+++ title = "Auditing" description = "Auditing" keywords = ["grafana", "auditing", "audit", "logs"] weight = 1100 +++

Auditing

> ***Note:*** *Only available in Grafana Enterprise v7.3+.*

Auditing allows you to track important changes to your Grafana instance. By default, audit logs are logged to file but the auditing feature also supports sending logs directly to Loki.

Audit logs

Audit logs are JSON objects representing user actions like:

- Modifications to resources such as dashboards and data sources.
- A user failing to log in.

Format

Audit logs contain the following fields. The fields followed by ***** are always available, the others depend on the type of action logged.

Field name	Type	Description
`timestamp`*	string	The date and time the request was made, in coordinated universal time (UTC) using the RFC3339 format.
`user`*	object	Information about the user that made the request. Either one of the `UserID` or `ApiKeyID` fields will contain content if `isAnonymous=false`.
`user.userId`	number	ID of the Grafana user that made the request.
`user.orgId`*	number	Current organization of the user that made the request.
`user.orgRole`	string	Current role of the user that made the request.
`user.name`	string	Name of the Grafana user that made the request.
`user.tokenId`	number	ID of the user authentication token.
`user.apiKeyId`	number	ID of the Grafana API key used to make the request.
`user.isAnonymous`*	boolean	If an anonymous user made the request, `true`. Otherwise, `false`.
`action`*	string	The request action. For example, `create`, `update`, or `manage-permissions`.
`request`*	object	Information about the HTTP request.
`request.params`	object	Request's path parameters.
`request.query`	object	Request's query parameters.
`request.body`	string	Request's body.
`result`*	object	Information about the HTTP response.

`result.statusType`	string	If the request action was successful, `success`. Otherwise, `failure`.
`result.statusCode`	number	HTTP status of the request.
`result.failureMessage`	string	HTTP error message.
`result.body`	string	Response body.
`resources`	array	Information about the resources that the request action affected. This field can be null for non-resource actions such as `login` or `logout`.
`resources[x].id`*	number	ID of the resource.
`resources[x].type`*	string	The type of the resource that was logged: `alert`, `alert-notification`, `annotation`, `api-key`, `auth-token`, `dashboard`, `datasource`, `folder`, `org`, `panel`, `playlist`, `report`, `team`, `user`, or `version`.
`requestUri`*	string	Request URI.
`ipAddress`*	string	IP address that the request was made from.
`userAgent`*	string	Agent through which the request was made.
`grafanaVersion`*	string	Current version of Grafana when this log is created.
`additionalData`	object	Additional information that can be provided about the request.

The `additionalData` field can contain the following information:

Field name	Action	Description
`loginUsername`	`login`	Login used in the Grafana authentication form.
`extUserInfo`	`login`	User information provided by the external system that was used to log in.
`authTokenCount`	`login`	Number of active authentication tokens for the user that logged in.
`terminationReason`	`logout`	The reason why the user logged out, such as a manual logout or a token expiring.

Recorded actions

The audit logs include records about the following categories of actions. Each action is distinguished by the `action` and `resources[...].type` fields in the JSON record.

For example, creating an API key produces an audit log like this:

```
{
  "action": "create",
  "resources": [
    {
      "id": 1,
      "type": "api-key"
    }
  ],
  "timestamp": "2021-11-12T22:12:36.144795692Z",
```

```
  "user": {
    "userId": 1,
    "orgId": 1,
    "orgRole": "Admin",
    "username": "admin",
    "isAnonymous": false,
    "authTokenId": 1
  },
  "request": {
    "body": "{\"name\":\"example\",\"role\":\"Viewer\",\"secondsToLive\":null}"
  },
  "result": {
    "statusType": "success",
    "statusCode": 200,
    "responseBody": "{\"id\":1,\"name\":\"example\"}"
  },
  "resources": [
    {
      "id": 1,
      "type": "api-key"
    }
  ],
  "requestUri": "/api/auth/keys",
  "ipAddress": "127.0.0.1:54652",
  "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:94.0) Gecko/20100101
Firefox/94.0",
  "grafanaVersion": "8.3.0-pre"
}
```


Some actions can only be distinguished by their `requestUri` fields. For those actions, the relevant pattern of the `requestUri` field is given.

