# Teamwork

## Git Ecosystem & Security

Thomas Domingues

# Summary

- Introduction
- GitHub vs GitLab
- Security
  - Branches protection
  - Project protection / Fork
  - SSH
  - GPG
- Project

# Introduction

Now you’re convinced Git is a great tool for version control, let’s see how can we access to project management features (which it doesn’t support) like:

- Authentication management and user rights
- Merge requests, issues, tracker…
- CI/CD

Answer: collaborative platforms (or social networks for developers)

# GitHub VS GitLab

What to choose between the two (wonderful) solutions? Depends on your tastes

## GitHub

- Bigger community
- Better UI
- Closed source and owned by Micro$oft

## GitLab

- Open Source
- Can be self-hosted
- More native functionalities (including CI)
- Unlimited free private repositories

# Security

Playing with secret (or important) development can be very risky if proper security measures are not taken into account. That’s why you only need to remind this simple operation:

GIT + SSH + GPG ~~= GITSSHGPG~~ = ❤️

# Branches protection

- First of all, as seen before with the Git Flow, some branches must not be directly touched (except for project init): Master, Develop and Release shouldn’t be never edited by anyone directly but with merge requests.
- Both GitHub (user/repo/settings/branches) and GitLab (user/repo/-/settings/repository) allow branches protection.

# Branches protection

What you'll see

```
user@/git-project$ git push github master
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote: error: GH006: Protected branch update failed for refs/heads/master.
remote: error: Cannot force-push to this protected branch
To github.com:user/repo.git
! [remote rejected] master -> master (protected branch hook declined)
error: failed to push some refs to 'git@github.com:user/repo.git'
```

# Project protection

- As a developer, you don’t want anyone to pollute your repository with random multiple new branches.
- By default, only the user / group owner has write access to the repo, but that can be modified in both GitHub (user/repo/settings/access) and GitLab (user/repo/-/project_members).

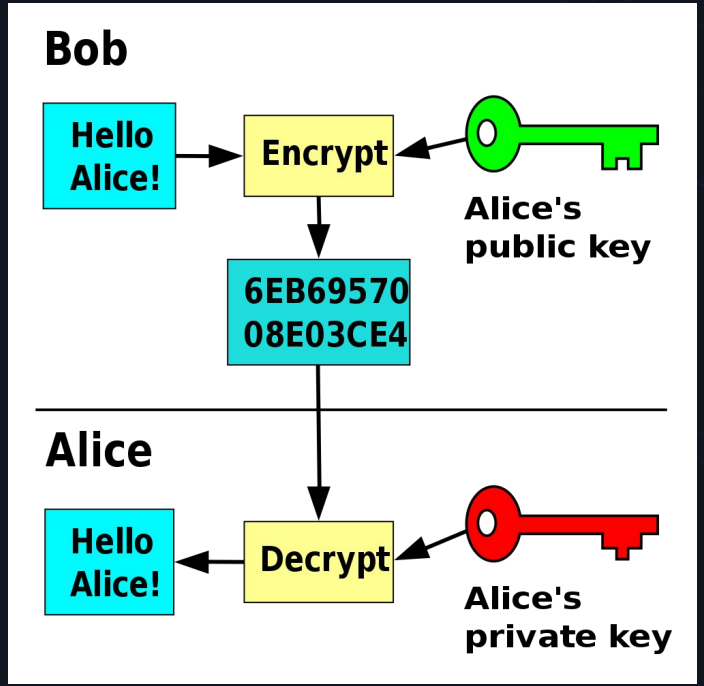
# Project protection - Fork

To collaborate to a project, the best way is to:

- Fork (make a copy in your user space) the project
- Create and push the branch into your fork
- Create a Merge Request from the branch to the original project (link done by default by GitHub and GitLab at MR creation)

# SSH

- SSH provides a secure channel over an unsecured network by using a client–server architecture, connecting an SSH client (you) with an SSH server (GitHub / GitLab)
- So a connection (transport) is open and encrypted to allow secured file exchange (at git pull and push)



# SSH

- Main issue with SSH: You can put any name you want.
- If you want to commit as Thomas Domingues, you can just change your git config as seen before:
  - `git config --global user.email "tdomingues@myges.fr"`
  - `git config --global user.name "Thomas Domingues"`
- To verify the author of the commit, another protocol is necessary: GPG

# GPG

- Same encryption / decryption process, but different purposes
- GPG allows to sign the commits and confirm to GitHub / GitLab that you're actually the author
- Two ways to sign commits
  - Always by default: `git config --global user.signingkey [YOUR_KEY_ID]`
  - Commit by commit: `git commit -S -m "Signed commit"`

# SSH & GPG

Generate your own SSH & GPG keys

# Project

As groups of 3 or 4 (see MyGES), do the following project:

- Start a random project (Symfony, RoR, Node ...)
- Setup and work on the repo following the Git Flow method:
    - Initialization
    - Branches (master, develop, release, feature)
    - Correct merges on master and develop
    - Tags
- Write issues of what you plan to implement, assigning one dev per issue

# Project

- Do at least one signed commits per developer. Here are the minimum requirements:
    - README.md explaining how to run your project
    - CONTRIBUTE.md explaining how to contribute (the content must follow Git Flow method)
- Write proper commit messages describing the feature implemented
- Protect project and important branches, still must be public and accessible by anyone

Expected delivery on MyGES: Only GitHub / GitLab link

Due date: Wednesday 20th - 9:45AM