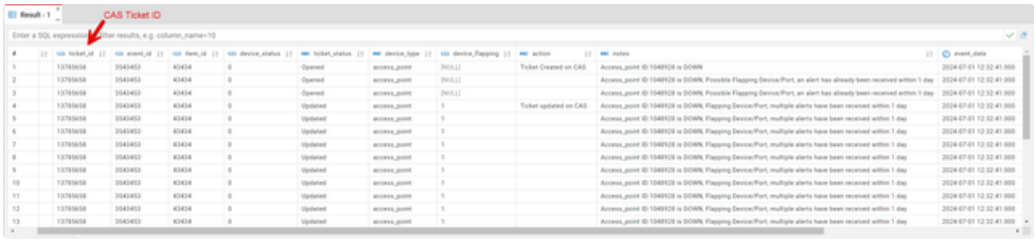


4. Expected log entries on success

5. Example of an unsuccessful response



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)



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


```
{
    "ticket_id": "13785658",
    "ticket_date_time": "7/30/2024 4:32:50",
    "site_code": "6661",
    "hotel_name": "SNA Training Video",
    "guest_user_name": "",
    "guest_name": "",
    "last_name": "",
    "room": "",
    "contact_no": "",
    "email": "",
    "issue": "SNAP Alert",
    "solution": "SNAP Alert",
    "os": "N/A",
    "server_status": "Down",
    "isp_status": "Down",
    "created_by": "Snap API",
    "assigned_to": "SNA Tier 2",
    "ip_address": "",
    "mac_address": "",
    "priority": "Low",
    "status": "Resolved (Close)",
    "recording": "No recording available yet.",
    "media_type": "",
    "severity_level": "",
    "notes": [
        {
            "created_by": "Snap API",
            "assigned_to": "SNA_ONMS",
            "call_type": "No Call",
            "caller_type": "Other",
            "start_time": "7/30/2024 4:32:50",
            "end_time": "7/30/2024 4:32:50",
            "total_time": "00:00:00",
            "content": "Access_point ID:1048928 is DOWN",
            "call_date_time": "",
            "call_duration": "",
            "call_time_to_answer": "",
            "call_ani": "",
            "call_queue_name": ""
        },
        {
            "created_by": "SNA Tier 2",
            "assigned_to": "SNA Tier 2",
            "call_type": "No Call",
            "caller_type": "Other",
            "start_time": "7/30/2024 4:33:05",
```

```
            "end_time": "7/30/2024 4:33:05",
            "total_time": "00:00:00",
            "content": "Access_point ID:1048928 is DOWN, Flapping Device/Port, multiple alerts have been received within 1 day",
            "call_date_time": "",
            "call_duration": "",
            "call_time_to_answer": "",
            "call_ani": "",
            "call_queue_name": ""
        },
    ]
}
```