

ti&m

Commit Signing

Ondrej Halaska | DevOps Engineer

Zurich, 18th April 2024

ti&m

About Me



DevOps Engineer @ ti&m

- Technology enthusiast
- Automation, cloud computing, architecture, DevOps
- Responsible for maintenance and development of the internal infrastructure
- Designing solutions that optimize a variety of processes



Previous employments

- Deloitte, IBM, Diebold Nixdorf



Hobbies

- Hiking, traveling, running, CrossFit



Table of Contents

1. Hard Facts
2. What, Why, How?
3. How to verify commit?
4. Solution Architecture
5. Commit Signing Flow
6. Commit Verifying Flow
7. Demo
8. Conclusion
9. Q&A (15 minutes)



Source: https://www.freepik.com/free-photo/html-css-collage-concept-with-hacker_36295469.htm#query=hacker&position=1&from_view=keyword&track=sph

> 80%

of organizations have had
more than one breach

2.3%

increase from the year 2022

277 days

to identify and contain a data
breach

USD 4.45 million

average cost of a data breach

**healthcare, finance,
pharmaceutical, energy
and technology industries**

Commit Signing



WHAT?

- OpenPGP standard that can be used for commit signing (private & public key).
- Digitally signed commits.



WHY?

- To *enhance security posture within SDLC* by ensuring of:
 - Data integrity.
 - Non-repudiation.
 - Authenticity.



HOW?

- `gnupg` utility for a GPG key pair generation.
- Set the GPG key pair within *less than 1 minute*:
 - Generate the GPG key pair and configure `.gitconfig` file.
 - Upload the GPG public key.
 - Download other developers' GPG public keys and verify their commit signature.



JavaScript



Commit Verification (git CLI: “*git log --show-signature*”)



Public key is
not imported

```
commit 76c8a7fd53da7745c744591ddbc2d80015bd5364
gpg: Signature made Di 27 Sep 18:18:08 2022 CEST
gpg:                using EDDSA key 73FD74409D97181BE5AC57068B164332308BF5E4
gpg: Can't check signature: No public key
Author: Yves Schumann <yves.schumann@ti8m.ch>
Date:   Tue Sep 27 18:18:07 2022 +0200

[ys] Improved help content
```



Public key is *imported*
but *not trusted*

```
commit a09fe220d0aafafbdee5ed538456e6fc1ee328e3 (HEAD -> feature/gpg-statistics)
gpg: Signature made Mi  1 Feb 13:38:51 2023 CET
gpg:                using EDDSA key 12BDEC94C3134A31F594F0852257F2A674DF1489
gpg: Good signature from "Johh Wick (GPG-test-key) <john.wick@gmail.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:                There is no indication that the signature belongs to the owner.
Primary key fingerprint: 12BD EC94 C313 4A31 F594  F085 2257 F2A6 74DF 1489
Author: John Wick <john.wick@gmail.com>
Date:   Wed Feb 1 13:38:51 2023 +0100

Add test file
```



Public key is *imported*
and *trusted*

```
commit dae778cafb614adab52abd0b6011648cbe48937d (HEAD -> feature/gpg-statistics)
gpg: Signature made Fr 27 Jan 11:07:10 2023 CET
gpg:                using EDDSA key BF285A3431EA3FCF39F12E09CE721E540DA2A5F4
gpg: Good signature from "GitLab ti&m (GPG key: GitLab ti8m) <ondrej.halaska@ti8m.ch>" [ultimate]
Author: Ondrej Halaska <ondrej.halaska@ti8m.ch>
Date:   Fri Jan 27 11:07:10 2023 +0100
```

Replace ampersand for another character and fix indentation

Commit Verification (git CLI: "*git verify-commit <commit-hash>*")



Public key is
not imported

```
→ gpg-key git:(feature/gpg-statistics) git verify-commit 76c8a7fd53da7745c744591ddbc2d80015bd5364
gpg: Signature made Di 27 Sep 18:18:08 2022 CEST
gpg:                using EDDSA key 73FD74409D97181BE5AC57068B164332308BF5E4
gpg: Can't check signature: No public key
```



