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

Admin API

The Admin HTTP API does not currently work with an API Token. API Tokens are currently only linked to an organization and an organization role. They cannot be given the permission of server admin, only users can be given that permission. So in order to use these API calls you will have to use Basic Auth and the Grafana user must have the Grafana Admin permission. (The default admin user is called `admin` and has permission to use this API.)

> *If you are running Grafana Enterprise and have [Fine-grained access control]({{< relref "../enterprise/access-control/_index.md" >}}) enabled, for some endpoints you would need to have relevant permissions. Refer to specific resources to understand what permissions are required.*

Fetch settings

`GET /api/admin/settings`

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
settings:read	settings:* *settings:auth.saml:* settings:auth.saml:enabled (property level)

Example Request:

```
GET /api/admin/settings
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{
  "DEFAULT": {
    "app_mode":"production"
  },
  "analytics": {
    "google_analytics_ua_id":"",
    "reporting_enabled":"false"
  },
  "auth.anonymous":{
    "enabled":"true",
```

```
    "org_name":"Main Org.",
    "org_role":"Viewer"
  },
  "auth.basic":{
    "enabled":"false"
  },
  "auth.github":{
    "allow_sign_up":"false",
    "allowed_domains":"",
    "allowed_organizations":"",
    "api_url":"https://api.github.com/user",
    "auth_url":"https://github.com/login/oauth/authorize",
    "client_id":"some_id",
    "client_secret":"************",
    "enabled":"false",
    "scopes":"user:email,read:org",
    "team_ids":"",
    "token_url":"https://github.com/login/oauth/access_token"
  },
  "auth.google":{
    "allow_sign_up":"false","allowed_domains":"",
    "api_url":"https://www.googleapis.com/oauth2/v1/userinfo",
    "auth_url":"https://accounts.google.com/o/oauth2/auth",
    "client_id":"some_client_id",
    "client_secret":"************",
    "enabled":"false",
    "scopes":"https://www.googleapis.com/auth/userinfo.profile
https://www.googleapis.com/auth/userinfo.email",
    "token_url":"https://accounts.google.com/o/oauth2/token"
  },
  "auth.ldap":{
    "config_file":"/etc/grafana/ldap.toml",
    "enabled":"false"
  },
  "auth.proxy":{
    "auto_sign_up":"true",
    "enabled":"false",
    "header_name":"X-WEBAUTH-USER",
    "header_property":"username"
  },
  "dashboards.json":{
    "enabled":"false",
    "path":"/var/lib/grafana/dashboards"
  },
  "database":{
    "host":"127.0.0.1:0000",
    "name":"grafana",
    "password":"************",
    "path":"grafana.db",
    "ssl_mode":"disable",
    "type":"sqlite3",
    "user":"root"
```

```
},
"emails":{
  "templates_pattern":"emails/*.html, emails/*.txt",
  "welcome_email_on_sign_up":"false",
  "content_types":"text/html"
},
"log":{
  "buffer_len":"10000",
  "level":"Info",
  "mode":"file"
},
"log.console":{
  "level":""
},
"log.file":{
  "daily_rotate":"true",
  "file_name":"",
  "level":"",
  "log_rotate":"true",
  "max_days":"7",
  "max_lines":"1000000",
  "max_lines_shift":"28",
  "max_size_shift":""
},
"paths":{
  "data":"/tsdb/grafana",
  "logs":"/logs/apps/grafana"},
  "security":{
  "admin_password":"************",
  "admin_user":"admin",
  "cookie_remember_name":"grafana_remember",
  "cookie_username":"grafana_user",
  "disable_gravatar":"false",
  "login_remember_days":"7",
  "secret_key":"************"
},
"server":{
  "cert_file":"",
  "cert_key":"",
  "domain":"mygraf.com",
  "enable_gzip":"false",
  "enforce_domain":"false",
  "http_addr":"127.0.0.1",
  "http_port":"0000",
  "protocol":"http",
  "root_url":"%(protocol)s://%(domain)s:%(http_port)s/",
  "router_logging":"true",
  "data_proxy_logging":"true",
  "static_root_path":"public"
},
"session":{
  "cookie_name":"grafana_sess",
```

```
    "cookie_secure":"false",
    "gc_interval_time":"",
    "provider":"file",
    "provider_config":"sessions",
    "session_life_time":"86400"
  },
  "smtp":{
    "cert_file":"",
    "enabled":"false",
    "from_address":"admin@grafana.localhost",
    "from_name":"Grafana",
    "ehlo_identity":"dashboard.example.com",
    "host":"localhost:25",
    "key_file":"",
    "password":"************",
    "skip_verify":"false",
    "user":""
  },
  "users":{
    "allow_org_create":"true",
    "allow_sign_up":"false",
    "auto_assign_org":"true",
    "auto_assign_org_role":"Viewer"
  }
}
```


