Reviewing the audit log for your organization

The audit log allows organization admins to quickly review the actions performed by members of your organization. It includes details such as who performed the action, what the action was, and when it was performed.

In this article

Accessing the audit log

> **Note:** Webhooks might be a good alternative to the audit log or API polling for certain use cases. Webhooks are a way for GitHub to notify your server when specific events occur for a repository, organization, or enterprise. Compared to the API or searching the audit log, webhooks can be more efficient if you just want to learn and possibly log when certain events occur on your enterprise, organization, or repository. See "Webhooks documentation."

The audit log lists events triggered by activities that affect your organization within the last 180 days. Only owners can access an organization's audit log.

By default, only events from the past three months are displayed. To view older events, you must specify a date range with the `created` parameter. See "Understanding the search syntax."

1. In the upper-right corner of GitHub, select your profile photo, then click **Your organizations**.
2. Next to the organization, click **Settings**.
3. In the "Archive" section of the sidebar, click **Logs**, then click **Audit log**.

Searching the audit log

The name for each audit log entry is composed of a category of events, followed by an operation type. For example, the `repo.create` entry refers to the `create` operation on the `repo` category.

- The user affected by the action
- Which repository an action was performed in
- The action that was performed
- Which country the action took place in
- The date and time the action occurred

Note that you cannot search for entries using text. You can, however, construct search queries using a variety of filters. Many operators used when querying the log, such as `-`, `>`, or `<`, match the same format as searching across GitHub. For more information, see "About searching on GitHub."

Search based on operation

Use the `operation` qualifier to limit actions to specific types of operations. For example:

- `operation:access` finds all events where a resource was accessed.
- `operation:authentication` finds all events where an authentication event was performed.
- `operation:create` finds all events where a resource was created.

- `operation:modify` finds all events where an existing resource was modified.
- `operation:remove` finds all events where an existing resource was removed.
- `operation:restore` finds all events where an existing resource was restored.
- `operation:transfer` finds all events where an existing resource was transferred.

Search based on repository

Use the `repo` qualifier to limit actions to a specific repository. For example:

- `repo:my-org/our-repo` finds all events that occurred for the `our-repo` repository in the `my-org` organization.
- `repo:my-org/our-repo repo:my-org/another-repo` finds all events that occurred for both the `our-repo` and `another-repo` repositories in the `my-org` organization.
- `-repo:my-org/not-this-repo` excludes all events that occurred for the `not-this-repo` repository in the `my-org` organization.

Note that you must include the account name within the `repo` qualifier; searching for just `repo:our-repo` will not work.

Search based on the user

The `actor` qualifier can scope events based on who performed the action. For example:

- `actor:octocat` finds all events performed by `octocat`.
- `actor:octocat actor:hubot` finds all events performed by `octocat` or `hubot`.
- `-actor:hubot` excludes all events performed by `hubot`.

Note that you can only use a GitHub username, not an individual's real name.

Search based on the action performed

To search for specific events, use the `action` qualifier in your query. Actions listed in the audit log are grouped in different categories. For the full list of events in each category, see "Audit log events for your organization."

Category name	Description

`account`	Contains all activities related to your organization account.
`advisory_credit`	Contains all activities related to crediting a contributor for a security advisory in the GitHub Advisory Database. For more information, see "About repository security advisories."
`auto_approve_personal_access_token_requests`	Contains activities related to your organization's approval policy for fine-grained personal access tokens. For more information, see "Setting a personal access token policy for your organization."
`billing`	Contains all activities related to your organization's billing.
`business`	Contains activities related to business settings for an enterprise.
`codespaces`	Contains all activities related to your organization's codespaces.
`copilot`	Contains all activities related to your GitHub Copilot Business or GitHub Copilot Enterprise subscription.
`dependabot_alerts`	Contains organization-level configuration activities for Dependabot alerts in existing repositories. For more information, see "About Dependabot alerts."
`dependabot_alerts_new_repos`	Contains organization-level configuration activities for Dependabot alerts in new repositories created in the organization.
`dependabot_security_updates`	Contains organization-level configuration activities for Dependabot security updates in existing repositories. For more information, see "Configuring Dependabot security updates."
`dependabot_security_updates_new_repos`	Contains organization-level configuration activities for Dependabot security updates for new repositories created in the organization.

`dependency_graph`	Contains organization-level configuration activities for dependency graphs for repositories. For more information, see "About the dependency graph."
`dependency_graph_new_repos`	Contains organization-level configuration activities for new repositories created in the organization.
`discussion_post`	Contains all activities related to discussions posted to a team page.
`discussion_post_reply`	Contains all activities related to replies to discussions posted to a team page.
`enterprise`	Contains activities related to enterprise settings.
`hook`	Contains all activities related to webhooks.
`integration_installation`	Contains activities related to integrations installed in an account.
`integration_installation_request`	Contains all activities related to organization member requests for owners to approve integrations for use in the organization.
`issue`	Contains activities related to deleting an issue.
`marketplace_agreement_signature`	Contains all activities related to signing the GitHub Marketplace Developer Agreement.
`marketplace_listing`	Contains all activities related to listing apps in GitHub Marketplace.
`members_can_create_pages`	Contains all activities related to managing the publication of GitHub Pages sites for repositories in the organization. For more information, see "Managing the publication of GitHub Pages sites for your organization."
`org`	Contains activities related to organization membership.
`org_secret_scanning_automatic_validity_checks`	Contains organization-level activities related to enabling and disabling automatic validity checks for secret

