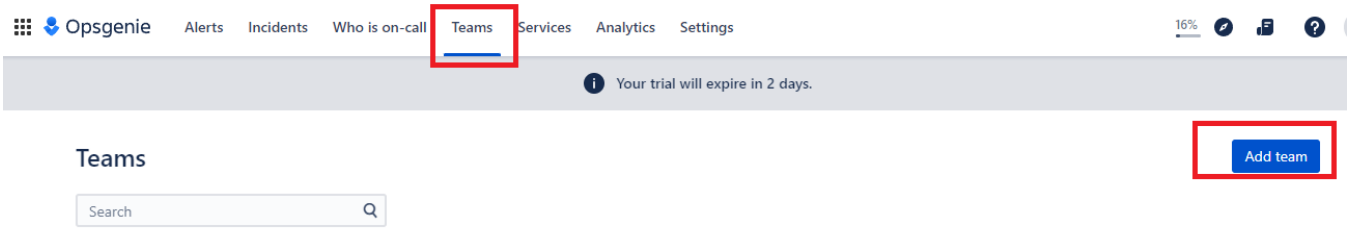
Opsgenie Integration

Introduction

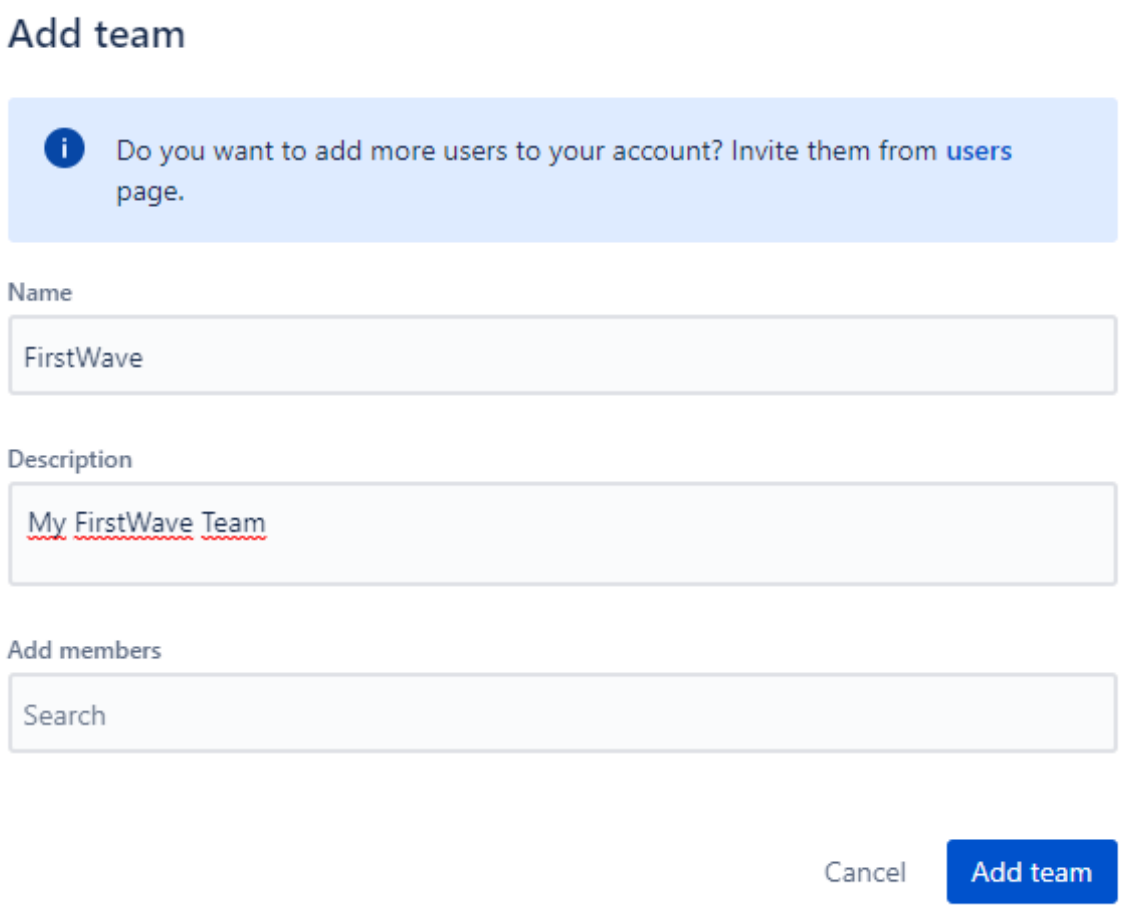
It is possible to push events generated via opEvents to external communication and collaboration tools, such as Opsgenie using a python script.

