SOWID – WYBRANE KOMENDY

PAWEŁ JÓZIAK
WYDZIAŁ MATEMATYKI I NAUK INFORMACYJNYCH
POLITECHNIKI WARSZAWSKIEJ

- Nigdy nie wykonuj poleceń metodą *copy&paste*. Przepisz je ręcznie!
- Nie redukuj pracy nad zadaniem do przeczytania go. Wykonaj je i upewnij się, że rozumiesz jak i dlaczego wynik jest taki, jak na ekranie. Zajrzyj do prezentacji bądź zapytaj prowadzącego, jeśli coś nie będzie jasne.
- Linia zaczynająca się od $ reprezentuje prompt zwykłego użytkownika (nie przepisuj go!). Komendą jest to, co następuje po nim.
- Ostrożnie ze spacjami: nie dodawaj ich więcej ani nie usuwaj takowych, jeśli są!

1. Podręcznik

(a) Wyświetl informację o podręczniku:
`$ man man`
Czy strony podręcznika są pogrupowane w sekcje (por. slajd #5) – zob. OPIS.
Czy format strony podręcznika odpowiada wzorcowi (por. slajd #5)?

(b) Sprawdź strony podręcznika komend, które poznawaliśmy mimochodem na wcześniejszych zajęciach:
`$ man ls`
`$ man sort`
`$ man touch`
`$ man rm`
`$ man cd`
`$ man pwd`
Tak, wszystkie te strony odwzorowują zadany format strony podręcznika.

(c) Czy wiesz, że komenda `'echo'` jest zaimplementowana dwukrotnie? Strona podręcznika
`$ man echo`
zawiera informacje o programie. Funkcje wbudowane powłoki mają pierwszeństwo, użyj
`$ man bash`
i znajdź (skorzystaj z przeszukiwania!) informacje o wbudowanej funkcji `'echo'`.

(d) Wyświetl stronę podręcznika o pliku `/etc/passwd`:
`$ man 5 passwd`
(pamiętaj: należy podać numer sekcji, oraz ominąć przedrostek `'/etc/'` związany z lokalizacją!). Co stanie się, jeśli nie wskażemy sekcji podręcznika?

(e) Nie pamiętasz syntaksu funkcji C `'printf'`? To nie problem, w podręczniku są strony dotyczące wszystkich standardowych funkcji C:
`$ man 3 printf`
Pamiętaj o podaniu sekcji, oraz niedodawaniu nawiasów wywołania!

(f) Jeśli to tylko możliwe, korzystaj z POSIXowych stron podręcznika (1P, 3P). Jeśli nie masz pewności – wywołaj wszystkie strony podręcznika:
`$ man -a kill`

2. Narzędzia sesji

(a) Czy potrzebujesz FLOSS-klienta SSH pod Windows? Odwiedź: `https://putty.org`. Na szczęście, pod Linuxem i Unixem możemy korzystać z natywnego klienta.

(b) Uruchom zdalny terminal, łącząc się z serwerem `'ssh'`:
`$ ssh twoj_uzytkownik@ssh.mini.pw.edu.pl`
Wybierz *'yes'* na zapytanie (tylko przy pierwszym połączeniu: akceptacja certyfikatu serwera zostanie zapamiętana). Dzięki temu masz teraz dostęp do lokalnego interfejsu, lecz komendy są wykonywane na zdalnej maszynie.
Pamiętasz, że domena – czyli `'.mini.pw.edu.pl'`, jeśli jest taka sama, może być ominięta? Podobnie z użytkownikiem. Zatem łącząc się z laboratorium, wystarczy
`$ ssh ssh`
(pierwsze `'ssh'` to nazwa komendy, drugie – to nazwa maszyny w lokalnej domenie). Tak, to może być mylące.
Serwer dostępny pod adresem `'ssh.mini.pw.edu.pl'` jest bramką do infrastruktury wydziałowej, nie korzystaj z niego do poważniejszych ćwiczeń (jego zasoby nie wystarczą do obsługi wielu osób ćwiczących równocześnie) – będąc już połączonym do niego, możesz przenieść się do wewnętrznej maszyny do ćwiczeń:
`$ ssh unix`
Aby zakończyć pracę `'ssh'`, wpisz w terminalu
`$ exit`

(c) Otwórz terminal. Wywołaj w nim komendę
`$ top`
(to rodzaj menedżera zadań; możemy obserwować w czasie rzeczywistym zużycie zasobów). Jeśli zamkniesz okno terminala, proces zostanie zamknięty.

