Notatki 1

Wstęp, podstawy Linuxa

Notatki, które będę umieszczał, nie aspirują do bycia kompletnym podręcznikiem. Stanowią zbiór haseł poruszanych na wykładzie, co do których znajomości i rozumienia warto się upewnić. Zachęcam do szerszego doczytywania we własnym zakresie, a przede wszystkim praktycznego eksperymentowania!

Zakres materiału:

1. Linux
2. Bash (wraz z typowymi programami powłoki)
3. Gnuplot
4. LaTeX
5. Mathematica
6. Python?
7. ...

Zasady zaliczenia:

- Egzamin końcowy
- Obecność na wykładzie nie jest obowiązkowa (ale oczywiście zachęcam do uczestnictwa)
- Jeśli to możliwe, polecam przynoszenie własnych laptopów – wtedy można testować wszystko na bieżąco
- Będę wrzucał notatki na stronę (`https://www.fuw.edu.pl/~pczachorowski/#teaching`)

Dystrybucja Linuxa – system zbudowany na bazie jądra Linuxa oraz menadżera pakietów (czyli programu pozwalającego na instalację i zarządzanie programami, np. apt, dnf, pacman...) współpracującego z repozytorium oprogramowania. Istnieją setki dystrybucji: Debian, Ubuntu, Mint, Fedora, Arch, Gentoo, Kali itp. (`https://distrowatch.com/`), które różnią się między sobą przede wszystkim wyborem konkretnego menadżera pakietów i oprogramowania w repozytorium, a także pewnymi rozwiązaniami konfiguracji. W chwili obecnej na komputerach w pracowniach FUW zainstalowana jest Fedora (a na niektórych innych dostępnych Państwu komputerach także Ubuntu).

Przydatne polecenia *(nawiasy ostrokątne `<,>` oznaczają argumenty polecenia i nie należy ich wpisywać przy wywoływaniu)*:

- `man/info <nazwa_polecenia>` – dokumentacja
- `ls <katalog>` – wypisuje zawartość katalogu (bez podania opcji pokazuje bieżący, opcja `-l` prezentuje wyniki w postaci listy, `-h` dodatkowo wypisuje wielkości plików w bardziej przejrzystym dla człowieka formacie)
- `pwd` – pokazuje gdzie się znajdujemy
- `cd <katalog>` – przechodzi do innego katalogu (w przypadku braku podania celu przenosi nas do katalogu domowego, bieżący katalog oznaczamy jako `.` i katalog wyżej jako `..`)
- `mkdir <katalog>` – tworzy nowy katalog
- `touch <plik>` – tworzy nowy (pusty) plik
- `rm <plik>` – usuwa plik (opcja `-r` pozwala też usunąć katalog)
- `cp <plik_a> <plik_b>` – kopiuje plik a do pliku b (opcja `-r` pozwala skopiować katalog)
- `mv <plik_a> <plik_b>` – przenosi plik a do pliku b (również działa z katalogami)
- `top/htop` – pokazuje aktywne procesy
- `kill <numer>/pkill <nazwa>` – zakończ proces (na podstawie jego numeru lub nazwy, jeżeli nie chce się zakończyć możemy „zabić” dodając opcję `-9`)
- `vim/emacs/nano <plik>` – przykładowe edytory tekstu

- `cat <plik>` – wypisuje zawartość pliku
- `echo <coś>` – wypisuje coś na ekran
- `sudo/su` – uruchom jako inny użytkownik (bez podania dodatkowych opcji pozwala uzyskać uprawnienia superużytkownika root)
- `passwd` – zmień hasło
- `chown` – zmień właściciela pliku/katalogu
- `chmod` – zmień uprawnienia dostępu do pliku/katalogu
- `ssh <użytkownik@zdalny.komputer>` – zdalna powłoka, pozwala zalogować się na innym komputerze
- `scp <plik_a> <plik_b>` – umożliwia kopiowanie plików między różnymi komputerami (plik na zdalnym komputerze podajemy w postaci `użytkownik@zdalny.komputer:plik`, opcja `-r` pozwala skopiować katalog)
- `clear` – wyczyść terminal
- `history` – historia poleceń (opcja `-c` czyści historię)
- `exit` – wyjście z powłoki
- `Ctrl+c` – przerwij bieżący proces (w razie niepowodzenia można też próbować bardziej radykalnego `Ctrl+\`)

