+++ title = "Authentication HTTP API " description = "Grafana Authentication HTTP API" keywords = ["grafana", "http", "documentation", "api", "authentication"] aliases = ["/docs/grafana/latest/http_api/authentication/"] +++

Authentication API

Tokens

Currently you can authenticate via an `API Token` or via a `Session cookie` (acquired using regular login or OAuth).

X-Grafana-Org-Id Header

X-Grafana-Org-Id is an optional property that specifies the organization to which the action is applied. If it is not set, the created key belongs to the current context org. Use this header in all requests except those regarding admin.

Example Request:

```
POST /api/auth/keys HTTP/1.1
Accept: application/json
Content-Type: application/json
X-Grafana-Org-Id: 2
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk

{
  "name": "mykey",
  "role": "Admin",
  "secondsToLive": 86400
}
```

Basic Auth

If basic auth is enabled (it is enabled by default), then you can authenticate your HTTP request via standard basic auth. Basic auth will also authenticate LDAP users.

curl example:

```
?curl http://admin:admin@localhost:3000/api/org
{"id":1,"name":"Main Org."}
```

Create API Token

Open the sidemenu and click the organization dropdown and select the `API Keys` option.

You use the token in all requests in the `Authorization` header, like this:

Example:

```
GET http://your.grafana.com/api/dashboards/db/mydash HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

The `Authorization` header value should be `Bearer <your api key>`.

The API Token can also be passed as a Basic authorization password with the special username `api_key`:

curl example:

```
?curl
http://api_key:eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk@localhost:3000/api/o

{"id":1,"name":"Main Org."}
```

Auth HTTP resources / actions

Api Keys

`GET /api/auth/keys`

Example Request:

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

Query Parameters:

- `includeExpired`: boolean. enable listing of expired keys. Optional.

Example Response:

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

[
  {
    "id": 3,
    "name": "API",
    "role": "Admin"
  },
  {
    "id": 1,
    "name": "TestAdmin",
    "role": "Admin",
    "expiration": "2019-06-26T10:52:03+03:00"
```

```
  }
]
```

Create API Key

`POST /api/auth/keys`

Example Request:

```
POST /api/auth/keys HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk

{
  "name": "mykey",
  "role": "Admin",
  "secondsToLive": 86400
}
```

JSON Body schema:

- **name** – The key name
- **role** – Sets the access level/Grafana Role for the key. Can be one of the following values: `Viewer`, `Editor` or `Admin`.
- **secondsToLive** – Sets the key expiration in seconds. It is optional. If it is a positive number an expiration date for the key is set. If it is null, zero or is omitted completely (unless `api_key_max_seconds_to_live` configuration option is set) the key will never expire.

Error statuses:

- **400** – `api_key_max_seconds_to_live` is set but no `secondsToLive` is specified or `secondsToLive` is greater than this value.
- **500** – The key was unable to be stored in the database.

Example Response:

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

{"name":"mykey","key":"eyJrIjoiWHZiSWd3NzdCYUZnNUtibE9obUpESmE3bzJYNDRIc0UiLCJuIjoibXl
```

Delete API Key

`DELETE /api/auth/keys/:id`

Example Request:

```
DELETE /api/auth/keys/3 HTTP/1.1
Accept: application/json
```

```
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

Example Response:

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

{"message":"API key deleted"}
```