Sessions

Action	Distinguishing fields
Log in	`{"action": "login-AUTH-MODULE"}` *
Log out **	`{"action": "logout"}`
Force logout for user	`{"action": "logout-user"}`
Remove user authentication token	`{"action": "revoke-auth-token", "resources": [{"type": "auth-token"}, {"type": "user"}]}`
Create API key	`{"action": "create", "resources": [{"type": "api-key"}]}`
Delete API key	`{"action": "delete", "resources": [{"type": "api-key"}]}`

* Where `AUTH-MODULE` is the name of the authentication module: `grafana`, `saml`, `ldap`, etc.

** Includes manual log out, token expired/revoked, and [SAML Single Logout]({{< relref "saml.md#single-logout" >}}).

User management

Action	Distinguishing fields
Create user	`{"action": "create", "resources": [{"type": "user"}]}`
Update user	`{"action": "update", "resources": [{"type": "user"}]}`
Delete user	`{"action": "delete", "resources": [{"type": "user"}]}`
Disable user	`{"action": "disable", "resources": [{"type": "user"}]}`
Enable user	`{"action": "enable", "resources": [{"type": "user"}]}`
Update password	`{"action": "update-password", "resources": [{"type": "user"}]}`
Send password reset email	`{"action": "send-reset-email"}`
Reset password	`{"action": "reset-password"}`
Update permissions	`{"action": "update-permissions", "resources": [{"type": "user"}]}`
Send signup email	`{"action": "signup-email"}`
Click signup link	`{"action": "signup"}`
Reload LDAP configuration	`{"action": "ldap-reload"}`
Get user in LDAP	`{"action": "ldap-search"}`
Sync user with LDAP	`{"action": "ldap-sync", "resources": [{"type": "user"}]`

Team and organization management

Action	Distinguishing fields
Add team	`{"action": "create", "requestUri": "/api/teams"}`
Update team	`{"action": "update", "requestUri": "/api/teams/TEAM-ID"}`*
Delete team	`{"action": "delete", "requestUri": "/api/teams/TEAM-ID"}`*
Add external group for team	`{"action": "create", "requestUri": "/api/teams/TEAM-ID/groups"}`*
Remove external group for team	`{"action": "delete", "requestUri": "/api/teams/TEAM-ID/groups/GROUP-ID"}`*
Add user to team	`{"action": "create", "resources": [{"type": "user"}, {"type": "team"}]}`
Update team member permissions	`{"action": "update", "resources": [{"type": "user"}, {"type": "team"}]}`
Remove user from team	`{"action": "delete", "resources": [{"type": "user"}, {"type": "team"}]}`
Create organization	`{"action": "create", "resources": [{"type": "org"}]}`
Update organization	`{"action": "update", "resources": [{"type": "org"}]}`

Delete organization	`{"action": "delete", "resources": [{"type": "org"}]}`
Add user to organization	`{"action": "create", "resources": [{"type": "org"}, {"type": "user"}]}`
Change user role in organization	`{"action": "update", "resources": [{"type": "user"}, {"type": "org"}]}`
Remove user from organization	`{"action": "delete", "resources": [{"type": "user"}, {"type": "org"}]}`
Invite external user to organization	`{"action": "org-invite", "resources": [{"type": "org"}, {"type": "user"}]}`
Revoke invitation	`{"action": "revoke-org-invite", "resources": [{"type": "org"}]}`

* Where `TEAM-ID` is the ID of the affected team, and `GROUP-ID` (if present) is the ID of the external group.

Folder and dashboard management

Action	Distinguishing fields
Create folder	`{"action": "create", "resources": [{"type": "folder"}]}`
Update folder	`{"action": "update", "resources": [{"type": "folder"}]}`
Update folder permissions	`{"action": "manage-permissions", "resources": [{"type": "folder"}]}`
Delete folder	`{"action": "delete", "resources": [{"type": "folder"}]}`
Create/update dashboard	`{"action": "create-update", "resources": [{"type": "dashboard"}]}`
Import dashboard	`{"action": "create", "resources": [{"type": "dashboard"}]}`
Update dashboard permissions	`{"action": "manage-permissions", "resources": [{"type": "dashboard"}]}`
Restore old dashboard version	`{"action": "restore", "resources": [{"type": "dashboard"}]}`
Delete dashboard	`{"action": "delete", "resources": [{"type": "dashboard"}]}`

