

EPAM University Programs
DevOps external course
Module 4 Linux Essentials with Bash
TASK 4.11

All scripts are available on my Git in dir 4.12_task

4.12.1 Создать автоматический генератор паролей пользователей. На вход скрипта подать файл users.txt в котором содержится список пользователей:

1. user1

2. user2

...

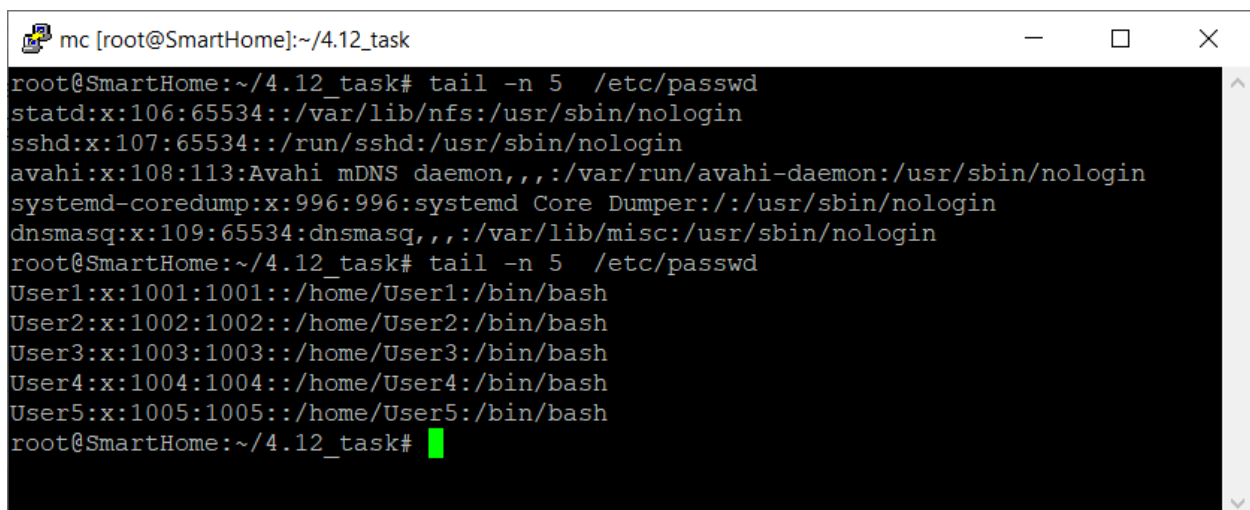
20. user20

Автоматически сгенерировать пароли для всех пользователей, создать в скрипте пользователей linux-системы со сгенерированными паролями, так чтобы вы могли войти под именем каждого из пользователей с созданным для него паролем (тут предполагается работа с openssl), а также создать для каждого пользователя файл user*-login-password.txt, в который поместить имя пользователя и сгенерированный пароль. Например:

user1 – uR44y6!#

script name (4.12.1 sub folder): add_users.sh

/etc/passwd before and after script execution:



```
mc [root@SmartHome]:~/4.12_task
root@SmartHome:~/4.12_task# tail -n 5 /etc/passwd
statd:x:106:65534::/var/lib/nfs:/usr/sbin/nologin
sshd:x:107:65534::/run/sshd:/usr/sbin/nologin
avahi:x:108:113:Avahi mDNS daemon,,:/var/run/avahi-daemon:/usr/sbin/nologin
systemd-coredump:x:996:996:systemd Core Dumper:/:/usr/sbin/nologin
dnsmasq:x:109:65534:dnsmasq,,:/var/lib/misc:/usr/sbin/nologin
root@SmartHome:~/4.12_task# tail -n 5 /etc/passwd
User1:x:1001:1001::/home/User1:/bin/bash
User2:x:1002:1002::/home/User2:/bin/bash
User3:x:1003:1003::/home/User3:/bin/bash
User4:x:1004:1004::/home/User4:/bin/bash
User5:x:1005:1005::/home/User5:/bin/bash
root@SmartHome:~/4.12_task#
```

