Create Own GitLab Testing Environment For wikisuite-packages

By following this guide, you will learn how to create a dedicated testing environment using your personal GitLab account. This will allow you to test changes to the WikiSuite installer (WikiSuite packages) and verify that the changes work as expected before being deployed to packages.wikisuite.org.

Let's get started!

1. Create a project on Gitlab and add wikisuite-packages source code.

- Follow this link to see how to create a gitlab blank project: https://docs.gitlab.com/ee/user/project.
- Once the project is created, clone it on your computer.
- Download the ZIP file of the wikisuite-packages source code on https://gitlab.com/wikisuite/wikisuite-packages.
- Extract the ZIP and move all the code into the source tree of your project you just cloned.

Now leave your project as is; we'll come back to it later...

2. Create a new, dedicated GPG key pair.

Important :
During the creation of the key, you will provide certain necessary information for its generation. Ensure that you do not have a key created with the same information you are providing. This could lead to pipeline errors when using the key on GitLab.

Open your teminal and enter the following command:

```
gpg --full-generate-key --openpgp
```

This command generates a key pair that consists of a public and a private key. A series of prompts directs you through the process:

- Select what kind of key you prefer:

```
Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
  (14) Existing key from card
```

```
Your selection?
```

Choose An RSA/RSA key that allows you not only to sign communications, but also to encrypt files.

- Choose the key size:

```
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want?  (3072)
```

You can take the default value.

- Specify how long the key should be valid:

```
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for?
```

- Valide the provided infos:

```
Is this correct? (y/N)
```

Enter **y** to finish the process.

- Enter your name and email address and comment for your GPG key:

```
GnuPG needs to construct a user ID to identify your key.

Real name: your real name
Email address: example@gmail.com
Comment: your comment
```

- Confirme your informations:

```
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit?
```

Enter the letter **O** to continue if all entries are correct.

- Passphrase for your secret key:

Important :
You will be asked for a passphrase to create your key. **Note that the GPG key should not have a passphrase**, leave it blank whenever prompted.

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

Please enter the passphrase to protect your key:
```

After that, the key is created and some information is displayed in the terminal

```
public and secret key created and signed.

pub   rsaXXXX 2023-11-30 [SC]
      xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
uid                      your real name <example@gmail.com>
sub   rsaXXXX 2023-11-30 [E]
```

Be sure to keep your uid (in our example: **your real name <example@gmail.com>**), it will be useful to you later.

- Export the private key as ASCII.

```
gpg --output private.pgp --armor --export-secret-key "your uid"
```

This will generate a private.pgp file containing your private key. Please make sure to not store/commit the private key into the source tree.

3. Configure your GitLab project.

On your project page, under the "Settings > CI/CD Pipelines", create a secret variable
Important: Your secret variable should not be protected, unchecked protected checkbox.

- The first one be called GPG_PRIVATE_KEY and copy/paste all the content of the private.pgp file in the value field.
- Create a second secret variable called SIGN_USER, whose value will be the user_ID (uid in our example: **your real name <example@gmail.com>**) of your private key.

4. Commit and push to main.

Now, let's get back to your project.

- On your project, create and checkout to a new branch main if not yet:

```
git checkout -b main
```

- Add and commit all your changes:

```
git add .
git commit -m "a message"
```

- Push to your repository:

```
git push origin main
```

or

```
git push --set-upstream origin main
```

if the branch does not exist on your GitLab repository.

After commit and push, once the pipilines have passed, the packages are automatically built by Gitlab-CI, then hosted on Gitlab-Pages. The access link for the packages is available under the "Deploy > Pages" section of your GitLab project.

5. Configure wikisuite-installer script.

The final step is to modify the wikisuite-installer script so that it can point to your project.

- Find the link to your project pages under the "**Deploy > Pages > Access pages**" section of your GitLab project.
- In your project's source tree, open the wikisuite-installer file. Find the

WIKISUITE_DEV_INSTALL_FROM_FOLDER var and change its value by to the link to your project pages.

Example:

if your link is: https://my-ws-packages-test.gitlab.io

```
WIKISUITE_DEV_INSTALL_FROM_FOLDER="https://my-ws-packages-test.gitlab.io"
```

- On the same file find:

```
run "cat ${WIKISUITE_DEV_INSTALL_FROM_FOLDER}/GPG_PUBLIC_KEY
```

and replace with:

```
run "curl -s -S -L ${WIKISUITE_DEV_INSTALL_FROM_FOLDER}/GPG_PUBLIC_KEY
```

- And then find:

```
run "echo 'deb file:${WIKISUITE_DEV_INSTALL_FROM_FOLDER}/${os_type}
```

and replace with:

```
run "echo 'deb ${WIKISUITE_DEV_INSTALL_FROM_FOLDER}/${os_type}
```

- Add, commit and push to main

```
git add wikisuite-installer
git commit -m "Update wikisuite-installer"
git push origin main
```

tips

1. How to test.

- Download the installation script to your server

```
curl -o wikisuite-installer link_to_your_wikisuite-installer_script
```

- Run the installation script

```
sudo bash wikisuite-installer
```

2. Revert all your changes and update your repository with the original wikisuite-packages

- Add an upstream that points to the original wikisuite-packages repository

```
git remote add upstream git@gitlab.com:wikisuite/wikisuite-packages.git
```

- Fetch the latest changes from the original wikisuite-packages repository

```
git fetch upstream
```

- Merge the changes from the original repository into your local branch

```
git merge upstream/main
```

- Push the changes from your local branch to your repository

```
git push origin main
```