

ФГБОУ ВО

Уфимский Государственный Авиационный Технический Университет

Кафедра ВВТИС

Отчет по

лабораторной работе №4

«РАБОТА С ПОЛЬЗОВАТЕЛЯМИ В LINUX. СОЗДАНИЕ МАКЕ-
ФАЙЛОВ.»

по дисциплине «Операционные системы и сети»

Выполнил:
ст. гр. МКН-216
Биганяков А.А.

Проверил:
старший преподаватель
Ямилева А.М.

Уфа 2022

Цель работы: Целью работы является изучение утилиты make для автоматизации сборки проектов и работа с пользователями в Linux.

Ход работы:

```
sumashedshiy@sumashedshiy-VirtualBox:~$ sudo adduser biganyakov
[sudo] пароль для sumashedshiy:
Добавляется пользователь «biganyakov» ...
Добавляется новая группа «biganyakov» (1001) ...
Добавляется новый пользователь «biganyakov» (1001) в группу «biganyakov» ...
Создаётся домашний каталог «/home/biganyakov» ...
Копирование файлов из «/etc/skel» ...
Новый пароль :
Повторите ввод нового пароля :
passwd: пароль успешно обновлён
Изменение информации о пользователе biganyakov
Введите новое значение или нажмите ENTER для выбора значения по умолчанию
    Полное имя []: BiganiakovArslan
    Номер комнаты []:
    Рабочий телефон []:
    Домашний телефон []:
    Другое []:
Данная информация корректна? [Y/n] Y
sumashedshiy@sumashedshiy-VirtualBox:~$
```

Задание 2. Создание нового пользователя

```
biganyakov@sumashedshiy-VirtualBox:~$
```

Задание 3. Запуск терминала от нового пользователя



```
main.cpp                                     twonull.h
1 int twonull(int m, int n)
2 {
3     int n1=0, m1=0, k=0, ostM, ostN;
4     while (m!=0)
5     {
6         ostM=m%2;
7         m/=2;
8         if (ostM==0)
9             m1++;
10    }
11    while (n!=0)
12    {
13        ostN=n%2;
14        n/=2;
15        if (ostN==0)
16            n1++;
17    }
18    if (m1>n1)
19        k=m1;
20    else
21        k=n1;
22    return k;
23 }
```

```

main.cpp
1 #include <iostream>
2 #include "twonull.h"
3 using namespace std;
4 int main()
5 {
6     int m,n;
7     cin >> m >> n;
8     int z = twonull(m,n);
9     cout << z << endl;
10 }

```

```

biganyakov@sumashedshiy-VirtualBox:~$ g++ main.cpp -o main
biganyakov@sumashedshiy-VirtualBox:~$ ./main
10
20
3
biganyakov@sumashedshiy-VirtualBox:~$

```

Задание 4. Создание программы на C++ состоящая из двух модулей

```

GNU nano 4.8
makefile
compile: main.cpp twonull.h
    g++ -o compile main.cpp
clear:
    rm compile
install:
    make compile
    cp ~/compile ~/app

```

```

biganyakov@sumashedshiy-VirtualBox:~$ make compile
g++ -o compile main.cpp
biganyakov@sumashedshiy-VirtualBox:~$ make clear
rm compile
biganyakov@sumashedshiy-VirtualBox:~$ make install
make compile
make[1]: вход в каталог «/home/biganyakov»
g++ -o compile main.cpp
make[1]: выход из каталога «/home/biganyakov»
cp ~/compile ~/app
biganyakov@sumashedshiy-VirtualBox:~$

```

```

biganyakov@sumashedshiy-VirtualBox:~$ ./compile
10
15
2
biganyakov@sumashedshiy-VirtualBox:~$

```

Задание 5. Написание скрипта для утилиты make

Вывод: В данной лабораторной работе были изучены утилиты make для автоматизации сборки проектов и работа с пользователями в Linux.