Update settings

`PUT /api/admin/settings`

> ***Note:*** *Available in Grafana Enterprise v8.0+.*

Updates / removes and reloads database settings. You must provide either `updates`, `removals` or both.

This endpoint only supports changes to `auth.saml` configuration.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
settings:write	settings:* *settings:auth.saml:* settings:auth.saml:enabled (property level)

Example request:

```
PUT /api/admin/settings
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlY2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

```
{
  "updates": {
    "auth.saml": {
      "enabled": "true"
    }
  },
  "removals": {
    "auth.saml": ["single_logout"]
  },
}
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 32

{
  "message":"Settings updated"
}
```

Status codes:

- **200** - OK
- **400** - Bad Request
- **401** - Unauthorized
- **403** - Forbidden
- **500** - Internal Server Error

Grafana Stats

`GET /api/admin/stats`

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
server.stats:read	n/a

Example Request:

```
GET /api/admin/stats
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{
  "users":2,
  "orgs":1,
  "dashboards":4,
  "snapshots":2,
  "tags":6,
  "datasources":1,
  "playlists":1,
  "stars":2,
  "alerts":2,
  "activeUsers":1
}
```

Grafana Usage Report preview

`GET /api/admin/usage-report-preview`

Preview usage report to be sent to vendor.

Only works with Basic Authentication (username and password). See introduction for an explanation.

Example Request:

```
GET /api/admin/usage-report-preview
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{
    "version": "8_4_0",
    "metrics": {
        "stats.active_admins.count": 1,
        "stats.active_editors.count": 1,
        "stats.active_sessions.count": 0,
        "stats.active_users.count": 2,
        "stats.active_viewers.count": 0,
        "stats.admins.count": 1,
        "stats.alert_rules.count": 0,
        "stats.alerting.ds.other.count": 0,
        "stats.alerts.count": 5,
        "stats.annotations.count": 6,
        "stats.api_keys.count": 1
```

```
    }
}
```

Global Users

`POST /api/admin/users`

Create new user. Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users:create	n/a

Example Request:

```
POST /api/admin/users HTTP/1.1
Accept: application/json
Content-Type: application/json

{
  "name":"User",
  "email":"user@graf.com",
  "login":"user",
  "password":"userpassword",
  "OrgId": 1
}
```

Note that `OrgId` is an optional parameter that can be used to assign a new user to a different organization when [auto_assign_org]({{< relref "../administration/configuration.md#auto-assign-org" >}}) is set to `true`.

Example Response:

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

{"id":5,"message":"User created"}
```

Password for User

`PUT /api/admin/users/:id/password`

Only works with Basic Authentication (username and password). See introduction for an explanation. Change password for a specific user.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users.password:update	global.users:*

Example Request:

```
PUT /api/admin/users/2/password HTTP/1.1
Accept: application/json
Content-Type: application/json

{"password":"userpassword"}
```

Example Response:

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

{"message": "User password updated"}
```

Permissions

`PUT /api/admin/users/:id/permissions`

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users.permissions:update	global.users:*

Example Request:

```
PUT /api/admin/users/2/permissions HTTP/1.1
Accept: application/json
Content-Type: application/json

{"isGrafanaAdmin": true}
```

Example Response:

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

{"message": "User permissions updated"}
```

Delete global User

`DELETE /api/admin/users/:id`

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users:delete	global.users:*

Example Request:

```
DELETE /api/admin/users/2 HTTP/1.1
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{"message": "User deleted"}
```

Pause all alerts

`POST /api/admin/pause-all-alerts`

Only works with Basic Authentication (username and password). See introduction for an explanation.