Opsgenie

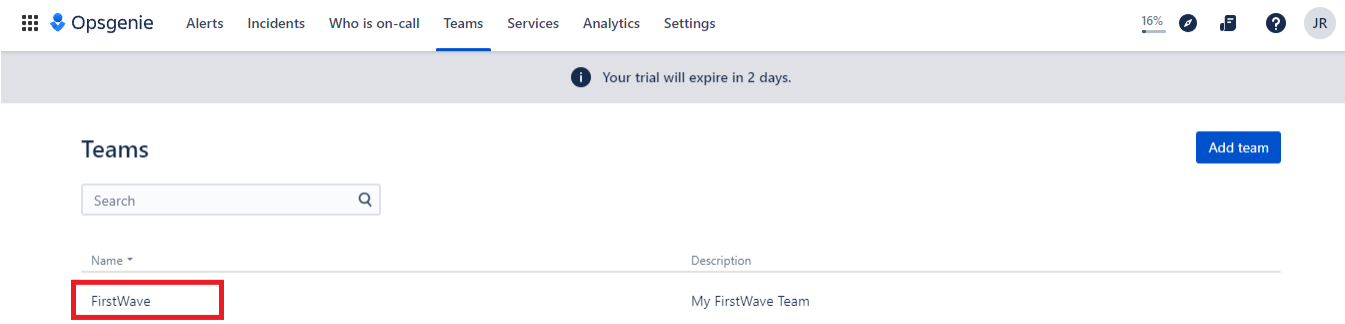
Before to start you must have a Opsgenie account, then you have to create a "Teams", navigate to Opsgenie portal > Teams > Add team

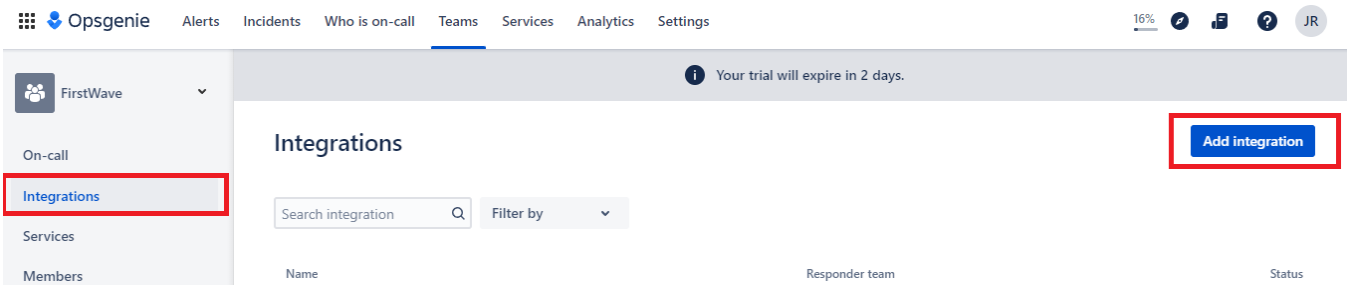


Please add "Name" and "Description" and click on "Add team".

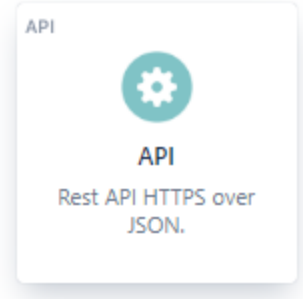


Click on the team that you created and navigate to integrations then click on "Add integration".