Script output and tests (5 users in users.txt file):

```
User1@SmartHome: ~
4.11_task/ 4.12_task/
root@SmartHome:~# cd 4.1
4.11_task/ 4.12_task/
root@SmartHome:~# cd 4.12_task/
root@SmartHome:~/4.12_task# ls
add_users.sh  users.txt
root@SmartHome:~/4.12_task# ./add_users.sh users.txt
New password: Retype new password: passwd: password updated successfully
New password: Retype new password: passwd: password updated successfully
New password: Retype new password: passwd: password updated successfully
New password: Retype new password: passwd: password updated successfully
New password: Retype new password: passwd: password updated successfully
root@SmartHome:~/4.12_task# ls
add_users.sh      User3-login-password.txt  users.txt
User1-login-password.txt  User4-login-password.txt
User2-login-password.txt  User5-login-password.txt
root@SmartHome:~/4.12_task# cat User1-login-password.txt
User1 - M1VhxdYV
root@SmartHome:~/4.12_task# su - pi
pi@SmartHome:~$ su - User1
Password:
User1@SmartHome:~$ whoami
User1
User1@SmartHome:~$ █
```

4.12.2 Взять за основу проект <https://habr.com/ru/post/155201/> . Написать скрипт выполняющий следующее:

1. При первоначальном запуске – вычисление контрольных сумм и архивация проекта.
2. периодическая проверка проекта на предмет изменений.

Via Cron for current user –

each 5 minutes:

crontab -e

`*/5 * * * * /root/task/4.12.2/simple_pipeline.sh > /dev/null 2>&1`

3. Если проект изменился, то записать новую версию в новый архив и запустить перекомпиляцию проекта.

I created my makefile (project mf is in 4.12.2 subfolder)

```
192.168.1.99 - PuTTY
root@SmartHome:~/4.12_task/4.12.2/mf# cat makefile
# Это комментарий, который говорит, что переменная CC указывает компилятор, используемый для сборки
CC=g++
#Это еще один комментарий. Он поясняет, что в переменной CFLAGS лежат флаги, которые передаются компилятору
CFLAGS=-c -Wall

all: hello

hello: main.o factorial.o hello.o
    $(CC) main.o factorial.o hello.o -o hello

main.o: main.cpp
    $(CC) $(CFLAGS) main.cpp

factorial.o: factorial.cpp
    $(CC) $(CFLAGS) factorial.cpp

hello.o: hello.cpp
    $(CC) $(CFLAGS) hello.cpp

clean:
    rm -rf *.o hello

install:
    cp -rp hello /usr/local/sbin/

root@SmartHome:~/4.12_task/4.12.2/mf#
```

and run compilation with command:

