Установка и настройка tulipe/jenkins/kallithea

Дороничев-Тедерсон Даниил, гр. ИВТ-32-БО

daniildorted@gmail.com

1 Используемые средства

- Ubuntu Linux 20.04
- Docker v19.03.13
- docker-compose v1.27.4

2 Описание файла docker-compose.yml

```
version: '3.2'
services:
    jenkins:
        container_name: jenkins
        build:
            context: jenkins
            dockerfile: Dockerfile-alpine
            args:
               JENKINS_VERSION: 2.235
    ports:
        - 8080:8080
        - 50000:50000
    volumes:
        - /tmp:/var/jenkins_home/secrets/
    tuleap:
        container_name: tuleap
        build:
            context: tuleap
     kallithea:
        container_name: kallithea
        build:
        context: kallithea
    ports:
        - 3080:80
```

Рис. 1: структура файла docker-compose.yaml

На рис. 1 можно видеть структуру файла docker compose. На каждый из трёх компонентов приходится свой компонент `services`. Контекстом каждого компонента по сути является отдельная директория. Для gitea и openproject используются Docker файлы представленные в master ветках репозиториев. Для jenkins в свою очередь используется модифицированный Dockerfile на основе alpine (модификация заключалась в изменении версии на более новую, а также удаление хеша проверки целостности SHA512, так как присутствующий хэш относился к старой версии jenkins). Необходимость наличия директивы `volumes` в описании jenkins обусловлена тем, что нам необходимо получить секретный одноразовый ключ необходимый при первом запуске всей инсталяции. Все порты проброшены согласно стандартной документации на каждый продукт, так имеем:

– jenkins работает на порту 8080 (web) и 50000 (jenkins agent)

– kallithea работает на порту 3080 (web)

– tuleap работает на порту 80 (web), но tuleap работает на спицифичном домене tuleap.local

3 Скрипт резервного копирования

```
#!/usr/bin/env bash

        echo "Runing backup..."
        for i in \
         $(docker ps --filter "name=kallithea|jenkins|tulipe" | tail -n+2 | awk '{  print $1
          ↪  }'); do docker export -o ${i}-ci-$(date +"%m-%d-%Y").tar ${i}; \
        done
```

Рис. 2: Скрипт резервного копирования

Скрипт представлен на рис.2. По сути данный скрипт производит фильтрацию запущенных контейнеров по имени, и экспортирует запущенные экземпляры в tar архивы с датой и временем. Далее архивы можно загрузить вновь используя команду `docker load -i <archive_name.tar>`