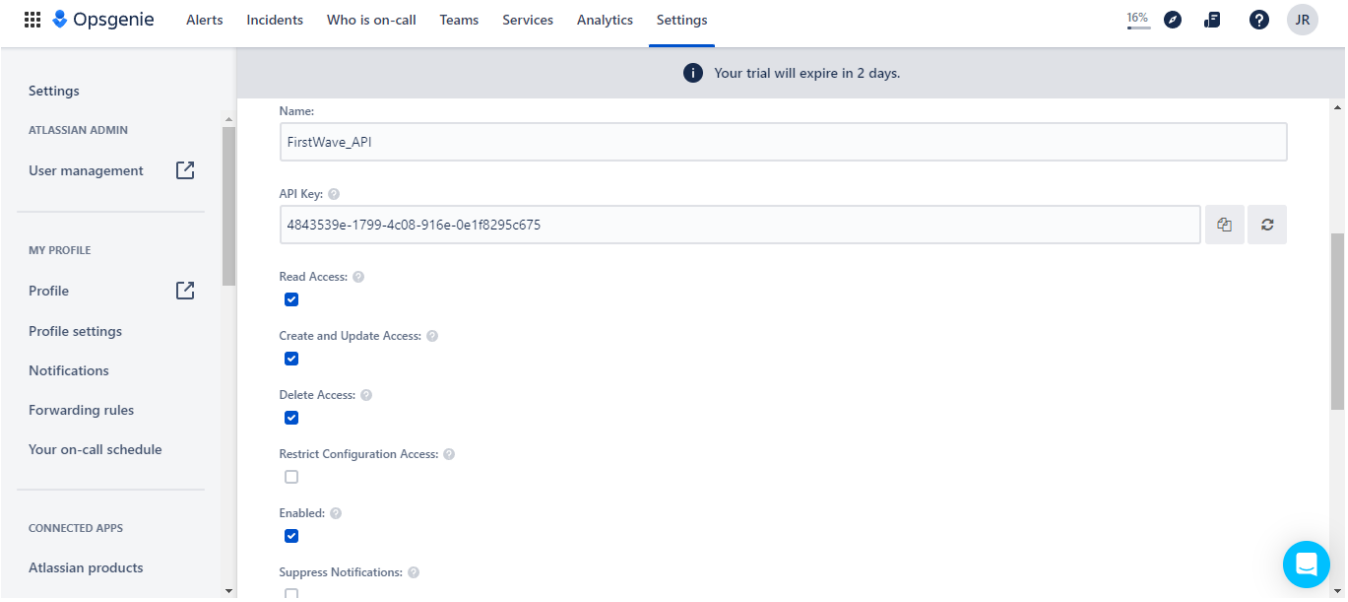




On integration select "API".



Please add "Name" and "Access". Then click on "Save Integration"



Python Scripts

In order to CREATE alerts on Opsgenie you have to copy the following script on OpEvents server.

```
# cd /usr/local/omk/script
# vi openOpsgenie.py
```

Copy the following script and replace "YOUR_API_KEY" with your actual Opsgenie API key.

```
#!/usr/bin/python3

import requests, json, sys


# Set your Opsgenie API key
API_KEY = "YOUR_API_KEY"

# Create a new alert in Opsgenie with the specified alias
def create_alert():
    url = "https://api.opsgenie.com/v2/alerts"
    headers = {"Content-Type": "application/json", "Authorization": "GenieKey " + API_KEY}
    data = {
        "alias": sys.argv[1],
        "message": sys.argv[2] +" : " + sys.argv[4],
        "priority": "P3",
        "tags": ["FirstWave", "Opmantek"],
        "description": sys.argv[2] + sys.argv[4] + ": This alert will write to the Opmantek event log.",
        "name": "FirstWave",
        "type": "team"
    }
    response = requests.post(url, headers=headers, data=json.dumps(data))
    if response.status_code == 202:
        print("Alert created successfully.")
    else:
        print("Failed to create alert.")

create_alert()
```

Change the owner and permissions.

```
# chown nmis:nmis openOpsgenie.py
# chmod 775 openOpsgenie.py
```

If you need to CLOSE and UPDATE the alerts you need to use the "stateful_eventids" opEvents key. This "stateful_eventids" is an array and the only way to get this value is with the opEvents API.

Copy the following script on OpEvents server. The "closeOpsgenie.py" script has the way to get the "stateful_eventids".

```
# cd /usr/local/omk/script
# vi closeOpsgenie.py
```

Copy the following script and replace:

