Podstawy Linux i Bash 2

Wojciech Barczyński

Spis treści

1. Podstawy basha – Przypomnienie

Zadanie na rozgrzewkę.

1. Utwórz następującą strukturę katalogów i plików:

```
magazyn
|- slodycze
|    \- produkty.txt
|       wedel,2
|       goplana,3
|
|- mleczne
     \- produkty.txt
        mlekovita,5
        zimnemleko,3
```

1. Wypisz za pomocną jednej komendy:
   5
   3
   2
   3

2. Wyszukaj plików zawierających **3**

2. Zanotuj swoimi słowami do czego służą następujące komendy:

cd	
mkdir	
mv	
cp	
touch	
grep	
cut	
cat	
paste	
pwd	
echo "${HOME}"	
sed	
tail	
less	
date	
tree	
find	
xargs	
sort	
uniq	

3. Terminal:

ctr-d	
ctr-c	
ctr-z	
fg	
ctr-r	

4. Komenda find – wyszukanie plików o określonej nazwie:

```
find /etc -iname '*passwd*'
```

Wyszukaj pliki lub foldery z poziomu katalogu głównego zawierające 'mount'

Przydatne:

```
find /etc -iname '*passwd*' 2>/dev/null
find / -iname '*etc*' -type f
```

5. Za wskazówkami wykładowcy, wyciąganie ciekawych informacji z /var/log/syslog z użyciem *grep*, *egrep*, *cut*, *wc*, np.
    - linii z UUID
    - zliczenie eventów per data
    - zliczenie event per data i godzina
    - znalezienie eventów z gnome desktop
    - które komponenty js gnome desktop najwięcej logują?

2. Podstawy Linuxa

Naszym zadaniem będzie poznać bliżej system linux.

1. Uaktualnienie pakietów

   ```
   $ sudo apt update

   # proszę nie uruchamiać
   # sudo apt upgrade
   ```

2. Używając poniższych komend dowiedzmy się więcej o naszym Linuxie:

   ```
   $ uname -a

   $ lsb_releases -a
   ```

3. Wprowadzenie i omówienie procesów na linuxie:

   ```
   $ ps
   $ ps -a
   $ ps -au
   $ ps -aux
   ```

4. Procesy 2, wyszukiwanie PID procesu:

   1. Uruchom drugie okno z terminalem albo zakładkę (shift-ctr t).

   2. [Terminal 1] Wyszukaj procesu o nazwie ipython:

      ```
      ps -a | grep ipython
      ```

   3. [Terminal 2] Uruchom ipython:

      ```
      $ ipython
      ```

   4. [Terminal 1] Teraz powinniśmy znaleźć proces:

      ```
      $ ps a | grep ipython
      ```

   5. Zamknij ipythona i wyświetl ponownie procesy.

5. Procesy 3, zamykanie niedziałających lub nieodpowiadających aplikacji

   1. Uruchom drugie okno z terminalem albo zakładkę (shift-ctr t).

   2. [Terminal 2] Uruchom ipython:
      ```
      $ ipython
      ```

   3. [Terminal 1] Teraz powinniśmy znaleźć proces:

      ```
      $ ps a | grep ipython
      ```

4. Wyobraźmy sobie, że ipython nie odpowiada albo nie mamy możliwość przełączenia się na konsolę, żeby go zamknąć.

5. [Terminal 1] Zamiast wyjść za pomocą komendy exit(), możemy wysłać sygnał SIGTERM do aplikacji:

```
$ kill <PID>
```

W przypadku, gdyby nasza aplikacja przestała odpowiadać na SIGTERM (wezwanie do zamknięcia, możemy są zabić. Uwaga: niebezpieczne.

```
$ kill -9 <PID>
```

6. Procesy 4, komendy pgrep i pkill:

```
$ pgrep python
$ pkill python
```

3. Prawa Dostępu

Naszym zadaniem będzie poznać w jaki sposób możemy dać dostęp do naszych plików i katalogów innym użytkownikom.

1. Prawa dostępu/własności 1, omówienie pola właściciel i grupa właściciela w wyniku komend:

   ```
   $ ls -la /home
   $ ls -la /home/tester
   ```

   Porównaj z:

   ```
   $ stat /home
   ```

   Razem z wykładowcą:

   ```
   $ cat /etc/group | grep tester
   # Co oznacza sudo ? ^

   $ cat /etc/passwd | grep tester
   ```

2. Prawa dostępu/własności 2, edycja plików innego użytkownika:

   ```
   $ mkdir prawa_dostepu
   $ cd prawa_dostepu
   $ touch plik_tester.txt
   $ sudo touch plik_root.txt

   # porownaj
   $ ls -la

   # sróbuj zedytować jako tester
   $ gedit plik_root.txt
   ```

   Skasuj cały katalog.

3. Przekazanie własności plików z komendą *chown* (bardzo przydatne jak skrypt uruchamiany z prawami roota zmieni właściciela plików w Twoim katalogu domowym):

   ```
   mkdir prawa_dostepu
   cd prawa_dostepu

   touch moj_plik.txt
   sudo touch plik_roota.txt

   # porownaj
   ls -la

   # spróbuj edytować
   nano plik_roota.txt

   sudo chown tester:tester plik_roota.txt

   # porownaj
   ls -la
   ```

```
# czy uda nam się plik wydedytować?
nano plik_roota.txt
```

Skasuj katalog *prawa_dostępu*.