Struktura katalogów Linuxa

- `/` – katalog główny (czasem nazywany „root", nie mylić z nazwą superużytkownika ani z jego katalogiem domowym)
- `/bin` – pliki wykonywalne
- `/boot` – pliki programu rozruchowego (w tym jądro Linuxa)
- `/dev` – urządzenia, Linux przedstawia je jako pliki, można z nich czytać i do nich zapisywać
- `/etc` – najważniejsze pliki konfiguracyjne
- `/home` – katalogi domowe użytkowników
- `/lib` – biblioteki, czasami występuje podział na 32- i 64-bitowe (np. dodatkowo jest katalog `/lib64`)
- `/mnt` – miejsce na dodatkowe punkty montowania dysków
- `/media` – punkty montowania nośników wymienialnych (płyty, pendrive)
- `/opt` – miejsce na opcjonalne programy, tu trafiają niektóre aplikacje instalowane spoza oficjalnego repozytorium
- `/proc` – informacje o aktualnie działających procesach oraz systemie (w tym np. plik `/proc/cpuinfo` z informacjami o modelu procesora)
- `/root` – katalog domowy superużytkownika root
- `/run` – tymczasowe pliki obecnie działających procesów systemowych
- `/sbin` – pliki wykonywalne do administracji systemem (przeznaczone dla superużytkownika root)
- `/srv` – pliki udostępniane (serwowane) przez system (np. strony WWW)
- `/sys` – informacje o jądrze i urządzeniach, pozwala na zarządzanie pewnymi ich ustawieniami
- `/tmp` – pliki tymczasowe
- `/usr` – tzw. „drugorzędowa struktura plików": niektóre foldery (np. `/bin` czy `/lib`) są tu powtórzone i zawierają pliki, które nie są kluczowe dla rozruchu systemu, a w niektórych nowych systemach wręcz to *te same pliki* – tzn. `/bin` jest dowiązaniem symbolicznym („skrótem") do `/usr/bin`
- `/var` – pliki często ulegające zmianom (np. logi systemowe) Warto tu wspomnieć, że do bieżącego katalogu odwołujemy się przez `.`, do katalogu w którym znajduje się katalog bieżący – poprzez `..` Np. polecenie `cd ..` pozwoli nam na wydostanie się z bieżącego katalogu.

Uprawnienia dostępu do plików – jeżeli wywołamy polecenie `ls -l` by wypisać zawartość jakiegoś katalogu, łatwo zauważymy charakterystyczną kolumnę pełną liter i myślników, np.

```
-rw-r--r-- 1 pczachorowski pczachorowski 105239 Apr 26 12:01 notatki1.pdf
```

Przede wszystkim interesuje nas tu pierwsze pole, a także trzecie i czwarte. W systemie Linux istnieją trzy kategorie uprawnień dostępu do pliku – odczyt, zapis i wykonanie[1], oznaczane odpowiednio `r`, `w` i `x`. Odczyt i zapis raczej są dość oczywiste, wykonanie dotyczy tego, czy plik można uruchomić (jak plik `.exe` lub skrypt). W przypadku katalogów odczyt pozwala na wypisanie zawartości, zapis – tworzenie i usuwanie plików, a wykonanie – czy do katalogu można wejść i uzyskać bardziej szczegółowe informacje o plikach. Uwaga – nawet bez uprawnienia do odczytu można otworzyć jakiś plik w tym katalogu jeżeli znamy jego dokładną nazwę a mamy prawa do odczytu dla pliku i wykonania dla katalogu.

