Personal Insights

Lars Hvam

March 25, 2022 | 1 minute read

On GitHub Personal Access Tokens(PATs)

When authenticating with GitHub a Personal Access Token(PAT) is typically used.

Note that a token with scope “repo” grants full access to all repositories, including private repositories, across all organizations your user can access(however, organizations with SMAL requires an additional step).

So, as GitHub recommends, it is important to

> *Treat your tokens like passwords and keep them secret*

abapGit approach

In abapGit, we have since ~2020 had the following approach

> *abapGit intentionally does not provide a way to store login data across sessions. This is because SAP systems are multi-user-systems and since abapGit is mostly used by developers on shared development systems who have extensive authorizations any approach to store passwords runs the risk of getting abused. On local single user systems you can easily implement the exit mentioned above to permanently store the login data in a RFC destination. Otherwise a password manager is the recommended approach to store login data.*

https://docs.abapgit.org/guide-authentication.html#security-considerations

By default, no tokens are stored in the system, they only reside in memory in the current session, when ZABAPGIT is exited, the memory is automatically cleared.

Secure Store

But how about the SAP Secure Store? Nope, cannot use it for abapGit,

> *Only SAP applications may use the secure storage*

The secure store is encrypted at rest(on the file system). And is secure(?) if ABAP code can write to the secure store and only kernel can read(ie. the secret is never in ABAP user memory) the information. In some setups, the ABAP code also reads from the store, and if this is present in a multi-user development system, there is a possibility that anyone can read the information in the secure store.

If the information in a development system secure store is also for development systems, its not a problem, as it can only impact non-critical systems. But GitHub PATs can potentially be used to access and modify proprietary or open source code.

Using the secure store to save PATs is ***not secure*** in a development system, given there is ABAP code to read from the store.

In case of leaks

If there is a possibility that a token has not been kept safe, consider it as leaked, and immediately revoke the token.

Plus inform all organizations where you have access, and perform a full audit of all commits potentially created using the token. Note that git is a multi user distributed versioning system, so without signed commits, its possible for everyone to author a commit with any email address plus rewrite any part of the git history.

Protecting users and developers

Enforcing a workflow requiring peer review, will make it impossible to do direct commits to the HEAD branch. This is not a direct solution, but limits auditing effort after a leaked token. So consider setting up,

- Protected branches
- Require pull requests
- Require peer review
- Require linear history

For more information see managing branch protection rules.

Stay safe 🤠

Alert Moderator

Assigned Tags

ABAP Development

git

github

Similar Blog Posts

How to bring your ABAP custom code to SAP BTP ABAP Environment

By Olga Dolinskaja Nov 11, 2019

API to Read and Write function module Test Data (SE37)

By Sandra Rossi Aug 12, 2021

Automatic checking of your ABAP code in Github/Gitlab with CI and abaplint

By Alexander Tsybulsky Dec 25, 2018

Related Questions

loop at screen

By Former Member Jul 15, 2008

Error when cloning GitHub repository - while setting up gCTS

By VIJAY RAO Sep 23, 2021

selecttion screen

By Former Member Nov 21, 2008

Join the Conversation

SAP TechEd

Tune in for tech talk. Stay for inspiration. Upskill your future.

SAP BTP Learning Group

SAP Business Technology Platform Learning Journeys.

Coffee Corner

Be the first to leave a comment

Add Comment

Find us on

Privacy	Terms of Use
Legal Disclosure	Copyright
Trademark	Cookie Preferences
Newsletter	Support