4. Przekazanie własności folderów z komendą *chown*:

```
mkdir prawa_dostepu
cd prawa_dostepu

mkdir moj_katalog
sudo mkdir katalog_roota
sudo touch katalog_roota/plik.txt

# spróbuj
mv katalog_roota/plik.txt ${HOME}/mój_plik

# przekaż własność katalogu
chown tester:tester -R katalog_roota/
```

Zauważ:

```
grep tester /etc/passwd
```

5. Prawa dostępu, omówienie pierwszej kolumny:

```
ls -la /

ls -la ~
```

6. Plik do który wszyscy mogą zapisywać:

```
touch ~/moj_plik.txt
# wszyscy mogą zapisawywać :D
chmod a+w ~/mój_plik
```

7. Katalog, którego inni nie mogą otworzyć:

```
ls -la ~ | grep Public

# nikt nie może otworzyć katalogu
chmod -R 644 ~/Public

# a teaz wszycy
chmod -R 755 ~/Public
chmod -R a+x ~/Public
```

8. Mój prywatny plik (na przykład klucze ssh i inne sekrety)

```
touch ~/Public/mój_prywatny_plik.txt
chmod 600 ~/public/mój_prywatny_plik.txt
```

9. Podsumowanie:

- Symboliczna reprezentacja:

Kto	Działanie	Uprawnienia
u użytkownik	+ dać	r odczyt
g grupa	- zabrać	w zapis
o inni	= przypisanie	x wykonanie
a wszyscy	s suid	

- Liczbowa:

Kto	Ustawione	Nie ustawione
r	4	0
w	2	0
x	1	0

- Quiz:

Symboliczna	Liczbowa
r	
rw	
wx	
rwx	

- Przykład, popularny, aka nie wiem co robię:

```
# zło w czystej postaci…
chmod 777 moj_plik.txt
```

10. Przyda się, kiedy przez przypadek zrobisz git comit jako root:

```
$ sudo chown tester:tester -R .git/
```

4. Omówienie katalogu głównego

Miejsce na notatki:

Kto	Opis
/etc	
/var/log	
/tmp	
/bin	
/usr/bin	
/var/www/html	
/media	
/mnt	

5. Skrypty Basha

Naszym zadaniem będzie napisanie nasze pierwszy skrypt basha.

1. Utwórz skypt zapisz_kat_glowny.sh (pierwsza linia definiuje w którym shellu uruchomić zawartość w pliku):

   ```
   #!/bin/bash

   moj_plik="${USER}.txt"
   moj_katalog="tmp-dane"

   mkdir -p "${moj_katalog}"

   ls -la / > "${moj_katalog}/${moj_plik}"
   ```

2. Uruchom, trzy sposoby:

   ```
   # bezpośrednio
   bash zapisz_kat_glowny.sh

   # tu polegamy na pierwszej linii pliku
   # - bash
   chmod +x zapisz_kat_glowny.sh
   ./zapisz_kat_glowny.sh
   ```

3. Wyświetlanie przebiegu wykonania skryptu:

   ```
   bash -x zapisz_kat_glowny.sh
   ```

4. Zapis procesów do pliku:

   Napisz program jak w zadaniu 1, ale zapisujący wszystkie działające procesy do innego pliku.

5. Ważne komendy, zanotuj co one oznaczają:

   ```
   set -o nounset
   set -o errexit
   set -x
   ```

   Zauważ długie i krótkie formy, e.g., `set -e.`

6. Dlaczego *nounset*, przetestuj z *set -o nounset* i bez:

   ```
   #!/bin/bash

   set -o nounset

   folder_do_skasowania="/${USERRR}"
   ```

```
echo "TO KASUJEMY ${folder_do_skasowania}"
```

7. Dlaczego *errexit*, przetestuj z *set -o errexit* i bez:

```
#!/bin/bash

set -o errexit

echo "PRZED"
echo "TU 300 innych linii"
grep HIMALAJE9 *.txt
echo "TO 400 innych linii do debugowania"
```

8. Instrukcje warunkowe dla zmiennych:

```
if [[ $USER = "tester" ]]; then
    echo "Witaj ${USER}";
fi;
```

9. [Zaawansowane] Co jeśli chcemy komendę chcemy wykonać nawet jak zwraca błąd:

```
if ! <possible failing command> ; then
    echo "failure ignored"
fi

rm -rf somedir || exit_on_error "Failed to remove the directory"
```

   Komenda, która kończy się błędem do wykorzystania powyżej:

```
      cat NIEISTNIEJACY_PLIK.txt
```

10. Jeśli chcemy jeszcze podnieść niezawodność naszych skryptów, warto dodać również:

    set -o pipefail

11. [Zaawansowane] Jak jest różnica między instrukcjami warunkowymi [[ ]] , a [ ] .

12. [Zaawansowane] Jeśli chcesz się więcej nauczyć o skryptach bash:
    a. Napisanie pętli do wypisania zawartości katalogu
    b. Instrukcje warunkowe czy plik lub katalog istnieje
    c. Funkcje

Pamiętaj na przyszłość:

- KISS - keep it simple.
- Pliki basha mogą być w bardzo krótkim czasie nieczytelne.
- Kod review bardzo pomaga szybko zidentyfikować bashowe monstra.
- Dodaj do zakładek:
  http://robertmuth.blogspot.com/2012/08/better-bash-scripting-in-15-minutes.html
- Bash linter:
  https://github.com/koalaman/shellcheck