Eclipse Foundation Coding Exercise

Summary

Name: 'secret_scanning' status checker

Description: Tool to check status of 'secret_scanning' security setting in repositories of a given GitHub organization

Repository: https://github.com/ideas-into-software/secret-scanning-status-checker

Requirements

- Command line tool
- Input
  - required
    - GitHub organisation name – name of GitHub organisation whose repositories should be checked
    - GitHub API token as environment variable
  - optional
    - name of GitHub API token environment variable – if not specified, defaults to ‘GITHUB_TOKEN’
    - name of JSON file to which results are output – if not specified, defaults to '[ORG_NAME].secret_scanning.json'
- Check status of `secret_scanning` security setting for each repository of a given GitHub organization
- If given GitHub organization exists and contains public repositories, collect results as array of objects with ‘full_name’ and ‘secret_scanning’ property for each
- Output results to file in JSON format

Non-functional requirements

`secret_scanning` is a property of `security_and_analysis` object returned by "**List organization repositories**" GitHub REST API endpoint, which has specific authorization requirements. More information:

- “**REST API endpoints for repositories: List organization repositories**”: https://docs.github.com/en/rest/repos/repos?apiVersion=2022-11-28#list-organization-repositories

  (...) *In order to see the `security_and_analysis` block for a repository you must have admin permissions for the repository or be an owner or security manager for the organization that owns the repository.* (…)

Technology stack

- Java 11
- picocli (command line library)
  - https://picocli.info/
  - https://github.com/remkop/picocli/tree/main
  - https://central.sonatype.com/artifact/info.picocli/picocli
- Gson (JSON library)
  - https://github.com/google/gson
  - https://central.sonatype.com/artifact/com.google.code.gson/gson
- SLF4J (logging library)
  - https://www.slf4j.org/api/org/slf4j/simple/SimpleLogger.html
- Maven (build and dependency management automation)

Possible extensions

The following elaborates on possible solutions to questions posed in original specification, i.e.

- Extract data for lots of organisations without hitting rate limit (“*What would you do when using the tool to extract data for lots of organisations without hitting the rate limit ?*”)
- Improve runtime performance when extracting data for lots of organisations (“How would you improve runtime performance when extracting data for lots of organisations ?”)
- Schedule such tool to monitor a set of organisations on a regular basis (“How would you schedule such a tool to monitor a set of organisations on regular basis ?”)

Following is proposed:

- Conditional requests
  - requires storing 'etag' response header for each organization
    - ➔ persistent key-value store / in-memory cache that can persist data on disk as this tool does not run continuously
  - if GitHub REST API conditional requests are used, rate limits do not apply
  - more information:
    - ➔ "**Best practices for using the REST API: Use conditional requests if appropriate**": https://docs.github.com/en/rest/using-the-rest-api/best-practices-for-using-the-rest-api?apiVersion=2022-11-28#use-conditional-requests-if-appropriate
- GraphQL API
  - For each repository in a given organization, only its 'full_name' and 'secret_scanning' property of 'security_and_analysis' object would be fetched
  - This would significantly minimize size of response payload, which needs to be transferred and parsed as JSON, then only that specific data extracted
  - more information:
    - ➔ "**Comparing GitHub's REST API and GraphQL API**": https://docs.github.com/en/rest/about-the-rest-api/comparing-githubs-rest-api-and-graphql-api?apiVersion=2022-11-28
      - (…) *The GraphQL API returns exactly the data that you request.* (…) *You can also accomplish the equivalent of multiple REST API request in a single GraphQL request. The ability to make fewer requests and fetch less data* (…)
- Scheduling
  - For 'secret_scanning' status checker tool as is (https://github.com/ideas-into-software/secret-scanning-status-checker) or similar, Quartz Job Scheduler (https://www.quartz-scheduler.org/) or simply cron could be used
  - For GitHub app with Webhooks version of https://github.com/ideas-into-software/secret-scanning-status-checker that would not be required – app would simply be run automatically in response to 'security_and_analysis' event subscribed to via webhook

Alternatively, instead of command line tool, using GitHub App with Webhooks

- Webhook subscribed to 'security_and_analysis' event

- run as serverless function
- If GitHub App is used, rate limits scale with organization size
- more information:
  - “**Deciding when to build a GitHub App**”: https://docs.github.com/en/apps/creating-github-apps/about-creating-github-apps/deciding-when-to-build-a-github-app

    ➔ (…) *GitHub Apps have scalable rate limits* (…) *GitHub Apps have built in webhooks* (…)
  - "**Building a GitHub App that responds to webhook events**": https://docs.github.com/en/apps/creating-github-apps/writing-code-for-a-github-app/building-a-github-app-that-responds-to-webhook-events

    ➔ (…) *Learn how to build a GitHub App that makes an API request in response to a webhook event.* (…)
  - “**Webhook events and payloads: security_and_analysis**”: https://docs.github.com/en/webhooks/webhook-events-and-payloads#security_and_analysis

    ➔ (…) *This event occurs when code security and analysis features are enabled or disabled for a repository* (…)
  - “**Types of webhooks: Organization webhooks**”: https://docs.github.com/en/webhooks/types-of-webhooks#organization-webhooks

    ➔ (…) *You can create webhooks in an organization to subscribe to events that occur in that organization. Organization webhooks can subscribe to events that happen in all repositories owned by the organization.* (…)
  - “**Rate limits for the REST API: Getting a higher rate limit**”: https://docs.github.com/en/rest/using-the-rest-api/rate-limits-for-the-rest-api?apiVersion=2022-11-28#getting-a-higher-rate-limit

    ➔ (…) *If you are using a personal access token for automation in your organization, consider whether a GitHub App will work instead. The rate limit for GitHub Apps using an installation access token scales with the number of repositories and number of organization users.* (…)
  - “**Managing your personal access tokens**”: https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/managing-your-personal-access-tokens

    ➔ (…) *Personal access tokens are intended to access GitHub resources on behalf of yourself. To access resources on behalf of an organization, or for long-lived integrations, you should use a GitHub App.* (…)