Uprawnienia te można przydzielić osobno: właścicielowi pliku (który jest też wypisany w trzecim polu), grupie użytkowników (wypisana w polu czwartym, często będzie po prostu to jednoosobowa grupa zawierająca tylko właściciela pliku i o takiej samej nazwie, ale w ogólności daje to pewną swobodę w manipulowaniu uprawnieniami) oraz wszystkim pozostałym użytkownikom. To też jest wypisane przez polecenie `ls -l`, właśnie w tej kolejności: `rwx` właściciela, `rwx` grupy i `rwx` innych. Jeśli jakiegoś uprawnienia nie przyznano, w odpowiednim polu jest minus. Plik `notatki1.pdf` z przykładu może być czytany przez właściciela, grupę i kogokolwiek innego, modyfikowany (zapis) tylko przez właściciela, a uprawnień do wykonania nie ma nikt (dla pdfa są tu raczej niepotrzebne). Dodatkowe pole na samym początku (tutaj minus) oznacza typ: minus dla pliku, `d` dla katalogu (ang. *directory*) i `l` dla dowiązania (ang. *link*).

Jakkolwiek nie interesuje nas to w ramach tego wykładu, dla kompletności wyjaśnijmy też pozostałe kolumny. Druga oznacza liczbę „twardych dowiązań" (*ang. hard links*) do danego pliku – te same dane fizycznie zapisane na dysku mogą być w systemie plików reprezentowane jako więcej niż jeden plik. Dla typowych plików zwykle będzie tu jedynka, dla katalogów – liczba podkatalogów plus dwa[2]. Piąta zawiera rozmiar pliku w bajtach (możemy użyć `ls -lh` by dostać wygodniejsze jednostki), szósta, siódma i ósma – datę ostatniej modyfikacji pliku, ostatnia – jego nazwę.

Uprawnienia modyfikujemy poleceniem `chmod`: np. `chmod u+x,g-r,o+wx notatki1.pdf` doda uprawnienia wykonywania właścicielowi pliku (`u`), zabierze uprawnienia odczytu grupie (`g`) i doda uprawnienia zapisu oraz wykonywania pozostałym użytkownikom (`o`). Poleceniem `chown` z kolei możemy zmienić właściciela oraz grupę danego pliku: np. `chown root:www-data notatki1.pdf` zmieni właściciela na użytkownika (`root`), a grupę na (`www-data`). Teraz to ich będą dotyczyć odpowiednie uprawnienia. Polecenie `chown` wymaga podwyższonych uprawnień administracyjnych (o których dalej).

W domyślnych instalacjach systemu często zakłada się zaufanie między użytkownikami i ich katalogi domowe mogą być przeglądane przez innych użytkowników niż właściciel. Między innymi dlatego też na komputerach pracownianych takich jak `tempac` czy `pracownia` nie poleca się przechowywać danych wrażliwych. Można temu łatwo zaradzić, zabierając prawa czytania i wykonywania swojego katalogu (`chmod g-rwx,o-rwx ~`). Tylda oznacza katalog domowy użytkownika wywołującego polecenie. Po jego wykonaniu jedynie my (oraz użytkownik `root`, który może wszystko) będziemy mogli zaglądać do naszego katalogu domowego. Z drugiej strony niekoniecznie jest to zawsze pożądane – bo blokuje to dostęp również niektórym programom uruchomionym jako inny użytkownik[3].

[1] Istnieją też pojęcia tzw. `suid`, `guid` oraz `sticky bit`, które tu można spotkać oznaczone literami `s` i `t`, niemniej nie będziemy się nimi zajmować.