make all install clean

```
192.168.1.99 - PuTTY
    $(CC) $(CFLAGS) hello.cpp

clean:
    rm -rf *.o hello

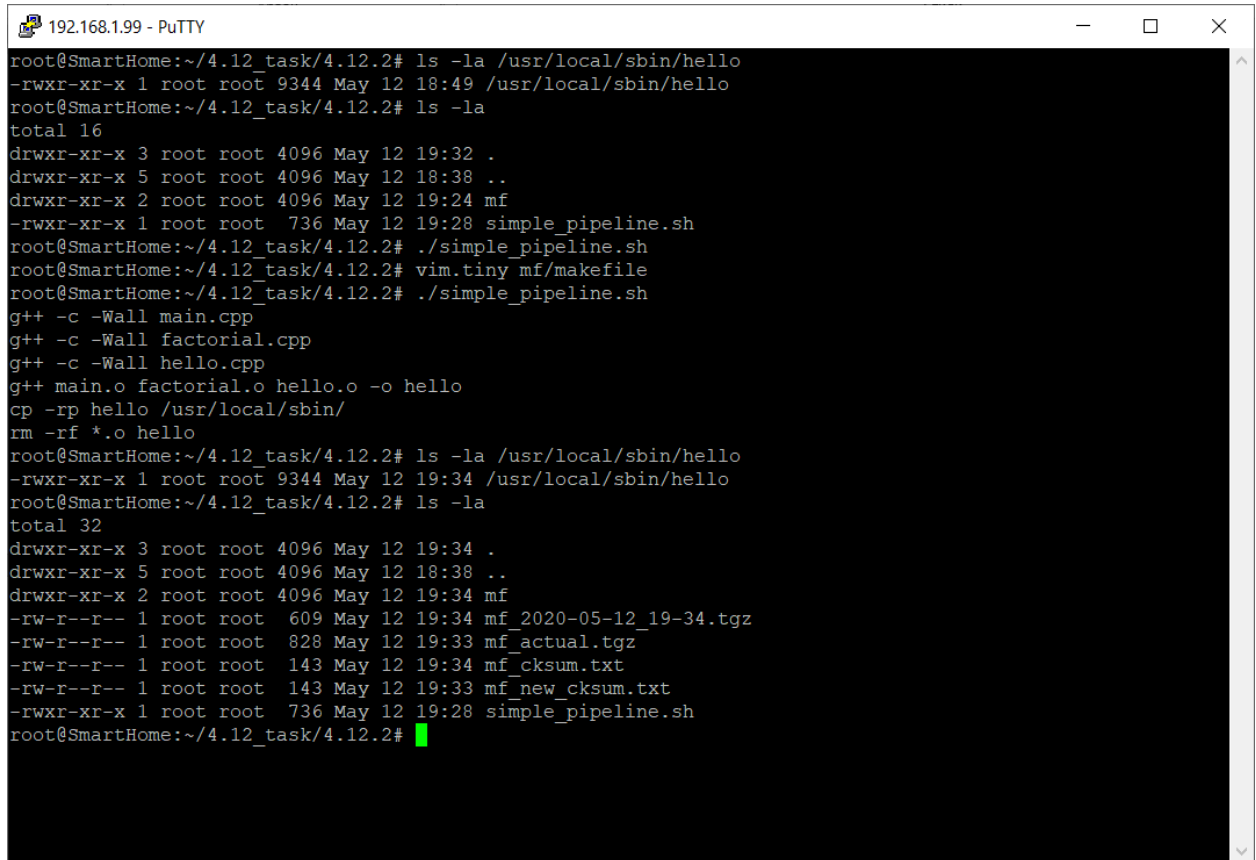
install:
    cp -rp hello /usr/local/sbin/

root@SmartHome:~/4.12_task/4.12.2/mf# ls
factorial.cpp  functions.h  hello.cpp  main.cpp  makefile  Makefile-1  Makefile-2  Makefile-3  Makefile-4
root@SmartHome:~/4.12_task/4.12.2/mf# rm M*
root@SmartHome:~/4.12_task/4.12.2/mf# ls
factorial.cpp  functions.h  hello.cpp  main.cpp  makefile
root@SmartHome:~/4.12_task/4.12.2/mf# make all install clean
g++ -c -Wall main.cpp
g++ -c -Wall factorial.cpp
g++ -c -Wall hello.cpp
g++ main.o factorial.o hello.o -o hello
cp -rp hello /usr/local/sbin/
rm -rf *.o hello
root@SmartHome:~/4.12_task/4.12.2/mf# he
head    hello    _help    helpztags  hex2hcd    hexdump
root@SmartHome:~/4.12_task/4.12.2/mf# hello
Hello World!
The factorial of 5 is 120
root@SmartHome:~/4.12_task/4.12.2/mf# whereis hello
hello: /usr/local/sbin/hello
root@SmartHome:~/4.12_task/4.12.2/mf#
```

Main result - "hello" copied to /usr/local/sbin/ dir and can be running on the System

Script name: simple_pipeline.sh (4.12.2 subfolder)