(d) Otwórz terminal. Wystartuj chronioną sesję
`$ screen`
i wywołaj w nim:
`$ top`
Możesz teraz odłączyć się z sesji albo zamknąć terminal. Otwórz osobne okno terminala i podłącz się do sesji:
`$ screen -r`
Jak widać, ten proces wciąż działa. Ten sposób funkcjonowania jest tożsamy dla sesji lokalnych, jak i zdalnych (via `ssh`).
Aby zamknąć `top`, naciśnij <Q>, a potem zamknij sesję chronioną:
`$ exit`

3. Informacje o użytkowniku i systemie

(a) Która teraz godzina? Wywołaj
`$ date`
Zajrzyj na strony podręcznika dla wyświetlania daty/czasu w innym formacie (niestety, nie ma tu jednolitej standaryzacji).

(b) Jakie są daty następnych kilku zajęć? Spróbuj
`$ cal -3`

(c) Sprawdź, czy Twój komputer ma system Linux:
`$ uname`
Owszem, tak jest. Zdobądź wszystkie dostępne informacje:
`$ uname -a`
Zajrzyj na strony podręcznika dla klaryfikacji dostępnych opcji.

(d) W systemie Linux możesz sprawdzić informacje o sprzęcie za pomocą komend:
`$ lscpu`
`$ lspci`
`$ lsusb`
Możesz użyć opcji `-v` dla dokładniejszej odpowiedzi ostatnich dwóch komend (czy staje się to bardziej zrozumiałe?)

(e) Sprawdź zużycie pamięci w sposób zrozumiały dla człowieka:
`$ free -h`

Sprawdź zajętość przestrzeni dyskowych:
`$ df -h`

(f) Wyświetl informacje o swoim użytkowniku i grupach:
`$ id`
bądź ogranicz się do samej informacji o użytkowniku:
`$ id -un`
bądź też ogranicz się do samej informacji o grupach:
`$ id -Gn`

(g) Połącz się do serwera `'ssh'`:
`$ ssh ssh`
i sprawdź kto jest zalogowany:
`$ who`
`$ w`

4. Otwieranie i edycja plików

Przeglądarki plików mogą być jako filtry strumieni, więc będziemy więcej ćwiczyć przy okazji kolejnego laboratorium dedykowanego właśnie filtrom strumieni. Poniżej tylko wstęp do tej rodziny.

(a) `'cat'` może być używany do wyświetlania bardzo krótkich plików. Podczas poprzedniego laboratorium sprawdzaliśmy Powłoki dostępne w systemie. Istnieje plik zawierający informacje o dostępnych Powłokach. Wyświetlmy go:
`$ cat /etc/shells`

(b) `'echo'` pozwala wyświetlać to, jak Powłoka *widzi* to, co do niej wpisujemy. Warto go użyć, jesli nie jesteśmy pewni, na przykład:
`$ echo "\"\\'\$"`
Można używać go także do wyświetlania treści zmiennych:
`$  echo $PATH`

(c) `'more'` jest prostą przeglądarką plików, która pozwala na stronicowanie. Warto używać do plików, które mogą mieć kilkadziesiąc linii:
`$ more /etc/passwd`

(d) `'less'` pozwala na faktyczną nawigację po pliku, a nie tylko stronicowanie – warto poćwiczyć przeszukiwanie w przód i wstecz. Weźmy do tego celu jakiś długi plik (sam plik użyty poniżej nie jest istotny, ważne jedynie, że jest długi):
`$ less /etc/services`
Zwróć uwagę, że znaki niealfanumeryczne mogą wymagać wy-escape'owania. Omówimy to dokładniej na kolejnych zajęciach.