	scanning. For more information, see "Managing security and analysis settings for your organization."
`organization_default_label`	Contains all activities related to default labels for repositories in your organization.
`oauth_application`	Contains all activities related to OAuth apps.
`packages`	Contains all activities related to GitHub Packages.
`payment_method`	Contains all activities related to how your organization pays for GitHub.
`personal_access_token`	Contains activities related to fine-grained personal access tokens in your organization. For more information, see "Managing your personal access tokens."
`profile_picture`	Contains all activities related to your organization's profile picture.
`project`	Contains all activities related to projects (classic).
`protected_branch`	Contains all activities related to protected branches.
`repo`	Contains activities related to the repositories owned by your organization.
`repository_advisory`	Contains repository-level activities related to security advisories in the GitHub Advisory Database. For more information, see "About repository security advisories."
`repository_content_analysis`	Contains all activities related to enabling or disabling data use for a private repository. For more information, see "Managing security and analysis settings for your repository."
`repository_dependency_graph`	Contains repository-level activities related to enabling or disabling the dependency graph for a private repository. For more information, see "About the dependency graph."

`repository_secret_scanning_automatic_validity_checks`	Contains repository-level activities related to enabling and disabling automatic validity checks for secret scanning. For more information, see "Enabling secret scanning for your repository."
`repository_vulnerability_alert`	Contains all activities related to Dependabot alerts.
`repository_vulnerability_alerts`	Contains repository-level configuration activities for Dependabot alerts.
`restore_member`	Triggered when an organization owner reinstates a member. For more information, see "Reinstating a former member of your organization."
`sponsors`	Contains all events related to sponsor buttons (see "Displaying a sponsor button in your repository")
`team`	Contains all activities related to teams in your organization.
`workflows`	Contains activities related to GitHub Actions workflows.

You can search for specific sets of actions using these terms. For example:

- `action:team` finds all events grouped within the team category.
- `-action:hook` excludes all events in the webhook category.

Each category has a set of associated actions that you can filter on. For example:

- `action:team.create` finds all events where a team was created.
- `-action:hook.events_changed` excludes all events where the events on a webhook have been altered.

Search based on time of action

Use the `created` qualifier to filter events in the audit log based on when they occurred. Date formatting must follow the ISO8601 standard, which is `YYYY-MM-DD` (year-month-day). You can also add optional time information `THH:MM:SS+00:00` after the date, to search by the hour, minute, and second. That's `T`, followed by `HH:MM:SS` (hour-minutes-seconds), and a UTC offset (`+00:00`).

When you search for a date, you can use greater than, less than, and range qualifiers to further filter results. For more information, see "Understanding the search syntax."

For example:

- `created:2014-07-08` finds all events that occurred on July 8th, 2014.
- `created:>=2014-07-08` finds all events that occurred on or after July 8th, 2014.
- `created:<=2014-07-08` finds all events that occurred on or before July 8th, 2014.
- `created:2014-07-01..2014-07-31` finds all events that occurred in the month of July 2014.

> **Note**: The audit log contains data for the last 180 days.

Search based on location

Using the qualifier `country`, you can filter events in the audit log based on the originating country. You can use a country's two-letter short code or its full name. Keep in mind that countries with spaces in their name will need to be wrapped in quotation marks. For example:

- `country:de` finds all events that occurred in Germany.
- `country:Mexico` finds all events that occurred in Mexico.
- `country:"United States"` all finds events that occurred in the United States.

Exporting the audit log

You can export the log as JSON data or a comma-separated value (CSV) file with the **Export** dropdown menu.

To filter the results in your export, search by one or more of these supported qualifiers before using the **Export** dropdown menu.

Qualifier	Example value
`action`	team.create
`actor`	octocat

`user`	codertocat
`org`	octo-org
`repo`	octo-org/documentation
`created`	2019-06-01

After you export the log, you'll see the following keys and values in the resulting file.

Key	Example value
`action`	team.create
`actor`	octocat
`user`	codertocat
`actor_location.country_code`	US
`org`	octo-org
`repo`	octo-org/documentation
`created_at`	1429548104000 (Timestamp shows the time since Epoch with milliseconds.)
`data.email`	octocat@nowhere.com
`data.hook_id`	245
`data.events`	["issues", "issue_comment", "pull_request", "pull_request_review_comment"]
`data.events_were`	["push", "pull_request", "issues"]
`data.target_login`	octocat

`data.old_user`	hubot
`data.team`	octo-org/engineering

Using the audit log API

Organizations that use GitHub Enterprise Cloud can interact with the audit log using the GraphQL API and REST API. For more information, see the GitHub Enterprise Cloud documentation.

Further reading

- "Keeping your organization secure"
- "Exporting member information for your organization"

Help and support

Contact support

Legal