Making authenticated API requests with a GitHub App in a GitHub Actions workflow

In this article

You can use an installation access token from a GitHub App to make authenticated API requests in a GitHub Actions workflow. You can also pass the token to a custom action to enable the action to make authenticated API requests.

About GitHub Actions authentication

If you need to make authenticated API requests in a GitHub Actions workflow or need to execute a custom action that requires a token, you should use the built-in `GITHUB_TOKEN` if possible. However, the `GITHUB_TOKEN` can only access resources within the workflow's repository. If you need to access additional resources, such as resources in an organization or in another repository, you can use a GitHub App. For more information about why you might use a GitHub App over a personal access token, see "About creating GitHub Apps."

Authenticating with a GitHub App

In order to use a GitHub App to make authenticated API requests, you must register a GitHub App, store your app's credentials, and install your app. Once this is done, you can use your app to create an installation access token, which can be used to make authenticated API requests in a GitHub Actions workflow. You can also pass the installation access token to a custom action that requires a token.

1. Register a GitHub App. Give your GitHub App registration the necessary permissions to access the desired resources. For more information, see "Registering a GitHub App" and "Choosing permissions for a GitHub App."
2. Store the app ID of your GitHub App as a GitHub Actions secret. You can find the app ID on the settings page for your app. The app ID is different from the client ID. For more information about navigating to the settings page for your GitHub App, see "Modifying a GitHub App registration." For more information about storing secrets, see "Using secrets in GitHub Actions."
3. Generate a private key for your app. Store the contents of the resulting file as a secret. (Store the entire contents of the file, including `-----BEGIN RSA PRIVATE KEY-----` and `-----END RSA PRIVATE KEY-----`.) For more information, see "Managing private keys for GitHub Apps."

4. Install the GitHub App on your user account or organization and grant it access to any repositories that you want your workflow to access. For more information, see "Installing your own GitHub App."

5. In your GitHub Actions workflow, create an installation access token, which you can use to make API requests.

   To do this, you can use a GitHub-owned action as demonstrated in the following example. If you prefer to not use this action, you can fork and modify the `actions/create-github-app-token` action, or you can write a script to make your workflow create an installation token manually. For more information, see "Authenticating as a GitHub App installation."

   The following example workflow uses the `actions/create-github-app-token` action to generate an installation access token. Then, the workflow uses the token to make an API request via the GitHub CLI.

   In the following workflow, replace `APP_ID` with the name of the secret where you stored your app ID. Replace `APP_PRIVATE_KEY` with the name of the secret where you stored your app private key.

YAML

```yaml
on:
  workflow_dispatch:
jobs:
  demo_app_authentication:
    runs-on: ubuntu-latest
    steps:
      - name: Generate a token
        id: generate_token
        uses: actions/create-github-app-token@v1
        with:
          app_id: ${{ secrets.APP_ID }}
          private_key: ${{ secrets.APP_PRIVATE_KEY }}

      - name: Use the token
        env:
          GITHUB_TOKEN: ${{ steps.generate_token.outputs.token }}
        run: |
          gh api octocat
```