Podstawy pracy z systemem UNIX

Większość współczesnych komputerów (i podobnych urządzeń, np tablety czy telefony) wyposażonych jest w złożone oprogramowanie które składa się na system operacyjny. Z całą pewnością korzystałeś z systemów firmy Microsoft czyli rodziny Windows. Mogłeś też zetknąć się z Android'em (czyli odmianą UNIXa) od Google czy iOS od Apple. W większości przypadków system posiada tzw. interfejs graficzny czyli GUI. Większość z nich jest zasadniczo podobna i np. uruchomienie przeglądarki internetowej czy przeglądanie dysku nie jest dla nikogo wyzwaniem.

Jednak nie każdy komputer posiada GUI, dotyczy to głównie dużych komputerów używanych w poważnych obliczeniach numerycznych (np. wyznaczanie właściwości aerodynamicznych samochodu z użyciem Fluent'a). W takim przypadku nie ma możliwości skorzystania z klawiatury i myszki czy podejrzenia czegoś na ekranie, ponieważ komputer znajduje się w serwerowni, czasami w innym kraju. Aby korzystać z takiego zdalnego komputera musimy połączyć się z nim przy pomocy specjalnego programu, następnie wydajemy mu polecenia w trybie tekstowym.

Na potrzeby tego laboratorium każdy otrzymał kartkę z loginem i hasłem. Są one ważne do końca semestru i można przy ich pomocy zalogować się na nasz szkolny serwer również spoza kampusu.

Gdy korzystasz z Windows'ów najwygodniej do połączenia wykorzystać darmowy program PuTTy. Po uruchomieniu pokaże się okno jak na obrazku obok. Należy podać: - `Hostname`– nazwa hosta, czyli `info3.meil.pw.edu.pl` - `Port`– Numer portu z których chcemy się połączyć, czyli 22, oraz protokół `SSH` z listy poniżej

Następnie pojawi się czarne okno z zapytaniem o login i hasło. Po podaniu i poprawny zalogowaniu zobaczysz informacje o dacie, licencji, wersji systemu itp. kończące się:

```
Last login: Thu Feb 21 06:23:38 2013 from xx.xx.xx.xx
stud-00@info3:~$
```

Zapis:

```
stud-00@info3:~$
```

oznacza, że jako użytkownik `stud-00` jesteśmy zalogowani na komputer `info3`. Między znakami : i $ znajduje się aktualny katalog. W tym przypadku ~, czyli katalog domowy, inaczej `/home/students/stud-00`

Ćwiczenia

Pierwsze starcie

Wpisz do konsoli `date` i wciśnij enter. Komputer wyświetli aktualną (jego zdaniem) datę. następnie ponownie wyświetli linijką kończącą się na $, oznaczającą że czeka na polecenia. Używając strzałki do góry możesz przeglądać historię poleceń. Jeśli wpiszesz `dat` i naciśniesz 2x klawisz tab, wyświetlona Ci zostanie lista poleceń zaczynających się na `dat` lub jeśli jest tylko jedno, nazwa zostanie dokończona. Pamiętaj o tych 2 trikach, bo znacznie ułatwiają pracę w trybie tekstowym.

Poruszanie się po katalogach

Pracując w trybie tekstowym, zawsze pracujemy w jakimś katalogu, tzw. katalogu bieżącym. Jeśli uruchomimy jakiś program, np. proste programy czytające z pliku z Informatyki I, będą one czytały z plików w tym katalogu. Każdemu programowi którego będziesz używać a który potrzebuje nazwy pliku lub katalogu (np. do kopiowania) może ją przyjąć w kilku postaciach. Po pierwsze ścieżka bezwzględna, zaczynająca się od znaku / np:

```
/home/students/stud-00
/usr/bin/bash
/etc
```

Sprawdź, w jakim katalogu się znajdujesz, wpisz `pwd` i wciśnij enter.

Aby zmienić katalog, wykorzystuje się polecenie `cd`, np.

```
stud-00@info3:~$ cd /tmp
stud-00@info3:/tmp$ pwd
/tmp
```

Teraz przejdź do katalogu `/home` i sprawdź czy się udało, z użyciem polecenia `pwd`

Dodatkowo, oprócz ścieżki bezwzględnej, można podać nazwę katalogu który nas interesuje na kilka innych sposobów.

- ~ zawsze oznacza katalog domowy

- `../` oznacza katalog nadrzędny
- `/` oznacza katalog główny, początek każdej ścieżki bezwzględnej
- `.` i `./` oznacza katalog bieżący, ten zwracany przez `pwd`

Przejdź teraz z powrotem do katalogu domowego i sprawdź czy się udało. Następnie 2 razy przejdź katalog wyżej i sprawdź, czy katalog bieżący to `/home`

Tworzenie i usuwanie katalogów

Do tworzenia katalogów służy polecenie `mkdir` np.

```
$ mkdir nazwa_katalogu
```

a do sprawdzenia zawartości aktualnego katalogu polecenie `ls`. Stwórz teraz katalogi A,B,C i D, każdy wewnątrz poprzedniego. Będziesz musiał stworzyć katalog A, przejść do niego, stworzyć B itd. Do usuwanie katalogów służy polecenie `rmdir`. Usuń teraz stworzone katalogi.