Public key is *imported*
but *not trusted*

```
→ gpg-key git:(feature/gpg-statistics) git verify-commit 3c9ecc76f5a1f619e686b37a6575be3141ef6daf
gpg: Signature made Di  7 Feb 16:46:21 2023 CET
gpg:                using EDDSA key 12BDEC94C3134A31F594F0852257F2A674DF1489
gpg: Good signature from "Johh Wick (GPG-test-key) <john.wick@gmail.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:                There is no indication that the signature belongs to the owner.
Primary key fingerprint: 12BD EC94 C313 4A31 F594  F085 2257 F2A6 74DF 1489
```



Public key is *imported*
and *trusted*

```
→ gpg-key git:(feature/gpg-statistics) git verify-commit dae778cafb614adab52abd0b6011648cbe48937d
gpg: Signature made Fr 27 Jan 11:07:10 2023 CET
gpg:                using EDDSA key BF285A3431EA3FCF39F12E09CE721E540DA2A5F4
gpg: Good signature from "GitLab ti&m (GPG key: GitLab ti8m) <ondrej.halaska@ti8m.ch>" [ultimate]
```

Commit Verification (GitLab UI)

Feature/gpg statistics

 Open Halaska Ondrej requested to merge `feature/gpg-statistics` into `dev` 1 month ago


Overview 11 **Commits 100** Pipelines 0 Changes 19

10 Jan, 2023 1 commit




Update default account name

Halaska Ondrej authored 6 days ago




This commit was signed with a **verified** signature and the committer email is verified to belong to the same user.

 **Halaska Ondrej**
@oha

GPG Key ID: CE721E540DA2A5F4
[Learn more about signing commits](#)

Verified

fc340bc9



Feature/gpg statistics

 Open Halaska Ondrej requested to merge `feature/gpg-statistics` into `dev` 2 months

Overview 29 **Commits 102** Pipelines 0 Changes 20

01 Feb, 2023 1 commit



Add test file

John Wick authored 4 minutes ago

Unverified signature


This commit was signed with an unverified signature.

GPG Key ID:
12BDEC94C3134A31F594F0852257F2A67DF1489

[Learn about signing commits](#)

Unverified

fa59a0af



Protected Branch (GitLab)

Branch defaults

[Expand](#)

Select the default branch for this project, and configure the template for branch names.

Push rules

[Collapse](#)

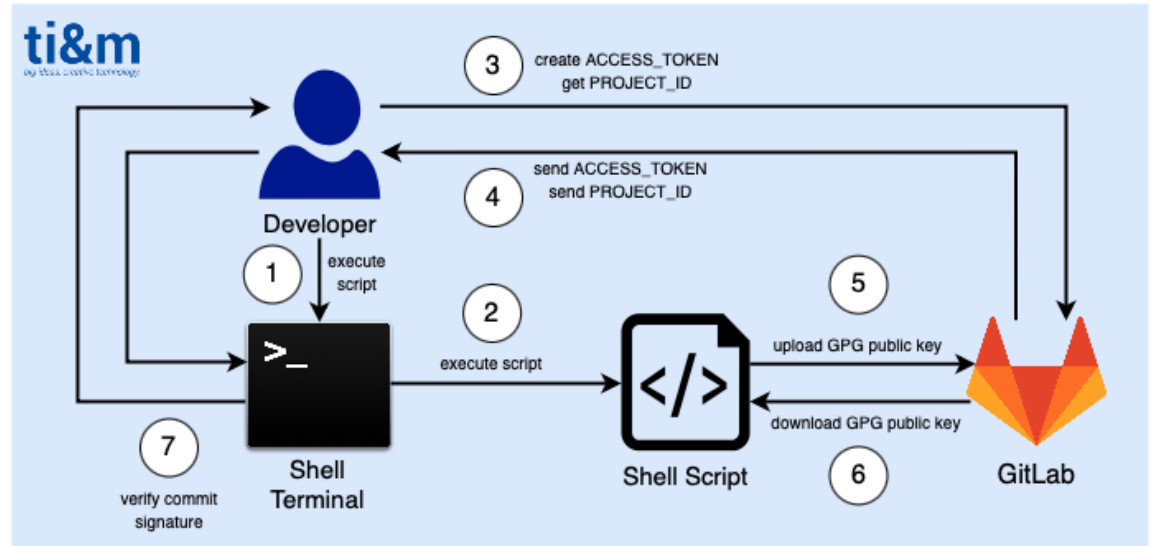
Restrict push operations for this project. [Learn more.](#)