- "YOUR_API_KEY" with your actual Opsgenie API key.
- "YOUR_IP_ADDRESS" with your opEvents IP Address.
- "YOUR_OPEVENTS_USER" with your opEvents User.
- "YOUR_OPEVENTS_PASSWORD" with your opEvents Password.

```
#!/usr/bin/python3

import sys, requests, json

API_KEY = "YOUR_API_KEY"


eventid = sys.argv[1]

host = 'http://YOUR_IP_ADDRESS/en/omk/opEvents'
loginurl = host + '/login'
eventurl = host + '/events/' + eventid + '.json'

s = requests.Session()

# Set proper headers
headers = {"Content-Type":"application/json","Accept":"application/json"}

user = 'YOUR_OPEVENTS_USER'
pwd = 'YOUR_OPEVENTS_PASSWORD'

auth = {'username': user, 'password': pwd}
loginresponse = s.post(loginurl, data = auth)
#print loginresponse
if loginresponse.status_code != 200:
    print('Login failed. Status:', loginresponse.status_code)
    sys.exit(1)

# Do the HTTP request
response = s.get(eventurl, headers=headers)

# Check for HTTP codes other than 200
if response.status_code != 200:
   print('Status:', response.status_code, 'Headers:', response.headers, 'Error Response:',response.json())
   exit(1)

# Decode the JSON response into a dictionary and use the data
result = response.json()
stateful_eventids = result.get('stateful_eventids', [])
statefulID = stateful_eventids[0]['$oid'] if stateful_eventids else None

print('Stateful ID:', statefulID)

# Close an existing alert in Opsgenie with the specified alias
def close_alert():
    url = 'https://api.opsgenie.com/v2/alerts/' + statefulID + '/close?identifierType=alias'
    headers = {"Content-Type": "application/json", "Authorization": "GenieKey " + API_KEY}
    data = {"user": "Opmantek@example.com", "note": sys.argv[2] + " : " + sys.argv[4] + ": Alert has been
resolved."}
    response = requests.post(url, headers=headers, data=json.dumps(data))
    if response.status_code == 202:
        print("Alert closed successfully.")
    else:
        print("Failed to close alert.")

# Example usage
close_alert()
```

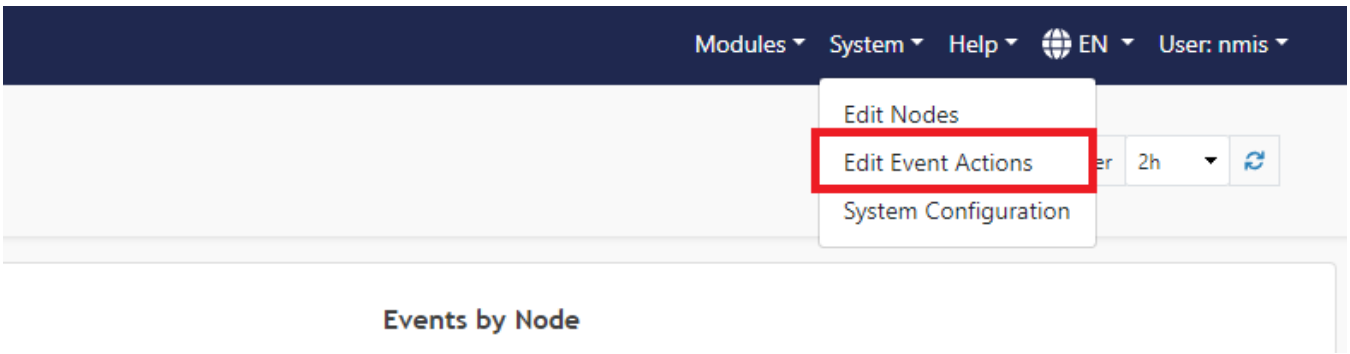
Note: In this case we are using the "alias" in order to CLOSE or UPDATE the alert on Opsgenie. Values are id, tiny and alias. Default value is id.

Change the owner and permissions.

```
# chown nmis:nmis openOpsgenie.py
# chmod 775 closeOpsgenie.py
```

OpEvents - EventActions

Log into your opEvents installation from your browser. Once authenticated, in the upper right, click System, then Edit Event Actions.