[2] Raz od katalogu, w którym się znajduje, drugi od elementu `.`, kolejne od elementu `..` w każdym podkatalogu

[3] Np. serwer www – przez który moglibyśmy udostępniać zawartość katalogu `public_html` (widoczna jest

Superużytkownik – w systemie Linux zwykły użytkownik jest objęty pewnymi ograniczeniami. Może jak najbardziej korzystać z większości programów i zarządzać swoimi własnymi plikami. Nie ma jednak uprawnień do ingerencji w ustawienia innych użytkowników i całego systemu, a także dostępu do odczytu niektórych danych. Nie ma np. prawa instalować, aktualizować i usuwać oprogramowania (chyba że poza menadżerem pakietów, na własny użytek, we własnym katalogu domowym), zmieniać właściciela plików poleceniem `chown`, ani podglądać plików takich jak `/etc/shadow` (w którym zebrane są zaszyfrowane hasła wszystkich użytkowników) i wiele logów systemowych. Do przeprowadzania tego typu operacji służy konto superużytkownika (`root`).

Do uzyskiwania supermocy `roota` służy polecenie `su -`[4]. Po wywołaniu `su` zażądane od nas będzie hasło `roota`, zatem oczywiście musimy je znać.

W wielu systemach konto `root` nie ma ustawionego hasła, więc nie można się na nie zalogować w standardowy sposób (w tym poprzez `su`). Zamiast tego preferowane jest użycie polecenia `sudo` – np. `sudo su` by uzyskać to samo co poleceniem `su`. Tutaj również żądane jest od nas hasło, ale jest to nasze własne hasło, a nie `roota`.

Obydwa podejścia mają swoje dobre i złe strony oraz zwolenników[5]. W przypadku typowego osobistego komputera, z którego korzysta tylko jeden użytkownik i robi to świadomie (dba o zasady bezpieczeństwa, zachowuje ostrożność podczas ingerowania w ważne pliki systemowe itp.) nie jest to aż tak istotny wybór – po prostu warto być świadomym istnienia obu tych mechanizmów i tego, że różne systemy mogą domyślnie wybierać któryś z nich. Niezależnie od tego typu dywagacji należy przede wszystkim pamiętać, by używać konta `root` ostrożnie i odpowiedzialnie – czyli w szczególności uważnie czytać wywoływane polecenia, nie wykonywać nieznanych programów i ograniczyć operacje na koncie superużytkownika do niezbędnego minimum!

Jak zacząć?

- Pracownia lub sala pracy własnej (1.23) – stacjonarnie.
- Zdalnie: `https://www.fuw.edu.pl/okwf-uslugi-stud.html`. Można zalogować się

pod adresem `studenci.fuw.edu.pl/~<nazwa_użytkownika>`).

[4]Myślnik na końcu nie jest konieczny, ale bywa wygodny. Powoduje on, że zmienne środowiskowe ustawiane są w taki sposób, jakbyśmy faktycznie logowali się jako `root`. W praktyce m.in. sprawia to, że domyślnie widoczne stają się dla nas programy w katalogu `/sbin`, dzięki czemu możemy np. napisać `reboot` zamiast `/sbin/reboot` by zrestartować komputer. Przeniesieni też zostajemy do katalogu `/root`. W przypadku braku myślnika zachowywane są zmienne i katalog takie jakie były dotychczas (co też może być w pewnych sytuacjach korzystne).

[5]Np. korzystając z `sudo` uprawnień można udzielać różnym użytkownikom w różnym zakresie, nie tak zerojedynkowo jak w przypadku `su` – gdy udostępniamy komuś hasło `roota` bądź nie. Jest to też wygodne gdy np. z jakiegoś powodu chcemy jednemu z użytkowników zmienić uprawnienia administracyjne – wówczas nie musimy zmieniać hasła `roota` (i podawać go na nowo wszystkim innym adminom), wystarczy modyfikacja w pliku `/etc/sudoers` dotycząca tego jednego użytkownika. Ponadto świadomość, że w systemie jest konto o tak dużych uprawnieniach i standardowej nazwie może stanowić łakomy kąsek dla włamywaczy.