4 Демонстрация установки

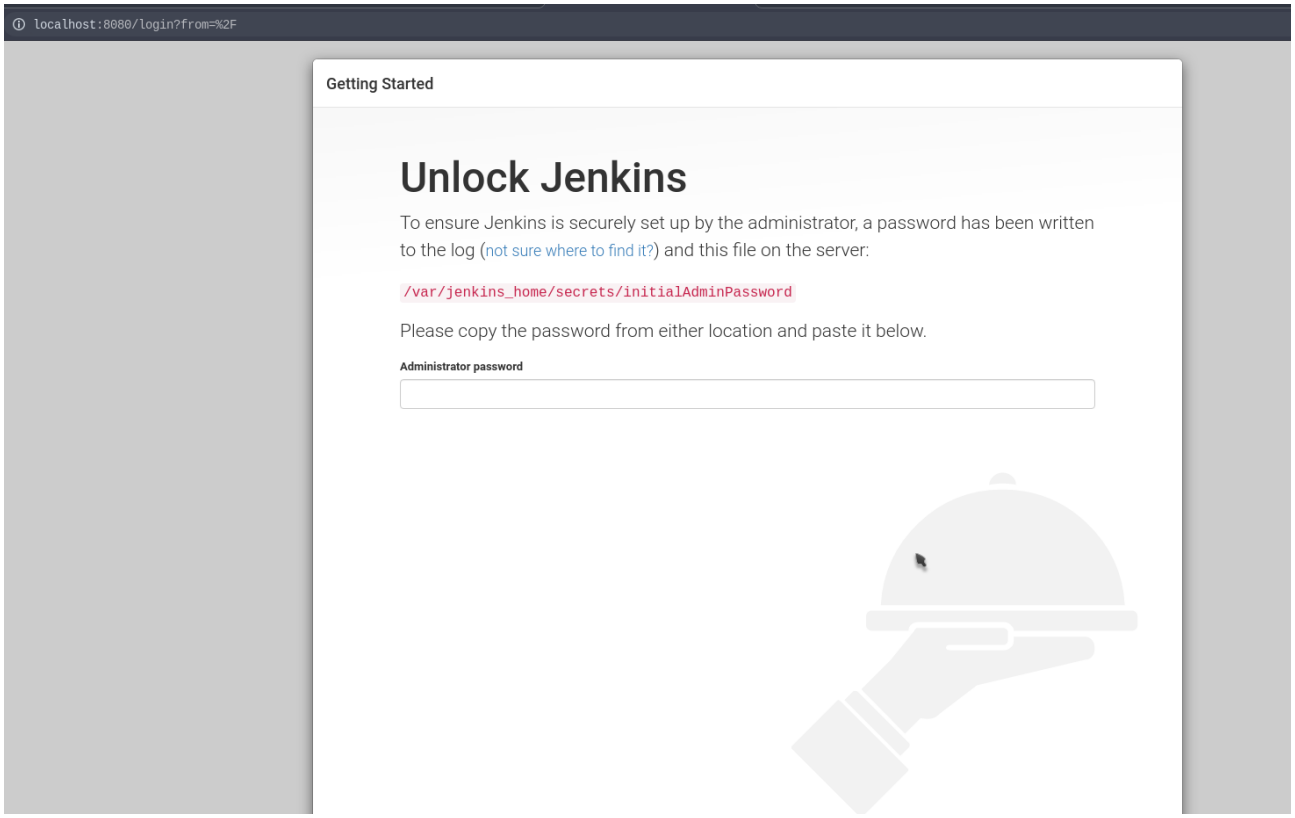

Рис. 3: Начальный экран jenkins, с запросом уникального ключа

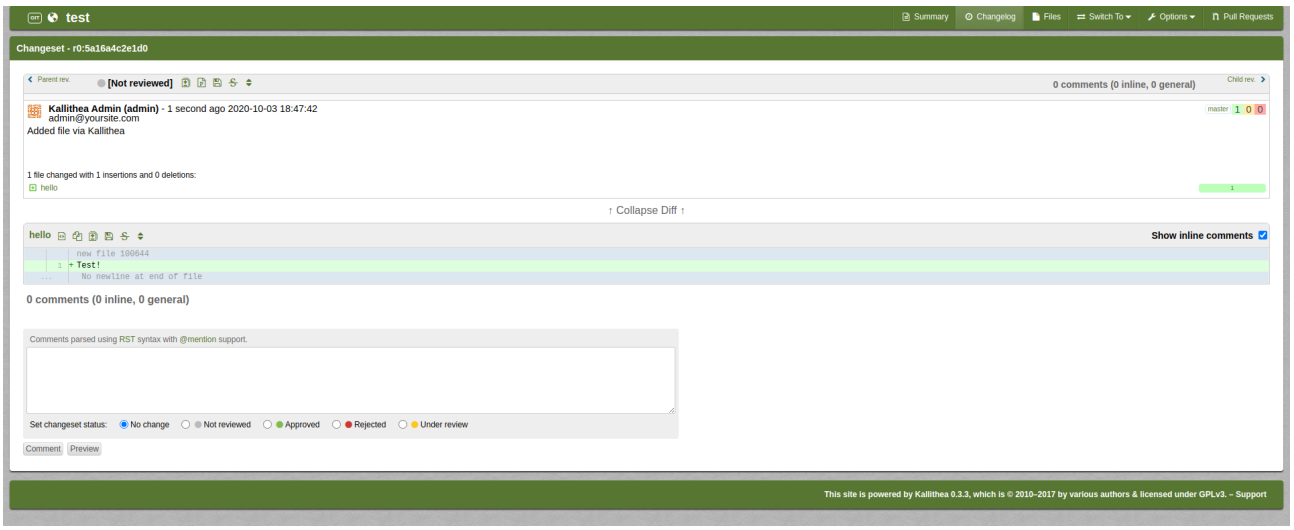

Рис. 4: Пример интерфейса kallithea с коммитом

Стоит сделать уточнительное пояснение. Для интеграции необходимо указать полный http-адрес до репозитория. Формально инсталяция kallithea расположена по адресу `localhost:8090`. Однако данный адрес не является внутренним адресом подсети докера, для того-чтобы подключения прошло успешно необходим именно внутренний адрес. В данном случае он был выяснен путём запуска команды `docker inspect ${CONTAINER` где `${CONTAINER_ID}` — идентификатор контейнера, в котором запущена инсталяция kallithea. В качестве данных для авторизации используются данные пользователя admin. Тоже самое необходимо сделать для контейнера с инсталяцией tulipe, это необходимо для ассоциации домена tulipe.local с реальным ip адресом контейнера путём редактирования файла /etc/hosts непосредственно на хост-машине.