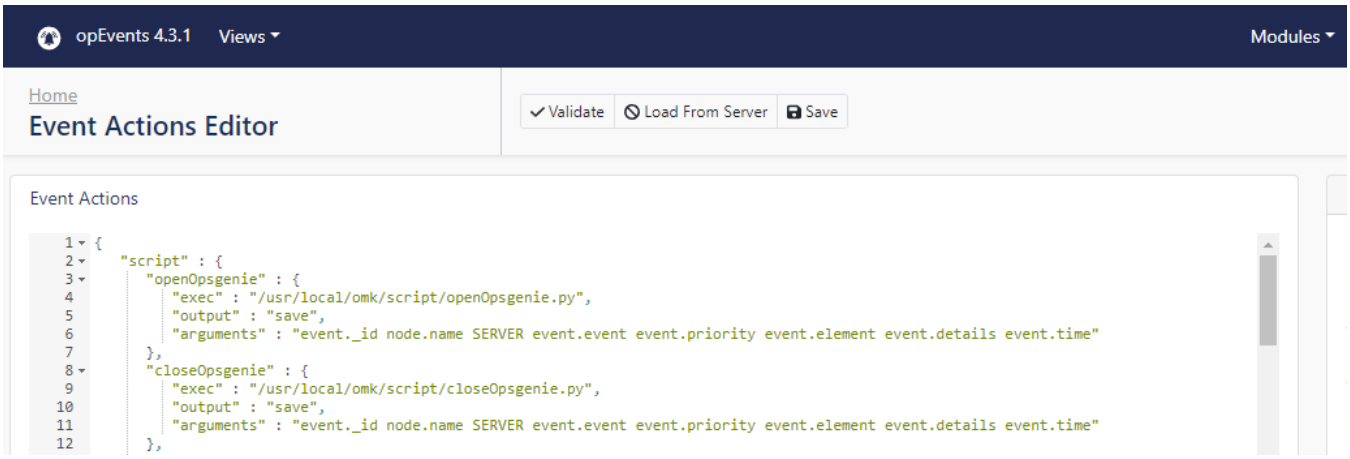
Immediately under the line that reads:

```
"script" : {
```

Add the text:

```
"openOpsgenie" : {
         "exec" : "/usr/local/omk/script/openOpsgenie.py",
         "output" : "save",
         "arguments" : "event._id node.name SERVER event.event event.priority event.element event.details event.
time"
      },
"closeOpsgenie" : {
         "exec" : "/usr/local/omk/script/closeOpsgenie.py",
         "output" : "save",
         "arguments" : "event._id node.name SERVER event.event event.priority event.element event.details event.
time"
      },
```

When complete this should look like the following:



Then click the Validate button. If you see a 'Syntax OK' prompt, proceed to click Save. You can also click Save and this will also check the syntax is OK.

You can now call the script through opEvents with the command:

script.openOpsgenie()

script.closeOpsgenie()

Add a call to the script when node goes down and when node goes up. This would look like the following:

```
        "38" : {
              "BREAK" : "false",
              "IF" : "event.event eq \"Node Up\"",
              "THEN" : [
                 "script.traceroute_node()",
                 "script.closeOpsgenie()"
              ]
           },
           "30" : {
              "BREAK" : "false",
              "IF" : "event.event eq \"Node Down\"",
              "THEN" : [
                 "script.traceroute_node()",
                 "script.openOpsgenie()",
                 "tag.isbroken(nodedown)",
                 "tag.verybad(42)"
              ]
           },
```

After modifying the Event Actions to suit your needs, you can then proceed to press the Save button, and verify the Syntax is OK. Once confirmed, you will then need to restart the opEvents daemon (opeventsd).

