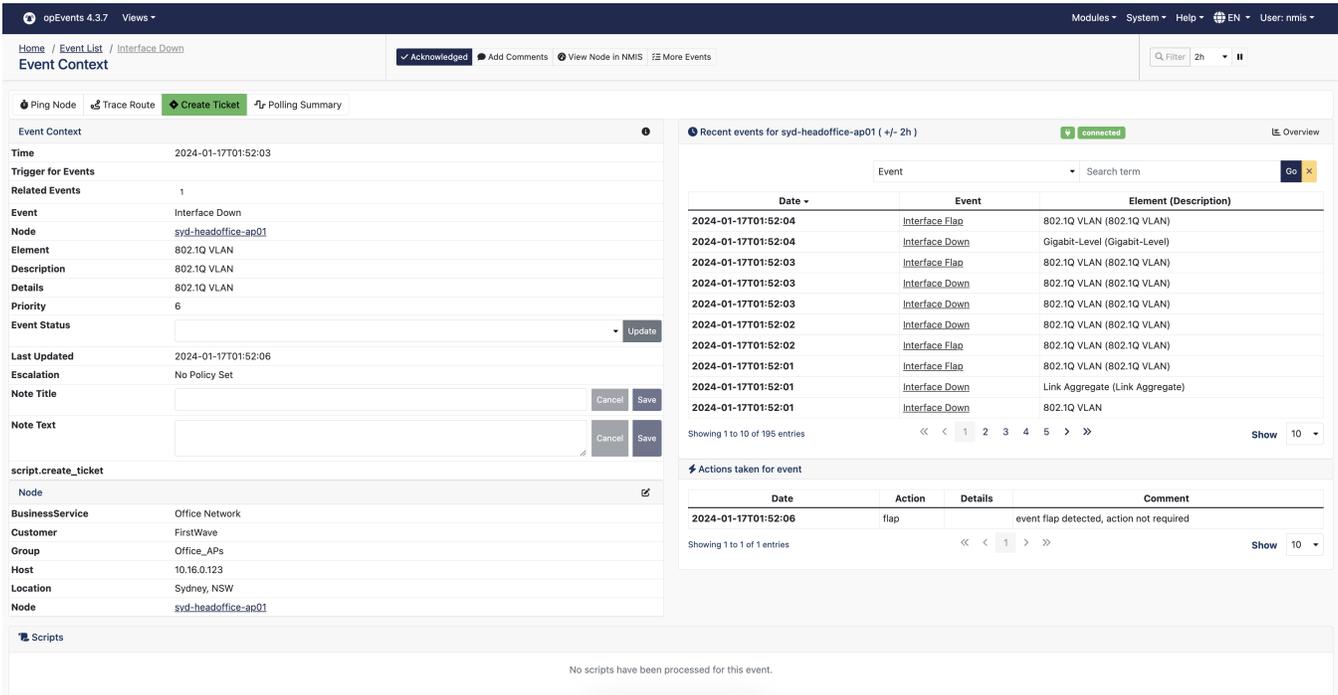


opEvents Programmable Button Actions

 Available in opEvents-3.2.2 and opEvents-2.6.1



The screenshot shows the opEvents 4.3.7 interface. The top navigation bar includes 'Home / Event List / Interface Down', 'Event Context', and user information 'User: nmis'. The main content area is divided into several sections:

- Event Context:** Displays details for an event on 2024-01-17T01:52:03. It includes fields for Time, Trigger for Events, Related Events (1), Event (Interface Down), Node (syd-headoffice-ap01), Element (802.1Q VLAN), Description (802.1Q VLAN), Details (802.1Q VLAN), Priority (6), Event Status, Last Updated (2024-01-17T01:52:06), Escalation (No Policy Set), Note Title, Note Text, and a script named 'script.create_ticket' with associated metadata like BusinessService (Office Network), Customer (FirstWave), Group (Office_APs), Host (10.16.0.123), Location (Sydney, NSW), and Node (syd-headoffice-ap01).
- Recent events for syd-headoffice-ap01 (+/- 2h):** A table listing recent events with columns for Date, Event, and Element (Description). The table shows multiple entries for 'Interface Flap' and 'Interface Down' on 802.1Q VLAN (802.1Q VLAN) and 'Link Aggregate (Link Aggregate)'. A search bar and pagination controls are present.
- Actions taken for event:** A table showing actions for the event, with one entry for 'flap' on 2024-01-17T01:52:06, with details 'event flap detected, action not required'.
- Scripts:** A section indicating 'No scripts have been processed for this event.'

opEvents programmable buttons allow scripts to be run against events to give operators greater flexibility in the use of opEvents in troubleshooting and triaging of events.

It uses the same pipeline as scripts in EventActions but now operators have the ability to manually kick off an action for an event.

Example use Cases

Create a Jira ticket

Configuration

Create a file in `omk/conf/table_schemas/opEvents_action-buttons.json`

This must be valid JSON schema or the buttons will fail to render. You should see an error in `opEvents.log` if this is the case.

```
[
  {
    "description": "Example Events Button Action",
    "label": "Ping Node",
    "fa_icon": "fas fa-table-tennis",
    "script": "ping_node",
    "tags": ["ping"]
  }
]
```

Add a policy in `omk/conf/EventActions.json|nmis` that triggers `show_button.tag()`

EventActions.json

```
"policy": {
  "5": {
    "IF": "event.any",
    "THEN": "show_button.ping()",
    "BREAK": "true"
  },
}
```

EventActions.nmis

```
%hash = (
  'policy' => {
    '5' => {
      IF => 'event.any',
      THEN => 'show_button.ping()',
      BREAK => 'true'
    },
  },
);
```

These are the supported keys and how the change operation and look of the button.

Key	Type	Required	Description
script	string	Yes	Name of the script defined in EventActions.json
label	string	Yes	Label which the button will display to the user
description	string	optional	Tool-tip help text to be displayed when you mouse over the button
tags	array [string]	optional	If no tags are defined the button will show on all events, if tags are defined the button will only show on events which have been tagged with <code>show_button.tag_name()</code>
run_once	boolean	optional	If set to true the button will look for script <code>script_name</code> key on the event, if found the button will disable itself. This allows manual actions to only be triggered once. Will not influence any defined EventActions.json operations.
fa_icon	string	optional	Icon to be displayed from the Font Awesome library shipped with opEvents example: "fas fa-table-tennis" Icons here https://fontawesome.com/icons?d=gallery
class	string	optional	Define a css class to colour the button, see Notes on Button Classes below to see a list of supported types

Note on Button Classes

Class
btn-default
btn-primary
btn-success
btn-info
btn-warning
btn-danger
btn-link

Note on Font Awesome

In opEvents-3.2.2 we are shipping the library 5.12.1

In opEvents-2.6.1 we are shipping the library 5.8.2