+++ title = “Alerting HTTP API” description = “Grafana Alerts HTTP API” keywords = [“grafana”, “http”, “documentation”, “api”, “alerting”, “alerts”] aliases = [“/docs/grafana/latest/http_api/alerting/”] +++

Alerting API

> **Note:** This topic is relevant for the [legacy dashboard alerts]({{< relref “../alerting/old-alerting/_index.md” >}}) only.

You can find Grafana 8 alerts API specification details here. Also, refer to [Grafana 8 alerts documentation]({{< relref “../alerting/unified-alerting/_index.md” >}}) for details on how to create and manage new alerts.

You can use the Alerting API to get information about legacy dashboard alerts and their states but this API cannot be used to modify the alert. To create new alerts or modify them you need to update the dashboard JSON that contains the alerts.

Get alerts

`GET /api/alerts/`

Example Request:

```
GET /api/alerts HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

Querystring Parameters:

These parameters are used as querystring parameters. For example:

`/api/alerts?dashboardId=1`

- **dashboardId** – Limit response to alerts in specified dashboard(s). You can specify multiple dashboards, e.g. dashboardId=23&dashboardId=35.
- **panelId** – Limit response to alert for a specified panel on a dashboard.
- **query** - Limit response to alerts having a name like this value.
- **state** - Return alerts with one or more of the following alert states: `ALL`,`no_data`, `paused`, `alerting`, `ok`, `pending`. To specify multiple states use the following format: `?state=paused&state=alerting`
- **limit** - Limit response to X number of alerts.
- **folderId** – Limit response to alerts of dashboards in specified folder(s). You can specify multiple folders, e.g. folderId=23&folderId=35.
- **dashboardQuery** - Limit response to alerts having a dashboard name like this value.

- **dashboardTag** - Limit response to alerts of dashboards with specified tags. To do an "AND" filtering with multiple tags, specify the tags parameter multiple times e.g. dashboardTag=tag1&dashboardTag=tag2.

Example Response:

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "dashboardId": 1,
    "dashboardUId": "ABcdEFghij"
    "dashboardSlug": "sensors",
    "panelId": 1,
    "name": "fire place sensor",
    "state": "alerting",
    "newStateDate": "2018-05-14T05:55:20+02:00",
    "evalDate": "0001-01-01T00:00:00Z",
    "evalData": null,
    "executionError": "",
    "url": "http://grafana.com/dashboard/db/sensors"
  }
]
```

Get alert by id

```
GET /api/alerts/:id
```

Example Request:

```
GET /api/alerts/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

Example Response:

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "dashboardId": 1,
  "dashboardUId": "ABcdEFghij"
  "dashboardSlug": "sensors",
  "panelId": 1,
  "name": "fire place sensor",
```

```
  "state": "alerting",
  "message": "Someone is trying to break in through the fire place",
  "newStateDate": "2018-05-14T05:55:20+02:00",
  "evalDate": "0001-01-01T00:00:00Z",
  "evalData": "evalMatches": [
    {
      "metric": "movement",
      "tags": {
        "name": "fireplace_chimney"
      },
      "value": 98.765
    }
  ],
  "executionError": "",
  "url": "http://grafana.com/dashboard/db/sensors"
}
```

Important Note: "evalMatches" data is cached in the db when and only when the state of the alert changes (e.g. transitioning from "ok" to "alerting" state).

If data from one server triggers the alert first and, before that server is seen leaving alerting state, a second server also enters a state that would trigger the alert, the second server will not be visible in "evalMatches" data.

Pause alert by id

`POST /api/alerts/:id/pause`

Example Request:

```
POST /api/alerts/1/pause HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk

{
  "paused": true
}
```

The :id query parameter is the id of the alert to be paused or unpaused.

JSON Body Schema:

- **paused** – Can be `true` or `false`. True to pause an alert. False to unpause an alert.

Example Response:

```
HTTP/1.1 200
Content-Type: application/json
```

```
{
  "alertId": 1,
  "state":   "Paused",
  "message": "alert paused"
}
```

Pause all alerts

See [Admin API]({{< relref “admin.md#pause-all-alerts” >}}).