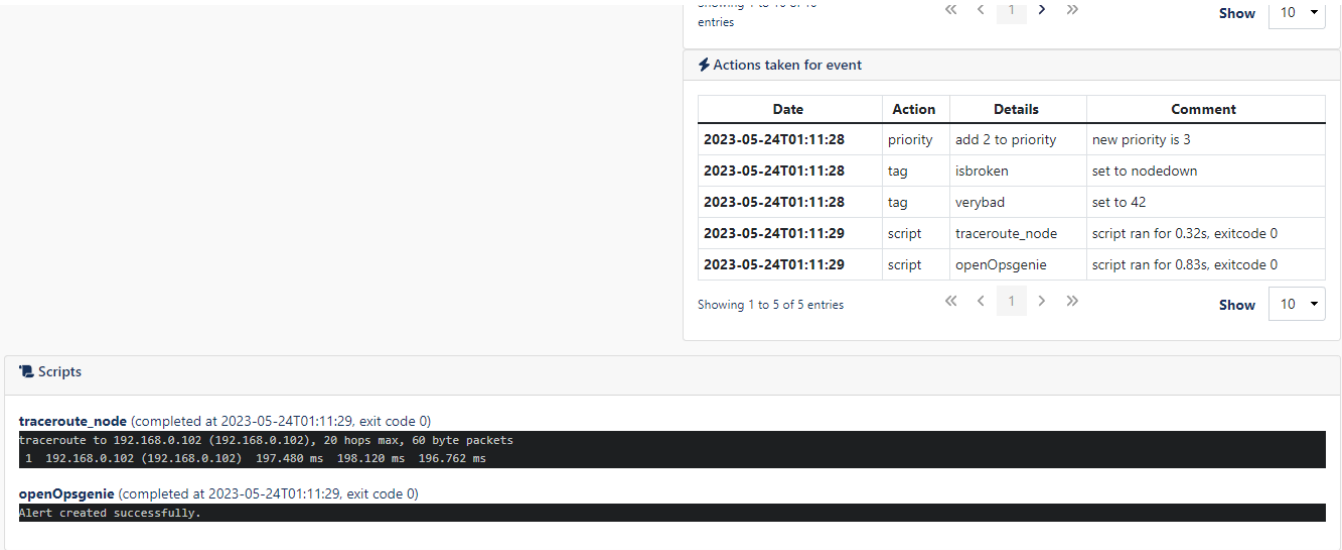
```
# systemctl restart opeventsd
```

Testing the integration

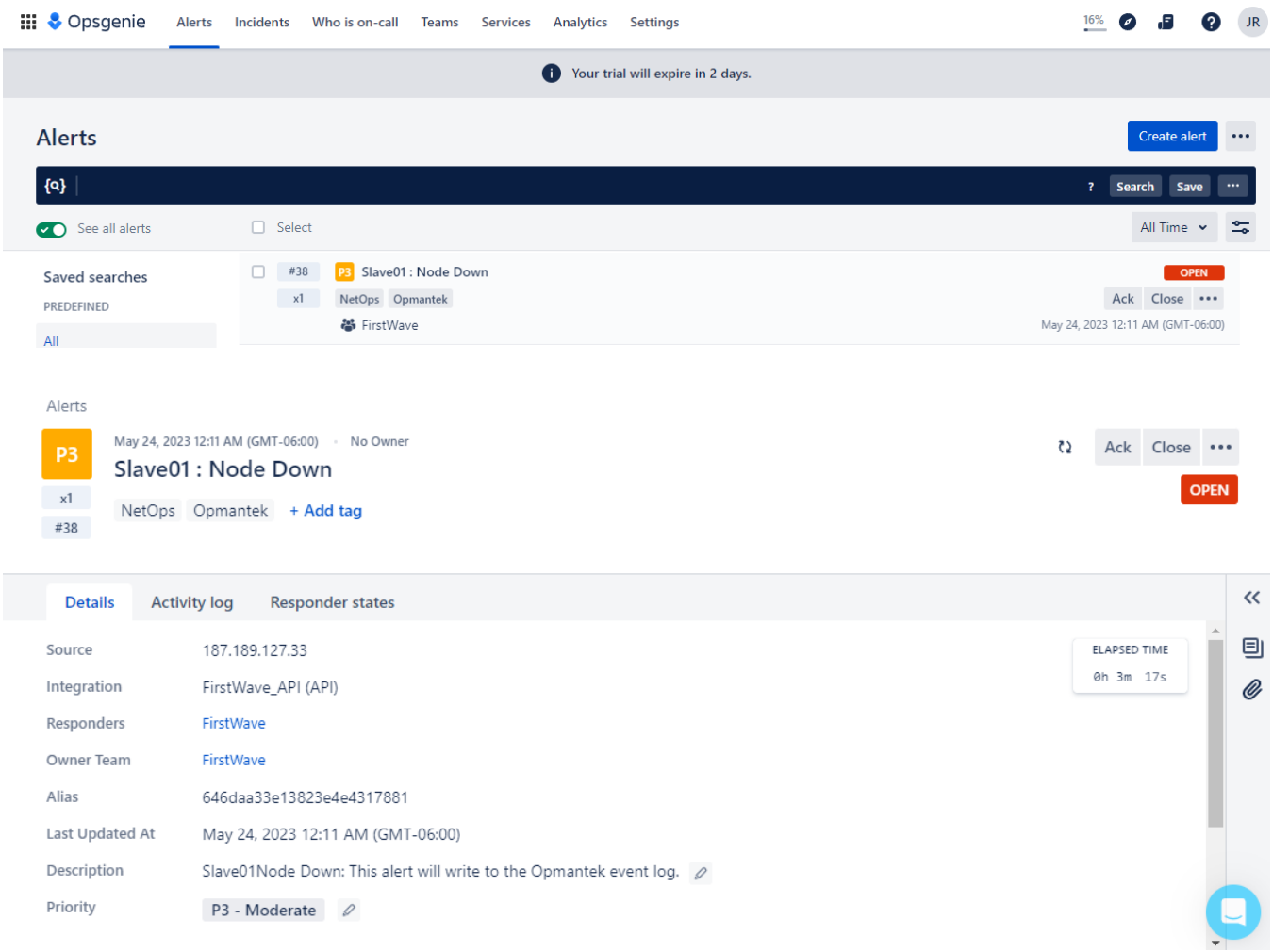
If you have applied the integration to an event that may not occur on a regular occurrence, you can run the following command to simulate the event and verify your integration works:

```
# /usr/local/omk/bin/opevents-cli.pl act=create-event node=Slave01 state=down event="Node Down" stateful="Node"
```