Select push rules

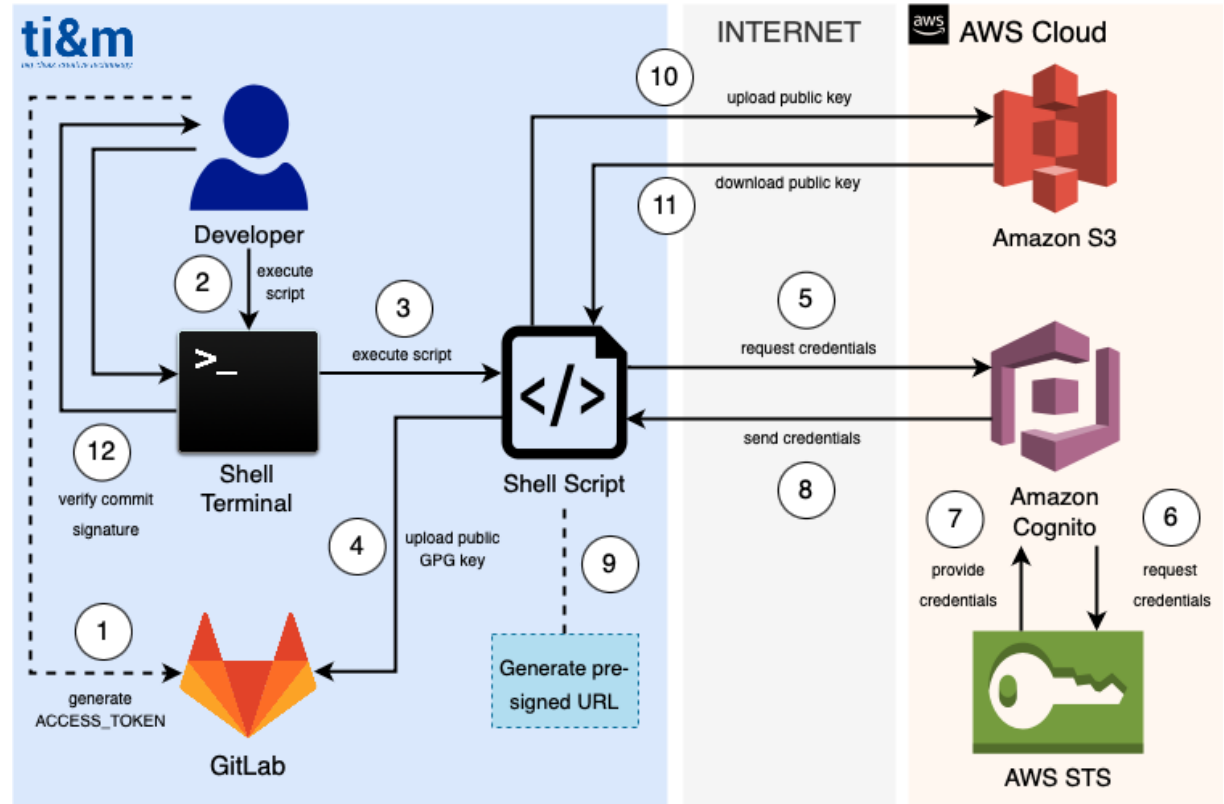
- ☐ Reject unverified users
Users can only push commits to this repository if the committer email is one of their own verified emails.
- ☒ Reject unsigned commits
Only signed commits can be pushed to this repository.
- ☐ Reject commits that aren't DCO certified
Only commits that include a `Signed-off-by:` element can be pushed to this repository.
- ☐ Do not allow users to remove Git tags with `git push`
Users can still delete tags through the GitLab UI.
- ☐ Check whether the commit author is a GitLab user
Restrict commits to existing GitLab users.
- ☐ Prevent pushing secret files
Reject any files likely to contain secrets. [What secret files are rejected?](#)

```
+ gpg-key git:(feature/gpg-statistics) git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 10 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 276 bytes | 276.00 KiB/s, done.
Total 3 (delta 1), reused 1 (delta 0), pack-reused 0
remote: GitLab: Commit must be signed with a GPG key
To gitlab.ti8m.ch:ti8m-forge-examples/gpg-key.git
! [remote rejected] feature/gpg-statistics -> feature/gpg-statistics (pre-receive hook declined)
error: failed to push some refs to 'gitlab.ti8m.ch:ti8m-forge-examples/gpg-key.git'
```

GPG Key Pair Generation & Commit Signing: Solution Architecture (on-prem)

