Repo Security

Frank Aragona Kathleen Conery

2023-04-09

🔓 Objectives

- Prevent sensitive information leaks to Github
- Set up guardrails, `.gitignore`, hooks
- Scrub private repos before they go public

If sensitive information is leaked and commited to the remote repo, then they will stay in the git history (and will require a lot of effort to remove them from the history). The following cannot be included in any repo **or any local commit!**:

Type	Examples
File Paths	• Network drives • Shared internal drives
Server Names	• ODBC Connections
Credentials	• SSH Keys • Tokens (REDCap, Azure, Github, etc) • Usernames • Passwords • Blob/bucket keys
Identifiable Information	• Addresses • Names • Any PHI

Prevent Credential Leaks with Env Variables

There are a number of ways to do this. We typically use a yaml file that can be filled out with personal credentials locally. The

file will not be committed to the remote repo

Create a private credentials file

The scripts use a `.yml` file that contains a list of API tokens, server names, and usernames/passwords specific to each individual user. There are two `.yml` files. One is a template (containing no actual passwords..) that exists in the repo and serves as a template so every individual user can keep up to date with new credential additions. The other is the individual `creds.yml` that is in the repo's `.gitignore`. This file will never exist in the repo and only exist locally (in the user's C drive).

`creds.yml` details

The `.yml` file can work with multiple programming languages including R and Python. They are read in the same way and can be easily adjusted when adding new passwords or using them as configuration files.

They look like this:

Listing 1 `local-credentials.yml`

```
# Default is needed to distinguish values.
# Leave a blank line (NO SPACES) as the last line in this file or things will break
# Quotes aren't necessary, but can be used.
default:
  conn_list_wdrs:
    Driver: "SQL Server Native Client 11.0"
    Server:
    Database:
    Trusted_connection:
    ApplicationIntent:

  fulgent:
    username: <USERNAME>
    password: <PASSWORD>
```

You can have different variables assigned to unique lists, which allows for easy configuration. For example, the list starting with `default` has variables `conn_list_wdrs` and `fulgent`. You can have a different list of variables within the same file like this:

Listing 2 `local-credentials.yml`

```
# Default is needed to distinguish values.
# Leave a blank line (NO SPACES) as the last line in this file or things will break
# Quotes aren't necessary, but can be used.
default:
  conn_list_wdrs:
    Driver: "SQL Server Native Client 11.0"
    Server:
    Database:
    Trusted_connection:
    ApplicationIntent:

  fulgent:
    username: <USERNAME>
    password: <PASSWORD>

test:
  conn_list_wdrs:
    Driver: "SQL Server Native Client 11.0"
    Server:
    Database:
    Trusted_connection:
    ApplicationIntent:
```

Now there is a `test` list with its own variables. This lets us switch a set of variables within our scripts. `default` applies to the main credentials where `test` can distinguish which variables should be test or dev scripts specific. Notice below that you can now call the credentials from a `.yml` file into an R or Python script and the actual credentials will never exist in the code pushed to the repo.