UWAGA: nie można w ten sposób usunąć katalogu posiadającego zawartość

Podstawowe operacja na plikach i katalogach

Komenda `echo` wypisuje na ekran ciąg znaków który podaje się jej jako argument. Sprawdź. Aby stworzyć pierwszy plik wpisz (o znaczeniu `>>` będzie na kolejnych zajęciach)

```
$ echo pierwszy plik >> plik.txt
```

Aby wyświetlić zawartość pliku a ekranie używamy `cat`

```
$ cat plik.txt
```

Kopiowanie i przenoszenie

Do kopiowanie służy komenda `cp CO GDZIE`. Stwórz teraz katalog i skopiuj do niego twój plik. Powinno to wyglądać tak:

```
$ cp plik.txt katalog
```

Aby przenieść/zmienić nazwę pliku lub katalogu używamy `mv CO GDZIE`. Przejdź do nowego katalogu i zmień nazwę pliku. Następnie usuń plik poleceniem `rm`

Pomoc

Znakomita większość komend trybu tekstowego posiada porządną dokumentację dostępną od ręki.

```
$ man rm
$ rm --help
```

W przypadku komendy `man` dostajemy kompetentniejszą dokumentację. Przewija się strzałkami, aby zakończyć wciśnij Q. Sprawdź instrukcje dla poleceń `who`, `whoami`, `finger` i `date`. Sprawdź jak działają.

Program Tar

Program `tar` służy do pakowania i rozpakowywania drzewa katalogów i plików w jeden plik. Niekoniecznie musi on być mniejszy niż oryginalne pliki. Dopiero użycie kompresji zmniejszy objętość. Najpierw przygotuj kilka plików do spakowania:

```
$ mkdir a
$ cd a
$ mkdir b
$ echo asdasd >> ./b/c
$ cat ./b/c
asdasd
```

Teraz spakuj a nastnie podejrzyj archiwum programem `mc`

```
$ tar -cf test.tar b
$ ls
b  test.tar
$ mc
```

Sprawdź zawartość katalogu, usuń to co przed chwilą spakowałeś do archiwum, następnie rozpakuj.

```
$ ls
b  test.tar.gz
$ rm -rf b
$ ls
$ tar -xf test.tar
$ ls
b  test.tar
```

Sprawdź poleceniem `ls -la` objętość archiwum, zapisz, następnie spakuj te same pliki z dodatkową flagą `z` zmieniając rozszerzenie na `tar.gz`. Sprawdź czy plik wynikowy jest mniejszy.

Proste skrypty

Najważniejszym aspektem pracy w trybie tekstowym jest możliwość tworzenie skryptów, czyli zapisanych w pliku kolejnych komend wykonywanych tak, jakbyśmy wpisywali je z klawiatury. Więcej o zaawansowanych skryptach dowiesz się na następnych laboratoriach, pierwszy napiszesz dzisiaj.

Prostym i dość wygodnym edytorem tekstu jest `nano` lub `vim`. Uruchom go komendą `nano NAZWAPLIKU` i zapisz do niego pierwszy skrypt:

```
#!/bin/bash
echo 1
echo 2
```

Natępnie trzeba zmienić uprawnienia, pozwolić na uruchomienie naszego skryptu:

```
$ chmod +x skrypt.sh
$ ./skrypt.sh
```

Zmienne

Bash obsługuje zmienne, jak w C. Aby stworzyć zmienną:

```
zmienna=$( JAKISPROGRAM )
zmienna=$( echo 1 )
```

Aby odczytać zmienną:

```
echo $zmienna
```

Pierwszy skrypt

Przygotuj strukturę katalogów:

- AA
- BB
  - plik.txt
- CC
  - DD
  - plik.txt

zawierając komendy w skrypcie, plik ma zawierać datę z użyciem `date`.

Zmodyfikuj skrypt tak, aby nazwa każdego katalogu zaczynała się od wielkości ze zmiennej `\$1`, która jest pierwszym argumentem skryptu w linii komend.

- $1_AA
- $1_BB
  - ...

Pętle

Przygotuj skrypt:

```
for i in *.txt
do
 cp $i $1_$i
done
```

i uruchom go w katalogu z plikiem `.txt`, jak działa? Pamiętaj o argumencie skryptu!

Napisz skrypt który tworzy katalog `\$1` i kopiuje do niego wszystkie pliki .txt dodając przedrostek `\$2`. Co się stanie jak nie podasz argumentów do skryptu?

GUI

Zresetuj komputer, uruchom Ubuntu i zaloguj się na konto quest. Używając menu z lewej strony uruchom terminal.

Połącz się programem `ssh LOGIN@HOST` z serwerem info3. Utwórz plik `copyme`.

Zakończ połączenie `Ctrl+D`. Użyj programu

```
scp LOGIN@HOST:SCIEZKA_DO_PLIKU GDZIE_KOPIOWAĆ
```

aby ściągnąć `copyme` na dysk lokalny. Program `scp` działa tak jak `cp`, z tą różnicą, że cel lub źródło znajduje się na innym komputerze obsługującym połączenia ssh.

Coś na deser *

Pre-rekwizyty

Sprawdź do czego służy program `write` z użyciem `man`. Użyj go. następnie porównaj "wyjście" komend:

```
$ who
$ who | awk '{print $1}'
$ who | awk '{print $2}'
```

Skrypt spamera

Stwórz skrypt:

```
#!/bin/bash
for u in $( who | awk '{print $1}' )
do
   echo $u
done
```

następnie zmodyfikuj go tak, aby "zaspamować" wszystkich zalogowanych.

Kolejnym krokiem będzie dodanie do skryptu pakowania uzyskanego drzewka, następnie usuwanie oryginału.