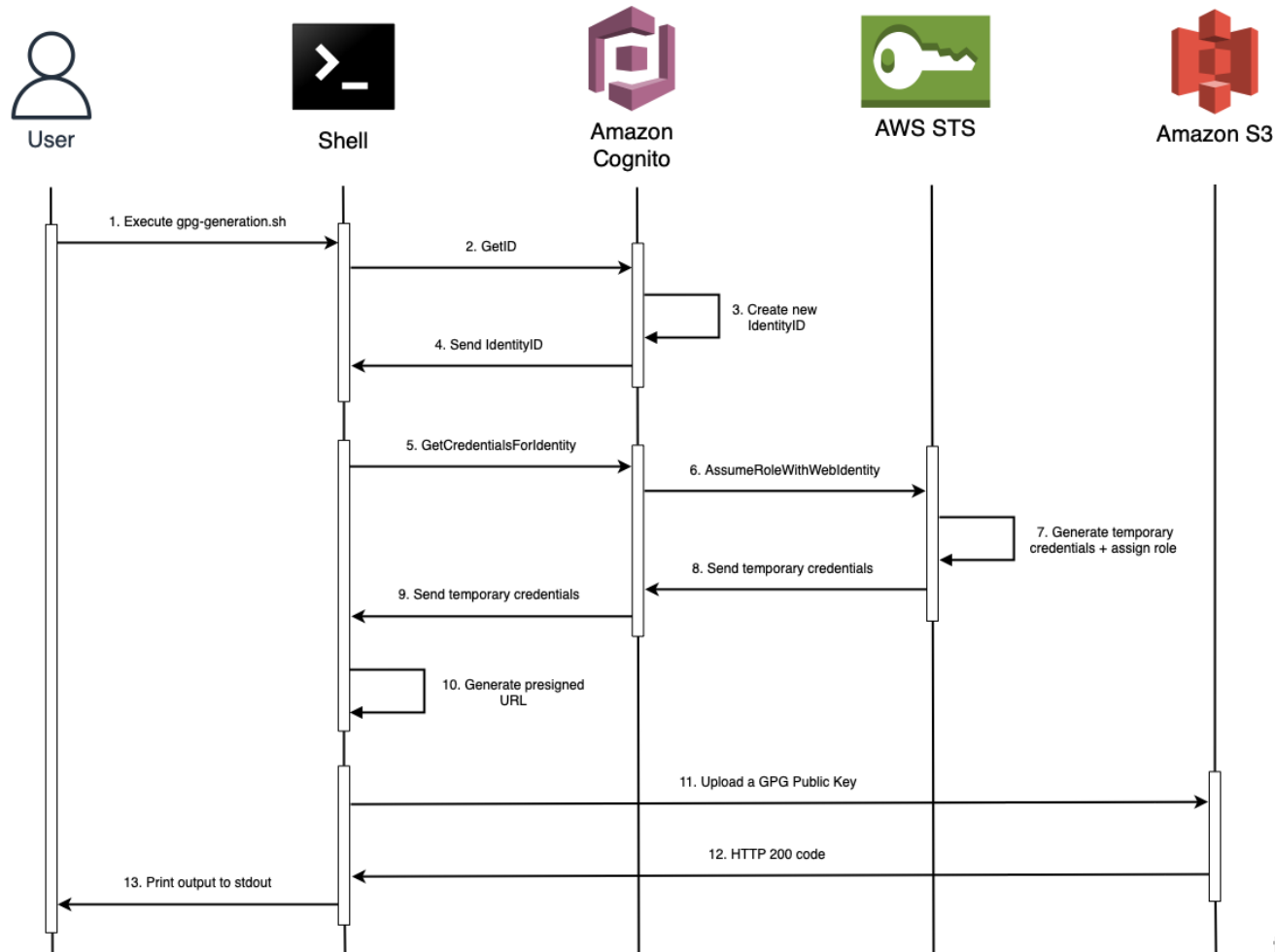


GPG Key Pair Generation & Commit Signing: Solution Architecture (cloud)