Result of script action on the screenshot:



```
192.168.1.99 - PuTTY
root@SmartHome:~/4.12_task/4.12.2# ls -la /usr/local/sbin/hello
-rwxr-xr-x 1 root root 9344 May 12 18:49 /usr/local/sbin/hello
root@SmartHome:~/4.12_task/4.12.2# ls -la
total 16
drwxr-xr-x 3 root root 4096 May 12 19:32 .
drwxr-xr-x 5 root root 4096 May 12 18:38 ..
drwxr-xr-x 2 root root 4096 May 12 19:24 mf
-rwxr-xr-x 1 root root 736 May 12 19:28 simple_pipeline.sh
root@SmartHome:~/4.12_task/4.12.2# ./simple_pipeline.sh
root@SmartHome:~/4.12_task/4.12.2# vim.tiny mf/makefile
root@SmartHome:~/4.12_task/4.12.2# ./simple_pipeline.sh
g++ -c -Wall main.cpp
g++ -c -Wall factorial.cpp
g++ -c -Wall hello.cpp
g++ main.o factorial.o hello.o -o hello
cp -rp hello /usr/local/sbin/
rm -rf *.o hello
root@SmartHome:~/4.12_task/4.12.2# ls -la /usr/local/sbin/hello
-rwxr-xr-x 1 root root 9344 May 12 19:34 /usr/local/sbin/hello
root@SmartHome:~/4.12_task/4.12.2# ls -la
total 32
drwxr-xr-x 3 root root 4096 May 12 19:34 .
drwxr-xr-x 5 root root 4096 May 12 18:38 ..
drwxr-xr-x 2 root root 4096 May 12 19:34 mf
-rw-r--r-- 1 root root 609 May 12 19:34 mf_2020-05-12_19-34.tgz
-rw-r--r-- 1 root root 828 May 12 19:33 mf_actual.tgz
-rw-r--r-- 1 root root 143 May 12 19:34 mf_cksum.txt
-rw-r--r-- 1 root root 143 May 12 19:33 mf_new_cksum.txt
-rwxr-xr-x 1 root root 736 May 12 19:28 simple_pipeline.sh
root@SmartHome:~/4.12_task/4.12.2#
```

4.12.3 Создать скрипт сбора статистики работы системы

На этапе инициализации:

Создайте задание для cron, согласно которому каждые 5 минут файл ~/memory/stat, а также логи, полученные основным скриптом, будут упаковываться в архив.

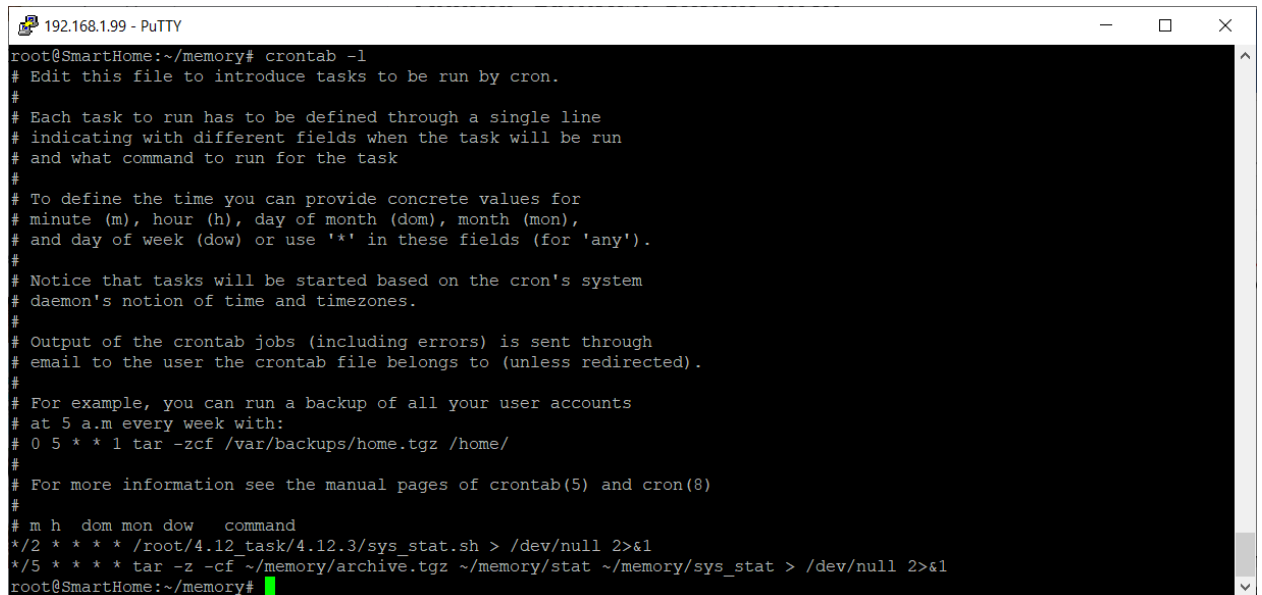
Add to Cron this command:

Tar -z -cf ~/memory/archive.tgz ~/memory/stat ~/memory/sys_stat > /dev/null 2>&1

crontab -e

result of adding to cron:

crontab -l



```
192.168.1.99 - PuTTY
root@SmartHome:~/memory# crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/2 * * * * /root/4.12_task/4.12.3/sys_stat.sh > /dev/null 2>&1
*/5 * * * * tar -z -cf ~/memory/archive.tgz ~/memory/stat ~/memory/sys_stat > /dev/null 2>&1
root@SmartHome:~/memory#
```

Создайте задание для cron, согласно которому каждые 2 минуты в файл ~/memory/stat будет добавляться информация о текущем состоянии памяти, без учета размера подкачки и заголовка.

Script name (4.12.2 sub folder): sys_stat.sh

free -h | grep -i '^Mem'

Основной этап выполнять каждые две минуты:

С помощью команды vmstat, в течении 30с с интервалом в 3с, собирайте статистику об использовании ресурсов системы. Посчитайте среднее количество переключений контекста ядра в секунду на заданном интервале времени. Информацию – в лог.

vmstat -t 3 10

Получите информацию о средней загрузенности процессора в течении последних 15m. Информацию – в лог.

uptime | awk -F ", " '{print \$5}'

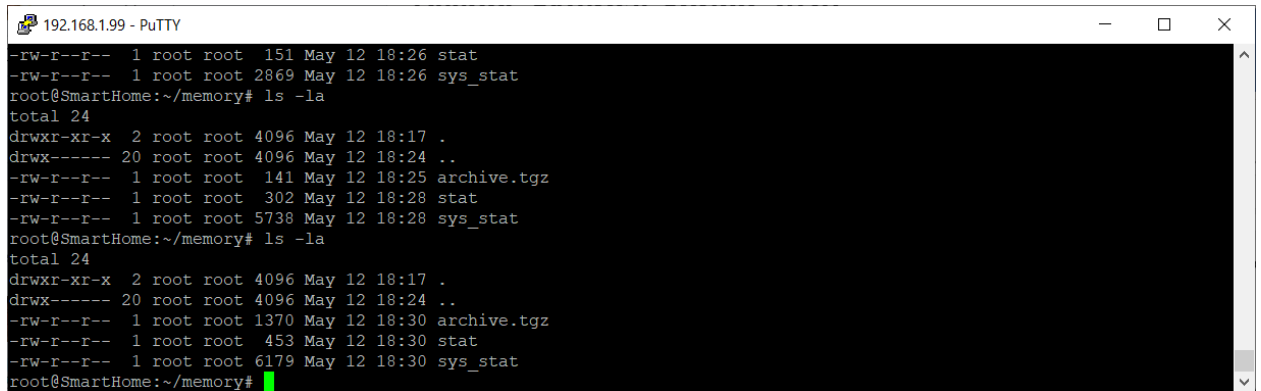
Опишите текущее состояние страниц памяти, доступных в вашей системе.
Информацию – в лог.

swapon -s

Опишите текущее состояние разделов жестких дисков, доступных в вашей системе. Информацию – в лог.

df -h

result of Cron actions (memory dir added to git 4.12.3 subfolder):



```
192.168.1.99 - PuTTY
-rw-r--r-- 1 root root 151 May 12 18:26 stat
-rw-r--r-- 1 root root 2869 May 12 18:26 sys_stat
root@SmartHome:~/memory# ls -la
total 24
drwxr-xr-x 2 root root 4096 May 12 18:17 .
drwx----- 20 root root 4096 May 12 18:24 ..
-rw-r--r-- 1 root root 141 May 12 18:25 archive.tgz
-rw-r--r-- 1 root root 302 May 12 18:28 stat
-rw-r--r-- 1 root root 5738 May 12 18:28 sys_stat
root@SmartHome:~/memory# ls -la
total 24
drwxr-xr-x 2 root root 4096 May 12 18:17 .
drwx----- 20 root root 4096 May 12 18:24 ..
-rw-r--r-- 1 root root 1370 May 12 18:30 archive.tgz
-rw-r--r-- 1 root root 453 May 12 18:30 stat
-rw-r--r-- 1 root root 6179 May 12 18:30 sys_stat
root@SmartHome:~/memory#
```