Z drugiej strony, możliwość zdalnego logowania (przez `ssh`) jako `root` zwykle łatwo wyłączyć, zresztą od lat praktyką jest jej ograniczanie w już domyślnie instalowanych ustawieniach, więc użytkownik nie musi się tym sam zajmować. Możemy też (i powinniśmy) ustawić wystarczająco silne hasło `roota`, by włamanie w inny sposób (niż przez `ssh`) było bardzo trudne. Przy cofaniu uprawnień admina z kolei należy mieć na uwadze, że jeśli jakiś użytkownik miałby złe intencje, to wystarczy, że miałby uprawnienia `roota` choćby przez chwilę, by zainstalować sobie alternatywne metody obejścia zabezpieczeń.

Co więcej, jeśli korzystamy z `sudo` i mamy uprawnienia do administrowania systemem, a nasze hasło zostanie przez kogoś poznane, nie ma już żadnej dodatkowej warstwy ochronnej, jaką byłaby konieczność znajomości również hasła `roota` (inna sprawa, że uzyskanie dostępu do naszych prywatnych plików może być znacznie gorsze, niż zdobycie przez napastnika uprawnień admina).

Dlatego też pytanie czy `su` czy też `sudo` jest bardziej bezpieczne budzi od lat burzliwe dyskusje i jak widać tak naprawdę odpowiedź na nie jest dość sytuacyjna.

przez `ssh`[6] na komputer `pracownia.okwf.fuw.edu.pl`. Jest on jednakże widoczny jedynie w wewnętrznej sieci wydziałowej, toteż najpierw prawdopodobnie trzeba zalogować się na (widoczny z zewnątrz) komputer „przesiadkowy" `tempac.fuw.edu.pl`.

- LiveUSB – popularne dystrybucje pozwalają wgrać obraz systemu na pendrive-a i uruchamiać system za jego pomocą, pozwala to na komfortowe testowanie bez większego ryzyka (chyba, że intencjonalnie zamontujemy partycje z dotychczasowym systemem). Szczegółowe instrukcje najlepiej czerpać ze strony konkretnej dystrybucji. Czasami w obiegu są także gotowe LiveDVD (dołączane np. do czasopism komputerowych).
- Instalacja na tym samym dysku co Windows – zmniejszyć partycję z systemem Windows i w wolnym miejscu zainstalować Linuxa. Instalację przeprowadza się zwykle z poziomu LiveUSB/LiveDVD.
- Instalacja na dodatkowym dysku – bardzo bezpieczne i wygodne rozwiązanie.
- Windows Subsystem for Linux: `https://learn.microsoft.com/en-us/windows/wsl/install`. System Linux zostanie zainstalowany jako pewnego rodzaju maszyna wirtualna w istniejącym systemie Windows. Chyba najlepsza opcja dla osób, które nie chcą instalować całego nowego systemu, a jedynie mieć dostęp do omawianych na wykładzie programów. Instalacja jest wyjątkowo prosta i nieinwazyjna.

Pytania kontrolne

1. Jakiego polecenia powłoki użyć w systemie Linux do przeniesienia pliku `plik1.txt` w katalogu `katalog1` do katalogu `katalog2`?
2. W jakim katalogu w systemie Linux należy szukać pliku z hasłami użytkowników?
3. Co oznacza `drwxr-x---` wyświetlone przez polecenie `ls`?
4. Do czego służy komenda `ls /root`? Czemu gdy wywołamy ją jako zwykły użytkownik najprawdopodobniej uzyskamy informację o braku uprawnień? Co zrobić żeby polecenie to osiągnęło zamierzony skutek?

[6] W systemie Windows często proponuje się `PuTTY` lub inny podobny program, ale w nowych wersjach systemu można również po prostu doinstalować `ssh` (Ustawienia→System→Funkcje opcjonalne→Klient OpenSSH, a potem wywołujemy `ssh` w Wierszu polecenia)