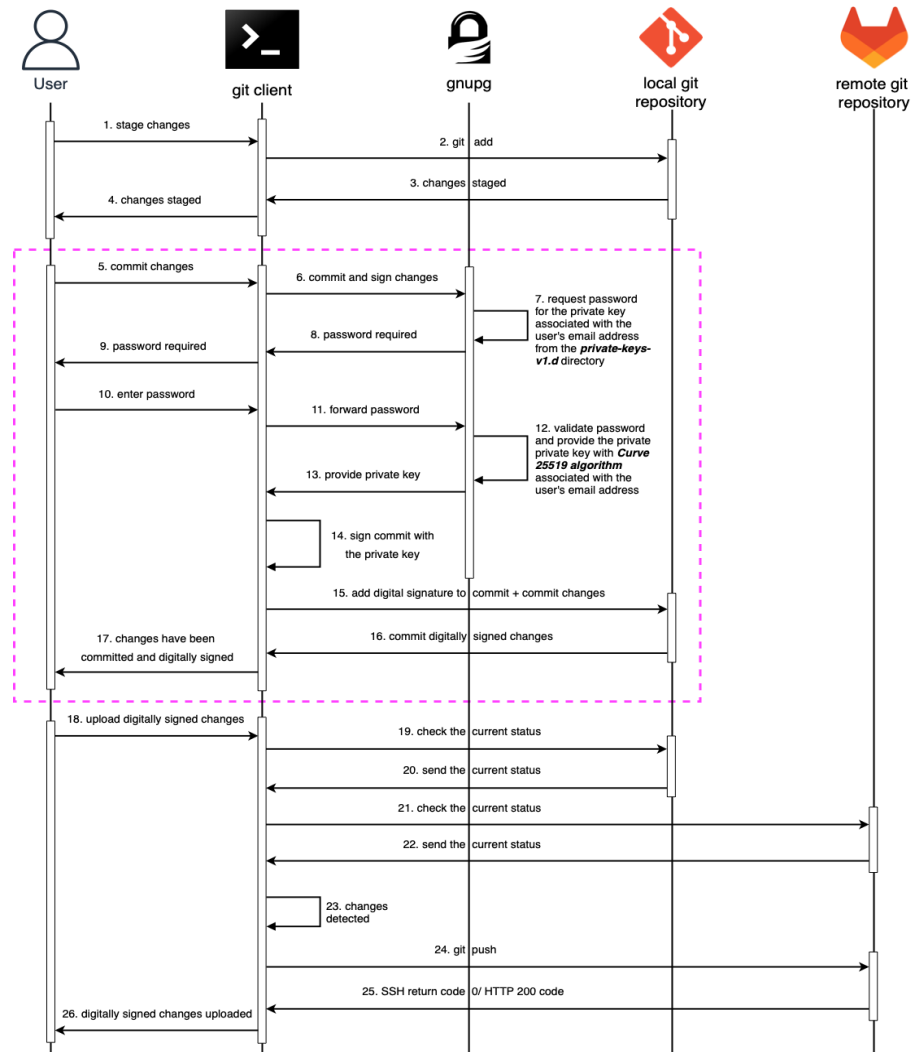


Requesting AWS Temporary Credentials Flow (Unauthorized User)