Library elements management

Action	Distinguishing fields
Create library element	`{"action": "create", "resources": [{"type": "library-element"}]}`
Update library element	`{"action": "update", "resources": [{"type": "library-element"}]}`
Delete library element	`{"action": "delete", "resources": [{"type": "library-element"}]}`

Data sources management

Action	Distinguishing fields
Create datasource	`{"action": "create", "resources": [{"type": "datasource"}]}`

Update datasource	`{"action": "update", "resources": [{"type": "datasource"}]}`
Delete datasource	`{"action": "delete", "resources": [{"type": "datasource"}]}`
Enable permissions for datasource	`{"action": "enable-permissions", "resources": [{"type": "datasource"}]}`
Disable permissions for datasource	`{"action": "disable-permissions", "resources": [{"type": "datasource"}]}`
Grant datasource permission to role, team, or user	`{"action": "create", "resources": [{"type": "datasource"}, {"type": "dspermission"}]}`*
Remove datasource permission	`{"action": "delete", "resources": [{"type": "datasource"}, {"type": "dspermission"}]}`
Enable caching for datasource	`{"action": "enable-cache", "resources": [{"type": "datasource"}]}`
Disable caching for datasource	`{"action": "disable-cache", "resources": [{"type": "datasource"}]}`
Update datasource caching configuration	`{"action": "update", "resources": [{"type": "datasource"}]}`

* `resources` may also contain a third item with `"type":` set to `"user"` or `"team"`.

Alerts and notification channels management

Action	Distinguishing fields
Save alert manager configuration	`{"action": "update", "requestUri": "/api/alertmanager/RECIPIENT/config/api/v1/alerts"}`
Reset alert manager configuration	`{"action": "delete", "requestUri": "/api/alertmanager/RECIPIENT/config/api/v1/alerts"}`
Create silence	`{"action": "create", "requestUri": "/api/alertmanager/RECIPIENT/api/v2/silences"}`
Delete silence	`{"action": "delete", "requestUri": "/api/alertmanager/RECIPIENT/api/v2/silences/SILENCE-ID"}`
Create alert	`{"action": "create", "requestUri": "/api/ruler/RECIPIENT/api/v2/alerts"}`
Create or update rule group	`{"action": "create-update", "requestUri": "/api/ruler/RECIPIENT/api/v1/rules/NAMESPACE"}`
Delete rule group	`{"action": "delete", "requestUri": "/api/ruler/RECIPIENT/api/v1/rules/NAMESPACE/GROUP-NAME"}`
Delete namespace	`{"action": "delete", "requestUri": "/api/ruler/RECIPIENT/api/v1/rules/NAMESPACE"}`
Test Grafana managed receivers	`{"action": "test", "requestUri":`

	`"/api/alertmanager/RECIPIENT/config/api/v1/receivers/test"}`
Create or update the NGalert configuration of the user's organization	`{"action": "create-update", "requestUri": "/api/v1/ngalert/admin_config"}`
Delete the NGalert configuration of the user's organization	`{"action": "delete", "requestUri": "/api/v1/ngalert/admin_config"}`

Where the following:

- `RECIPIENT` is `grafana` for requests handled by Grafana or the numeric data source ID for requests forwarded to a data source.
- `NAMESPACE` is the string identifier for the rules namespace.
- `GROUP-NAME` is the string identifier for the rules group.
- `SILENCE-ID` is the ID of the affected silence.

The following legacy alerting actions are still supported:

Action	Distinguishing fields
Test alert rule	`{"action": "test", "resources": [{"type": "panel"}]}`
Pause alert	`{"action": "pause", "resources": [{"type": "alert"}]}`
Pause all alerts	`{"action": "pause-all"}`
Test alert notification channel	`{"action": "test", "resources": [{"type": "alert-notification"}]}`
Create alert notification channel	`{"action": "create", "resources": [{"type": "alert-notification"}]}`
Update alert notification channel	`{"action": "update", "resources": [{"type": "alert-notification"}]}`
Delete alert notification channel	`{"action": "delete", "resources": [{"type": "alert-notification"}]}`

