

ЛР 1: [C++ UNIX]: UNIX знакомство: useradd, nano, chmod, docker, GIT, CI, CD

Готов Константин
Z33431

Цель работы

Познакомить студента с основами администрирования программных комплексов в ОС семейства UNIX, продемонстрировать особенности виртуализации и контейнеризации, продемонстрировать преимущества использования систем контроля версий (на примере GIT)

1 [ОС] Работа в ОС, использование файловой системы, прав доступа, исполнение файлов

1.1 В папке /USR/LOCAL/ создать 2 директории: folder_max, folder_min

```
1 cd /usr/local/  
2 mkdir folder_max folder_min
```

1.2 Создать 2-х группы пользователей: group_max, group_min

```
1 sudo groupadd group_max  
2 sudo groupadd group_min
```

1.3 Создать 2-х пользователей: user_max_1, user_min_1

```
1 sudo useradd user_max_1  
2 sudo useradd user_min_1  
3 # sudo passwd user_max_1  
4 # sudo passwd user_min_1  
5  
6 sudo usermod -a group_max user_max_1  
7 sudo usermod -a group_min user_min_1
```

1.4 Для пользователей из группы *_max дать полный доступ на директории *_max и *_min. Для пользователей группы *_min дать полный доступ только на директорию *_min

1.5 Создать и исполнить (пользователем из той же категории) скрипт в директории folder_max, который пишет текущую дату/время в файл output.log в текущей директории

```
1 vim folder_max/script.sh
```

```
1 date >> log.log
```

Листинг 1: script.sh

```
1 su user_max_1
2 cd /usr/local/folder_max/
3 ./script.sh
```

- 1.6 Создать и исполнить (пользователем из той же категории) скрипт в директории `folder_max`, который пишет текущую дату/время в файл `output.log` в директории `*_min`

```
1 su user_max_1
2 cd /usr/local/folder_min/
3 ../../folder_max/script.sh
```

- 1.7 Исполнить (пользователем `*_min`) скрипт в директории `folder_max`, который пишет текущую дату/время в файл `output.log` в директории `*_min`

```
1 su user_min_1
2 cd /usr/local/folder_min/
3 ../../folder_max/script.sh
```

- 1.8 Создать и исполнить (пользователем из той же категории) скрипт в директории `folder_min`, который пишет текущую дату/время в файл `output.log` в директории `*_max`

```
1 su user_min_1
2 cd /usr/local/folder_max/
3 ./script.sh
```

- 1.9 Вывести перечень прав доступа у папок `*_min/` `*_max`, а также у всего содержимого внутри

```
1 ls -l
```

2 [КОНТЕЙНЕР] docker build / run / ps / images

- 2.1 Создать скрипт, который пишет текущую дату/время в файл `output.log` в текущей директории

```
1 date > output.log
```

2.2 Собрать образ со скриптами выше и с пакетом nano

```
1 FROM ubuntu:18.04
2 WORKDIR /
3 COPY folder_max folder_max
4 COPY folder_min folder_min
5 COPY script.sh .
6
7 RUN apt-get update
8 RUN apt-get install nano
```

2.3 Собрать и запустить образ (docker run)

```
1 sudo docker build -t echo .
2 sudo docker run -it echo
```

2.4 Вывести список пользователей в собранном образе

```
1 less /etc/passwd
```

При выводе списка пользователей в Docker не было пользователей созданных ранее, что логично.

3 [GIT] GitHub / GitLab, в котором будут содержаться все выполненные ЛР

3.1 Создать репозиторий в GitHub или GitLab. Создать структуру репозитория:

```
1 mkdir CPP-labs
2 cd CPP-labs
3 mkdir lab_01 utils
4 cd lab_01
5 mkdir build src doc
6 git init
7 git push
```

3.2 Создать ветки dev / stg / prd, удалить ранее существующие ветки удаленно и локально

```
1 git checkout -b dev
2 git checkout -b stg
3 git checkout -b prd
4 git checkout dev
```

3.3 Создать скрипт автоматического переноса ревизий из ветки dev в ветку stg с установкой метки времени (tag). Скрипт в корень репозитория

```

1 git checkout stg
2
3 echo "=====
4 echo "Merging develop branch"
5 echo "=====
6
7 git merge dev
8
9 timestamp=$(date +%m_%d_%Y_%H_%M")
10 echo "=====
11 echo "Tagging as ${timestamp}"
12 echo "=====
13 git tag -a "$timestamp" -m "dev2stg"
14
15 echo "=====
16 echo "Pushing commits and tags"
17 echo "=====
18 git push --set-upstream origin stg
19 git push --tags
20
21 echo "=====
22 echo "Checking out develop branch"
23 echo "=====
24 git checkout dev
25 git push

```

3.4 Создать скрипт автоматического переноса ревизий из ветки stg в ветку prd с установкой метки времени (tag). Скрипт в корень репозитория

```
1 git checkout prd
2
3 echo "===== "
4 echo "Merging stg branch"
5 echo "===== "
6
7 git merge stg
8
9 timestamp=$(date +%m_%d_%Y_%H_%M)
10 echo "===== "
11 echo "Tagging as ${timestamp}"
12 echo "===== "
13 git tag -a "${timestamp}" -m "stg2prd"
14
15 echo "===== "
16 echo "Pushing commits and tags"
17 echo "===== "
18 git push --set-upstream origin prd
19 git push --tags
20
21 echo "===== "
22 echo "Checking out develop branch"
23 echo "===== "
24 git checkout dev
25 git push
```

4 Выводы

В ходе лабораторной работы было произведено ознакомление с основными командами ОС семейства UNIX. Была осуществлена работа с контейнеаризацией. Также была произведена работа с системой контроля версий и создан репозиторий на GitHub. В целом, лабораторную работу считаю выполненной успешно.