(e) Większość systemów Uniksowych używa `'less'` dla wyświetlania stron podręcznika. Zamiast ćwiczenia bezpośrednio `'less'`, można praktykować komendę `'man'`.

(f) Kilka prostych przykładów na `'head'` i `'tail'`:
`$ head /etc/services`
`$ head -n5 /etc/services`
`$ tail /etc/services`

(g) Poćwicz edytor `'nano'`. Jako edytor należący do ekosystemu GNU, jest domyślnym edytorem w wielu systemach GNU/Linux.

5. Zarządzanie plikami

(a) Przechodzenie między lokalizacjami:
`$ cd /etc; pwd`
Powtórz następującą komendę kilkukrotnie:
`$ cd -; pwd`
W końcu, wróć do katalogu domowego:
`$ cd`

(b) Zamiast wywoływania komendy `'pwd'` można sprawdzić zawartość zmiennej `'PWD'`:
`$ pwd`

```
$ echo $PWD
```
Zmiana katalogu skutkuje powstaniem nowej zmiennej:
```
$ echo $OLDPWD
```

(c) Ile plików tworzymy?
```
$ touch a b c
$ touch "d e f"
```
Podobne zjawisko (zmienna liczba plików będących argumentami komendy) ma zastosowanie dla wielu komend. Usuń te cztery pliki:
```
$ rm a b c "d e f"
```
Zwróć uwagę, że `'rm'` usuwa pliki permanentnie, nie przesuwa ich do "śmietnika" na pulpicie!

(d) Dla `'cp'` oraz `'mv'` drugi argument może być nazwą bądź ścieżką. Poniższe dwie komendy będą miały ten sam efekt:
```
$ cp /etc/passwd passwd
$ cp /etc/passwd .
```
Zwróć uwagę, że `'cp'` nie ma domyślnej postaci drugiego argumentu, więc poniższa komenda nie zadziała:
```
$ cp /etc/passwd
```

(e) *Zmiana nazwy* pliku to nic innego, jak przesunięcie go do nowej nazwy:
```
$ touch file5; mv file5 file5e
```

(f) Czy symlinki mogą wskazywać na nieistniejący plik? Tak, wówczas mówi się o *zerwanych symlinkach*. W takiej sytuacji komunikaty Powłoki mogą nie wskazywać faktycznego źródła problemu. Sprawdź:
```
$ nano file5f
```
(dodaj jakąś treść)
```
$ ln -s file5f symlink5f; ls -l
$ cat symlink5f
$ nano symlink5f
```
(zmień treść)
```
$ cat file5f
$ rm file5f; ls -l
$ cat symlink5f
```
Na koniec, sprzątanie:
```
$ rm symlink5f
```

(g) Pamiętaj: odpalaj najpierw `'rsync'` w trybie testowania (*dry-run*). Dlaczego? Nadmiarowy ukośnik na końcu (na przykład dodawany przez autouzupełnienie ścieżki) może reprezentować inne katalogi używane do synchronizacji (dla opisu opcji, zob. slajd #28).
```
$ mkdir A B; touch A/a A/b A/c
$ rsync -avzn --delete A/ B
$ rsync -avzn --delete A/ B/
$ rsync -avzn --delete A B
$ rsync -avzn --delete A B/
```
Różnica jest podobna do różnicy między
```
$ cp -R A/* B/
```
a
```
$ cp -R A B/
```
Co więcej, pamiętaj że `'rsync'` oraz `'cp'` mają inne role. Przykład z `'cp'` był użyty tylko by pokazać różnice w zachowaniu `'rsync'` przy innym formacie źródłowego katalogu.
Porządki: usuń oba katalogi:
```
$ rm -r A B
```