OpEvents Node Down



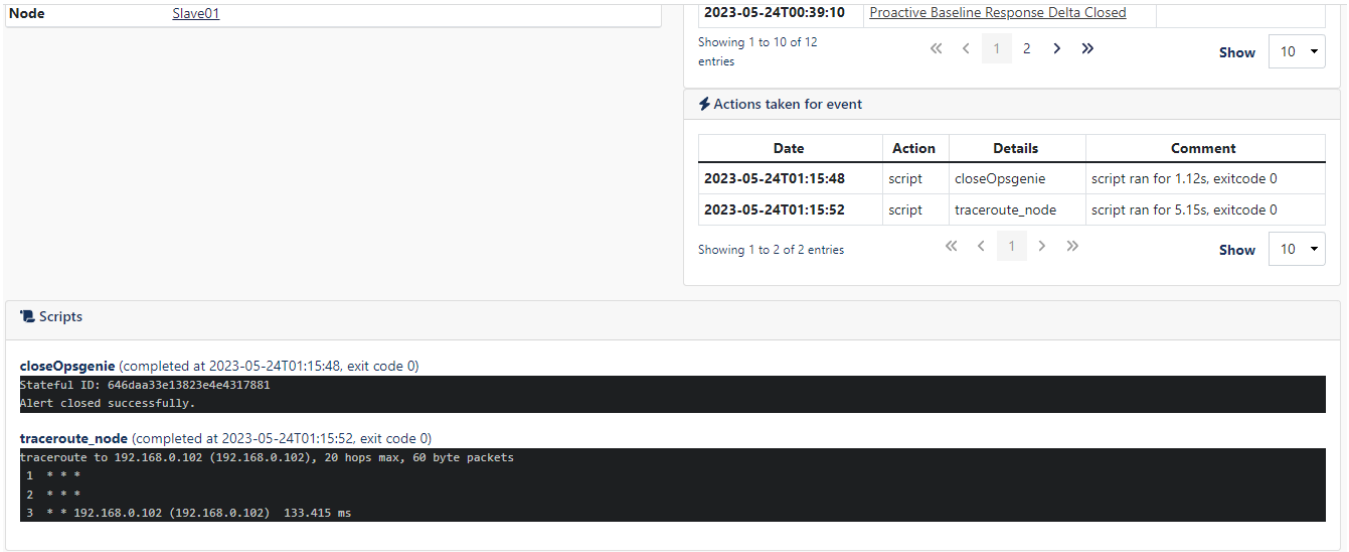
Open Opsgenie Alert



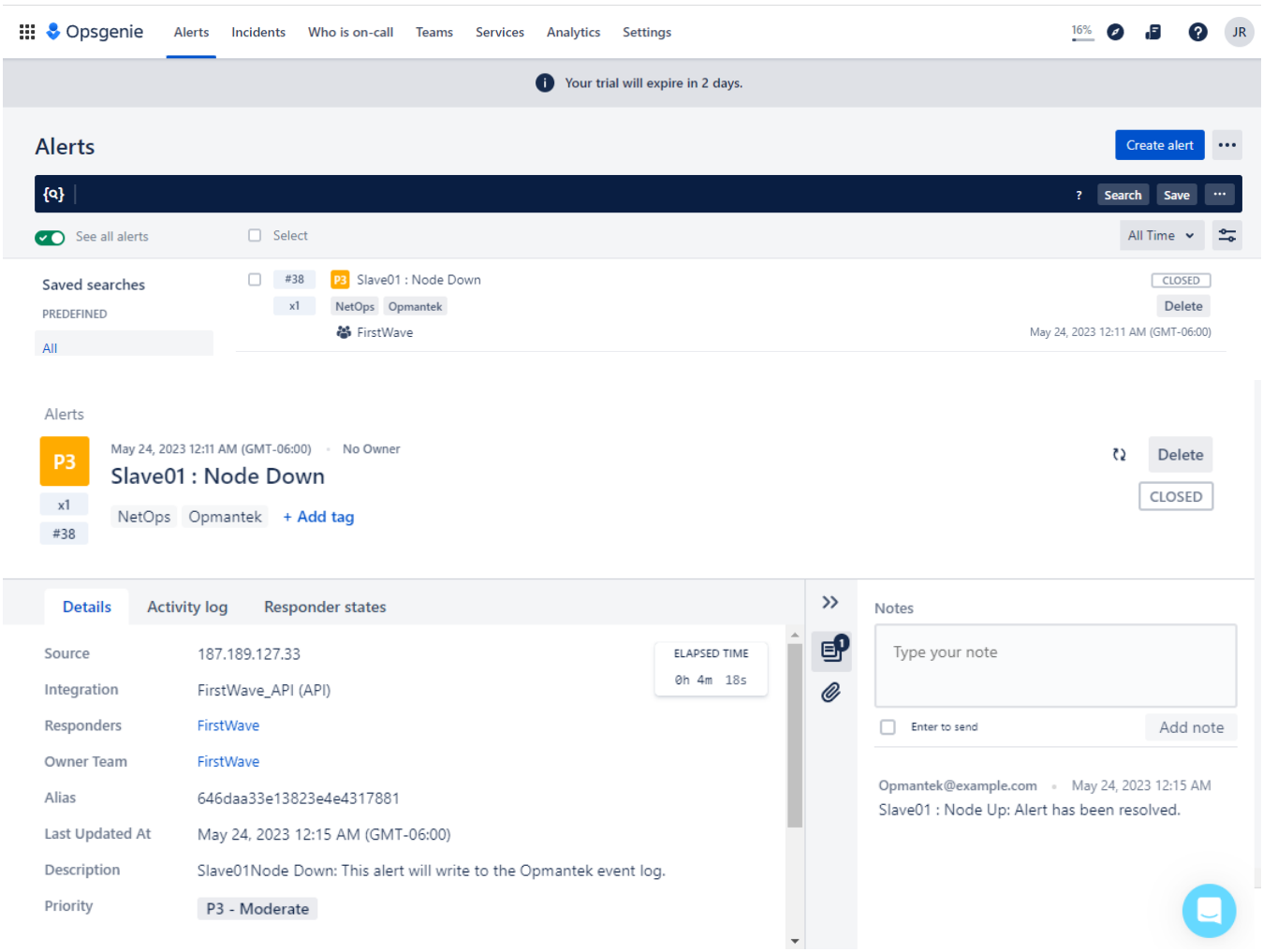
Close the Opsgenie Alarm

```
# /usr/local/omk/bin/opevents-cli.pl act=create-event node=Slave01 state=up event="Node Up" stateful="Node"
```

OpEvents Node Up



Close Opsgenie Alert



Alerts

P3

x1

#38

May 24, 2023 12:11 AM (GMT-06:00) • No Owner

Slave01 : Node Down

NetOps Opmantek + Add tag

Delete

CLOSED

Details | Activity log | Responder states

May 24 12:15 AM **comment** • Opmantek@example.com
Commented via API[FirstWave_API-Close Alert] with incomingDataId[cfb11e30-e089-467b-8b14-a8aafec1aaec]

May 24 12:15 AM **system** • Opmantek@example.com
Alert closed via API[FirstWave_API-Close Alert] with incomingDataId[cfb11e30-e089-467b-8b14-a8aafec1aaec]

May 24 12:11 AM **system** • System
No on-call user exists for Schedule [FirstWave_schedule].

May 24 12:11 AM **system** • System
Alert created via API[FirstWave_API-Create Alert] with incomingDataId[6e78eef1-7db1-4228-b55c-0336933d7404] with customSource[187.189.127.33] with tiny id [38] id [bfd6f333-f972-4ec1-9b7c-ba77e292a2a8-1684908689167]