Example Request:

```
POST /api/admin/pause-all-alerts HTTP/1.1
Accept: application/json
Content-Type: application/json

{
  "paused": true
}
```

JSON Body schema:

- **paused** – If true then all alerts are to be paused, false unpauses all alerts.

Example Response:

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

{
  "state":   "Paused",
```

```
  "message": "alert paused",
  "alertsAffected": 1
}
```

Auth tokens for User

`GET /api/admin/users/:id/auth-tokens`

Return a list of all auth tokens (devices) that the user currently have logged in from.

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users.authtoken:list	global.users:*

Example Request:

```
GET /api/admin/users/1/auth-tokens HTTP/1.1
Accept: application/json
Content-Type: application/json
```

Example Response:

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

[
  {
    "id": 361,
    "isActive": false,
    "clientIp": "127.0.0.1",
    "browser": "Chrome",
    "browserVersion": "72.0",
    "os": "Linux",
    "osVersion": "",
    "device": "Other",
    "createdAt": "2019-03-05T21:22:54+01:00",
    "seenAt": "2019-03-06T19:41:06+01:00"
  },
  {
    "id": 364,
    "isActive": false,
    "clientIp": "127.0.0.1",
    "browser": "Mobile Safari",
    "browserVersion": "11.0",
    "os": "iOS",
    "osVersion": "11.0",
```

```
    "device": "iPhone",
    "createdAt": "2019-03-06T19:41:19+01:00",
    "seenAt": "2019-03-06T19:41:21+01:00"
  }
]
```

Revoke auth token for User

`POST /api/admin/users/:id/revoke-auth-token`

Revokes the given auth token (device) for the user. User of issued auth token (device) will no longer be logged in and will be required to authenticate again upon next activity.

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users.authtoken:update	global.users:*

Example Request:

```
POST /api/admin/users/1/revoke-auth-token HTTP/1.1
Accept: application/json
Content-Type: application/json

{
  "authTokenId": 364
}
```

Example Response:

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

{
  "message": "User auth token revoked"
}
```

Logout User

`POST /api/admin/users/:id/logout`

Logout user revokes all auth tokens (devices) for the user. User of issued auth tokens (devices) will no longer be logged in and will be required to authenticate again upon next activity.

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope
users.logout	global.users:*

Example Request:

```
POST /api/admin/users/1/logout HTTP/1.1
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{
  "message": "User auth token revoked"
}
```

Reload provisioning configurations

`POST /api/admin/provisioning/dashboards/reload`

`POST /api/admin/provisioning/datasources/reload`

`POST /api/admin/provisioning/plugins/reload`

`POST /api/admin/provisioning/notifications/reload`

`POST /api/admin/provisioning/access-control/reload`

Reloads the provisioning config files for specified type and provision entities again. It won't return until the new provisioned entities are already stored in the database. In case of dashboards, it will stop polling for changes in dashboard files and then restart it with new configurations after returning.

Only works with Basic Authentication (username and password). See introduction for an explanation.

Required permissions

See note in the [introduction]({{< ref "#admin-api" >}}) for an explanation.

Action	Scope	Provision entity
provisioning:reload	provisioners:accesscontrol	accesscontrol
provisioning:reload	provisioners:dashboards	dashboards
provisioning:reload	provisioners:datasources	datasources
provisioning:reload	provisioners:plugins	plugins
provisioning:reload	provisioners:notifications	notifications

Example Request:

```
POST /api/admin/provisioning/dashboards/reload HTTP/1.1
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{
  "message": "Dashboards config reloaded"
}
```

Reload LDAP configuration

`POST /api/admin/ldap/reload`

Reloads the LDAP configuration.

Only works with Basic Authentication (username and password). See introduction for an explanation.

Example Request:

```
POST /api/admin/ldap/reload HTTP/1.1
Accept: application/json
Content-Type: application/json
```

Example Response:

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

{
  "message": "LDAP config reloaded"
}
```