(h) Dla wygody i bezpieczeństwa, skopiuj plik `'/etc/passwd'` do swojego katalogu domowego (por. ćwiczenie 5d). Edytuj kopię lokalną: usuń jakieś linie, dodan nowe, zmodyfikuj kilka istniejących. Warto powiększyć swój terminal do całego ekranu (spróbuj <F11>) i porównaj oba pliki:

```
$ diff -y passwd /etc/passwd
```
A w formacie skróconym:
```
$ diff -y --suppress-common-lines passwd /etc/passwd
```
Format mniej wygodny dla człowieka:
```
$ diff passwd /etc/passwd
```
może być zapisany jako plik różnicowy – jakie zmiany pierwszego pliku pozwolą go przekształcić w drugi plik (dokładne wyjaśnienie operatora > pojawi się na kolejnych zajęciach):
```
$ diff passwd /etc/passwd > file.patch
```
Pliki różnicowe mogą być wówczas zaaplikowane:
```
$ patch passwd file.patch; diff passwd /etc/passwd
```
bądź wycofane:
```
$ patch -R passwd file.patch; diff passwd /etc/passwd
```
Pamiętaj: aktualizacje programów FLOSS są często rozprowadzane jako pliki różnicowe!
Porządki: usuń pliki.
```
$ rm passwd file.patch
```

6. Archiwa

(a) Co się stanie, gdy próbujemy skompresować kilka plików jednocześnie?
```
$ touch a b c
$ gzip a b c; ls
```
Zdekompresujmy je:
```
$ gunzip ?.gz; ls
```
(pamiętasz ten znak jokera, prawda?)

(b) Skompresuj pliki o nazwach jednoliterowych do pojedynczego archiwum jak mistrz Uniksa:
```
$ tar -cvzf archiwum.tar.gz ?
```
Pamiętaj: opcja `'-f'` jest opcją klucz-wartość, więc musi być ostatnia w grupie, oraz zaraz po niej musi następować wartość (nazwa docelowego archiwum).
Możesz teraz bezpiecznie usunąć pliki dodane do archiwum:
```
$ ls
$ rm ?; ls
```
A teraz możemy je wypakować:
```
$ tar -xvzf archiwum.tar.gz
```
a teraz możesz usunąć zbędne archiwum:
```
$ rm archiwum.tar.gz
```

(c) Powtórz to ćwiczenie używając algorytmu kompresji *bzip2* zamiast *gzip* (por. slajd #32 i slajd #34).

(d) Powtórz to ćwiczenie używając algorytmu kompresji *xz* zamiast *gzip* (por. slajd #32 i slajd #34).

(e) Posprzataj pliki po wykonaniu tego ćwiczenia:
```
$ ls
$ rm ?; ls
```

7. Szukanie

(a) Znajdź w `'/etc'` pliki, których nazwy kończą się na `'.conf'`, zmodyfikowane dawniej niż 60 dni temu:
```
$ find /etc -name "*.conf" -mtime +60 2>/dev/null
```
Użyć cudzysłowów okalających jokera . Są one przeznaczone do przeczytania przez `'find'`, nie dla Powłoki. Sufiks `'2>/dev/null'` eliminuje wyświetlanie błędów `Permission denied` (a w zasadzie: wszystkie komunikaty błędów). Omówimy tę konstrukcję na następnym laboratorium.

(b) Zaktualizuj czas ostatniej modifikacji plików ostatnio modyfikowanych tydzień temu:
```
$ find ~ -mtime 7 -exec touch {} \;
```

(c) Utwórz katalog w swoim katalogu domowym:
```
$ mkdir nowy
```
Znajdź wszystkie zwykłe pliki w katalogu `'/etc'` o wiekości większej niż 100kB i modyfikowanych dawniej niż 60 dni temu, i skopiuj je do swojego katalogu domowego:
```
$ find /etc/ -size +100k -type f -mtime -60 -exec cp {} nowy \; 2>/dev/null
$ ls nowy
```
Czas na porządki:
```
$ rm -rf nowy
```