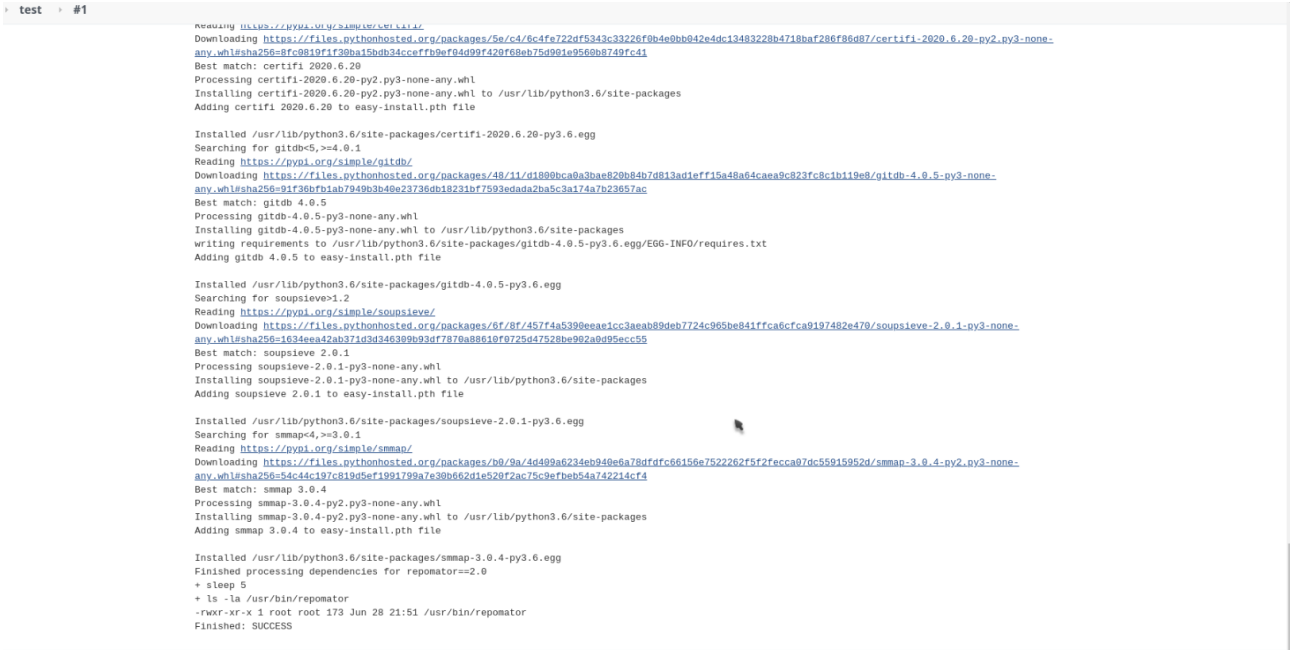

Рис. 5: Пример успешной сборки проекта с помощью интеграции

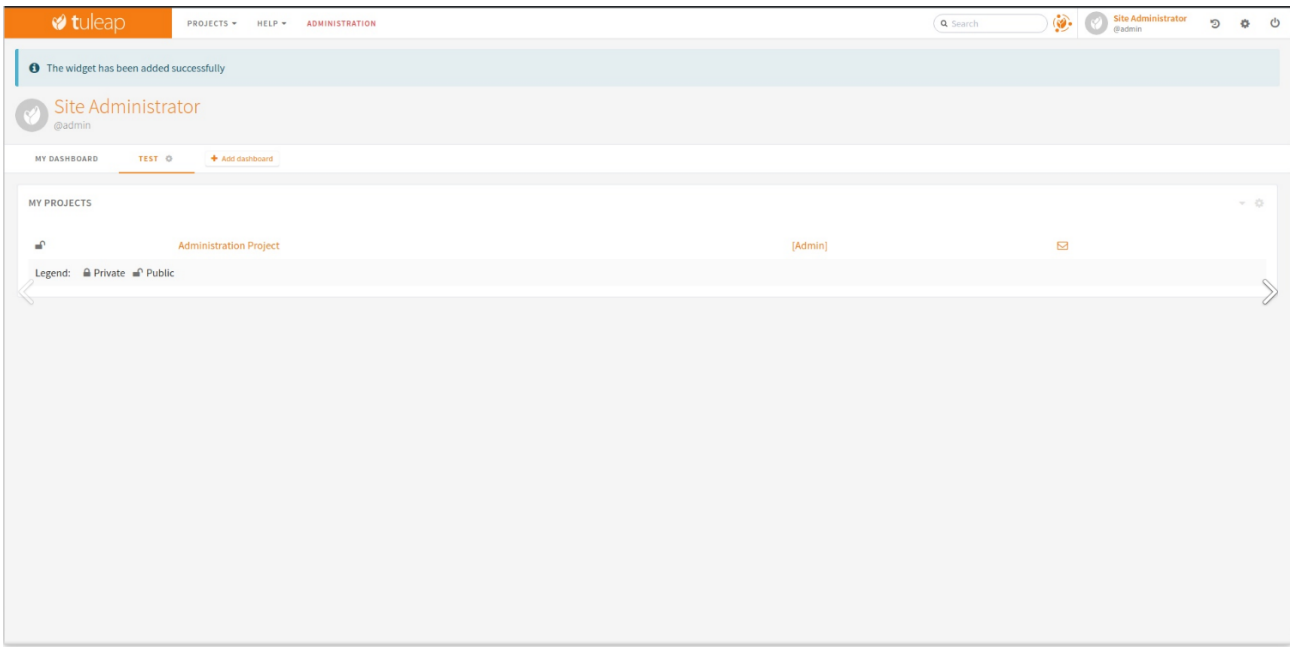

Рис. 6: Пример создания тикета в tulipe

К сожаление, согласно документации, (а также форумам поддержки всех трёх проектов), не существует интеграции kallithea с tulipe или tulipe с jenkins.