Reporting

Action	Distinguishing fields
Create report	`{"action": "create", "resources": [{"type": "report"}, {"type": "dashboard"}]}`
Update report	`{"action": "update", "resources": [{"type": "report"}, {"type": "dashboard"}]}`
Delete report	`{"action": "delete", "resources": [{"type": "report"}]}`
Send report by email	`{"action": "email", "resources": [{"type": "report"}]}`
Update reporting settings	`{"action": "change-settings"}`

Annotations, playlists and snapshots management

Action	Distinguishing fields
Create annotation	`{"action": "create", "resources": [{"type": "annotation"}]}`

Create Graphite annotation	`{"action": "create-graphite", "resources": [{"type": "annotation"}]}`
Update annotation	`{"action": "update", "resources": [{"type": "annotation"}]}`
Patch annotation	`{"action": "patch", "resources": [{"type": "annotation"}]}`
Delete annotation	`{"action": "delete", "resources": [{"type": "annotation"}]}`
Delete all annotations from panel	`{"action": "mass-delete", "resources": [{"type": "dashboard"}, {"type": "panel"}]}`
Create playlist	`{"action": "create", "resources": [{"type": "playlist"}]}`
Update playlist	`{"action": "update", "resources": [{"type": "playlist"}]}`
Delete playlist	`{"action": "delete", "resources": [{"type": "playlist"}]}`
Create a snapshot	`{"action": "create", "resources": [{"type": "dashboard"}, {"type": "snapshot"}]}`
Delete a snapshot	`{"action": "delete", "resources": [{"type": "snapshot"}]}`

Provisioning

Action	Distinguishing fields
Reload provisioned dashboards	`{"action": "provisioning-dashboards"}`
Reload provisioned datasources	`{"action": "provisioning-datasources"}`
Reload provisioned plugins	`{"action": "provisioning-plugins"}`
Reload provisioned notifications	`{"action": "provisioning-notifications"}`

Plugins management

Action	Distinguishing fields
Install plugin	`{"action": "install"}`
Uninstall plugin	`{"action": "uninstall"}`

Miscellaneous

Action	Distinguishing fields
Set licensing token	`{"action": "create", "requestUri": "/api/licensing/token"}`

Configuration

> ***Note:*** *The auditing feature is disabled by default.*

Audit logs can be saved into files, sent to a Loki instance or sent to the Grafana default logger. By default, only the file exporter is enabled. You can choose which exporter to use in the [configuration file]({{< relref "../administration/configuration.md" >}}).

Options are `file`, `loki`, and `logger`. Use spaces to separate multiple modes, such as `file loki`.

By default, when a user creates or updates a dashboard, its content will not appear in the logs as it can significantly increase the size of your logs. If this is important information for you and you can handle the amount of data generated, then you can enable this option in the configuration.

```
[auditing]
# Enable the auditing feature
enabled = false
# List of enabled loggers
loggers = file
# Keep dashboard content in the logs (request or response fields); this can
significantly increase the size of your logs.
log_dashboard_content = false
```

Each exporter has its own configuration fields.

File exporter

Audit logs are saved into files. You can configure the folder to use to save these files. Logs are rotated when the file size is exceeded and at the start of a new day.

```
[auditing.logs.file]
# Path to logs folder
path = data/log
# Maximum log files to keep
max_files = 5
# Max size in megabytes per log file
max_file_size_mb = 256
```

Loki exporter

Audit logs are sent to a Loki service, through HTTP or gRPC.

> *The HTTP option for the Loki exporter is only available in Grafana Enterprise v7.4+.*

```
[auditing.logs.loki]
# Set the communication protocol to use with Loki (can be grpc or http)
type = grpc
# Set the address for writing logs to Loki (format must be host:port)
url = localhost:9095
# Defaults to true. If true, it establishes a secure connection to Loki
tls = true
```

If you have multiple Grafana instances sending logs to the same Loki service or if you are using Loki for non-audit logs, audit logs come with additional labels to help identifying them:

- **host** - OS hostname on which the Grafana instance is running.
- **grafana_instance** - Application URL.
- **kind** - `auditing`

Console exporter

Audit logs are sent to the Grafana default logger. The audit logs use the `auditing.console` logger and are logged on `debug`-level, learn how to enable debug logging in the [log configuration]({{< relref "../administration/configuration.md#log" >}}) section of the documentation. Accessing the audit logs in this way is not recommended for production use.