Ответы на контрольные вопросы:

1. Какие типы пользователей существуют в Linux

Ответ: В Linux существуют три типа пользователей:

- Администраторы — привилегированные пользователи с полным доступом к системе. По умолчанию на Linux-сервере после установки операционной системы всегда есть один такой пользователь — root.
- Локальные пользователи — непривилегированные пользователи. Их учётные записи создаёт администратор. Особенность таких аккаунтов в ограниченном доступе к серверу — они не могут пользоваться некоторыми системными утилитами, могут работать только с теми файлами и папками, к которым им открыт доступ.
- Системные пользователи — учётные записи, автоматически создаваемые системой для работы внутренних процессов и служб. Например, после установки веб-сервера Apache, который входит в состав стека LAMP, на сервере появляется пользователь www-data (на CentOS apache), от имени которого потом работает веб-сервер. Такие пользователи нужны для повышения безопасности.

2. Где хранится вся информация и пользователей системы Linux?

Ответ: Вся информация о пользователях хранится в следующих файлах:

- passwd (etc/passwd) - содержит информацию о пользователях, имеет следующий формат записи - "user_name:password:UID:GID:full_name:home_directory:login_shell". Элементы записи должны разделяться символом - ":" (двоеточие) и записываются без пробелов. Если пароль хранится в зашифрованном виде в файле /etc/shadow, то вместо пароля указывается - "x".
- group (etc/group) - информация о группах, формат - "group_name:password:GID:user1,user2,user3". Элементы записи должны разделяться символом - ":" (двоеточие) записываются без пробелов. Имена пользователей записываются через запятую. У файлов /etc/passwd и /etc/group всегда определенные права доступа: чтение и запись для root, для остальных только чтение.
- shadow (etc/shadow) - в этом файле хранятся так называемые "теневые пароли", информация о паролях пользователей в зашифрованном виде. Сделано это для безопасности, так как файл /etc/passwd может читаться кем угодно, а файл /etc/shadow может прочитать только root.
- gshadow (etc/gshadow) - то же самое что и shadow, только для паролей групп.

3. Какие команды позволяют создать и изменять пользователей в Linux?

Ответ: Для управления пользователями используются следующие команды:

- `useradd` или `adduser` - добавить нового пользователя.
- `passwd` - задать пароль для пользователя.
- `usermod` - изменить параметры учетной записи пользователя.
- `userdel` или `deluser` - удалить учетную запись пользователя.

Для управления группами используются следующие команды:

- `groupadd` - добавляет новую группу.
- `gpasswd` - устанавливает пароль группы.
- `groupmod` - изменение параметров группы.
- `groupdel` - удаление группы.

4. Для чего нужна утилита `make`?

Ответ: `make` — утилита предназначенная для автоматизации преобразования файлов из одной формы в другую. Правила преобразования задаются в скрипте с именем `Makefile`, который должен находиться в корне рабочей директории проекта. Сам скрипт состоит из набора правил, которые в свою очередь описываются:

- целями (то, что данное правило делает);
- реквизитами (то, что необходимо для выполнения правила и получения целей);
- командами (выполняющими данные преобразования).

5. Что такое фиктивные цели в `make`? Для чего они нужны?

Ответ: Командой `make` производят компиляцию программы, командой `make install` — установку. Такой подход весьма удобен, поскольку все необходимое для сборки и развертывания приложения в целевой системе включено в один файл (забудем на время о скрипте `configure`). Обратите внимание на то, что в первом случае мы не указываем цель, а во втором целью является вовсе не создание файла `install`, а процесс установки приложения в систему. Прodelывать такие фокусы нам позволяют так называемые фиктивные (phony) цели. Вот краткий список стандартных целей:

- `all` — является стандартной целью по умолчанию. При вызове `make` ее можно явно не указывать.
- `clean` — очистить каталог от всех файлов полученных в результате компиляции.
- `install` — произвести инсталляцию
- `uninstall` — и деинсталляцию соответственно.