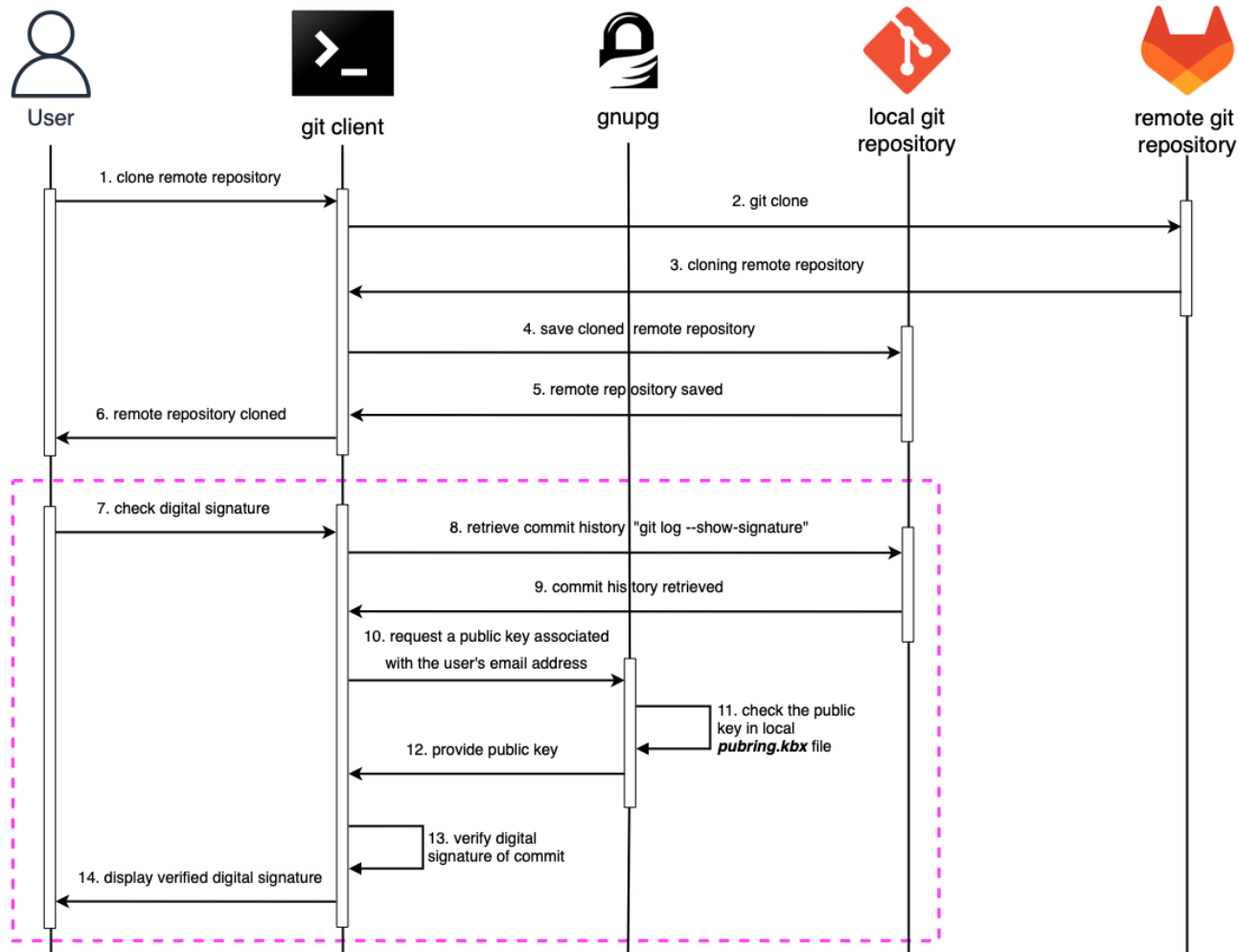
Note:
Amazon Cognito Identity Pool must be created (by AWS administrator).



Commit Signing Flow



Commit Verifying Flow



Spoofing git commits to change history

Once you have write access to a git repository, it is child's play to spoof commits pretending to be someone else. Here's a step-by-step on how to do it and how to protect against it.



<https://medium.com/@pjbgr/spoofing-git-commits-7bef357d72f0>



NOTE:

- Tested on macOS and WSL.
- Conditions for *git profiles* are not implemented yet.
- AWS resources created manually.
- A proper test scenarios must be carried out.

Conclusion



What?

- Allows to *verify* the source code origin.



Why?

- Supports *security fundamentals* (authenticity, integrity, non-repudiation) and *compliance* (e.g., *NIST 800-53*, *PCI DSS Version 4.0*, *GDPR*, *ISO 27001:2022*).
- Detects *injection of the malicious* code or publishing code from an unauthorised origin.
- Helps to *mitigate data* and/ or *financial loss*, and the *reputation* damage.
- Increases the *trustworthiness* and *professionalism*.



How?

- Developers can *set up* commit signing *in less than 1 minute* by running a shell script.

Demo Time

Q&A Session

Contact



Email: *ondrej.halaska@ti&m.ch*

LinkedIn: *www.linkedin.com/in/ondrej-h*

We digitalize your company.

Thank you!

Ondrej Halaska
DevOps Engineer
ondrej.halaska@ti8m.ch

ti8m.com

ti&m AG
Buckhauserstrasse 24
8048 Zurich
SWITZERLAND
+41 44 497 75 00

ti&m AG
Monbijoustrasse 68
3007 Bern
SWITZERLAND
+41 31 960 15 55

ti&m AG
Innere Margarethenstrasse 5
4051 Basel
SWITZERLAND
+41 61 501 29 99

ti&m GmbH
Schaumainkai 91
60596 Frankfurt am Main
GERMANY
+49 69 24745268-0

ti&m GmbH
Kesselstrasse 3
40221 Dusseldorf
GERMANY
+49 211 90989580

ti&m Pte. Ltd.
18 Robinson Road #15-16
Singapore 048547
SINGAPORE
+65 6983 9530

ti&m