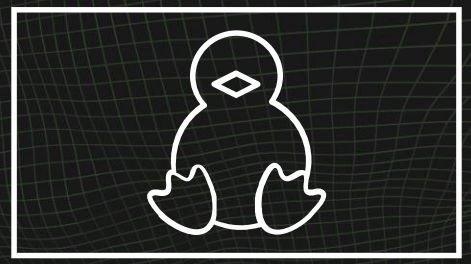
Podstawy Linux

Zajęcia nr 1



Spis treści

Wstęp

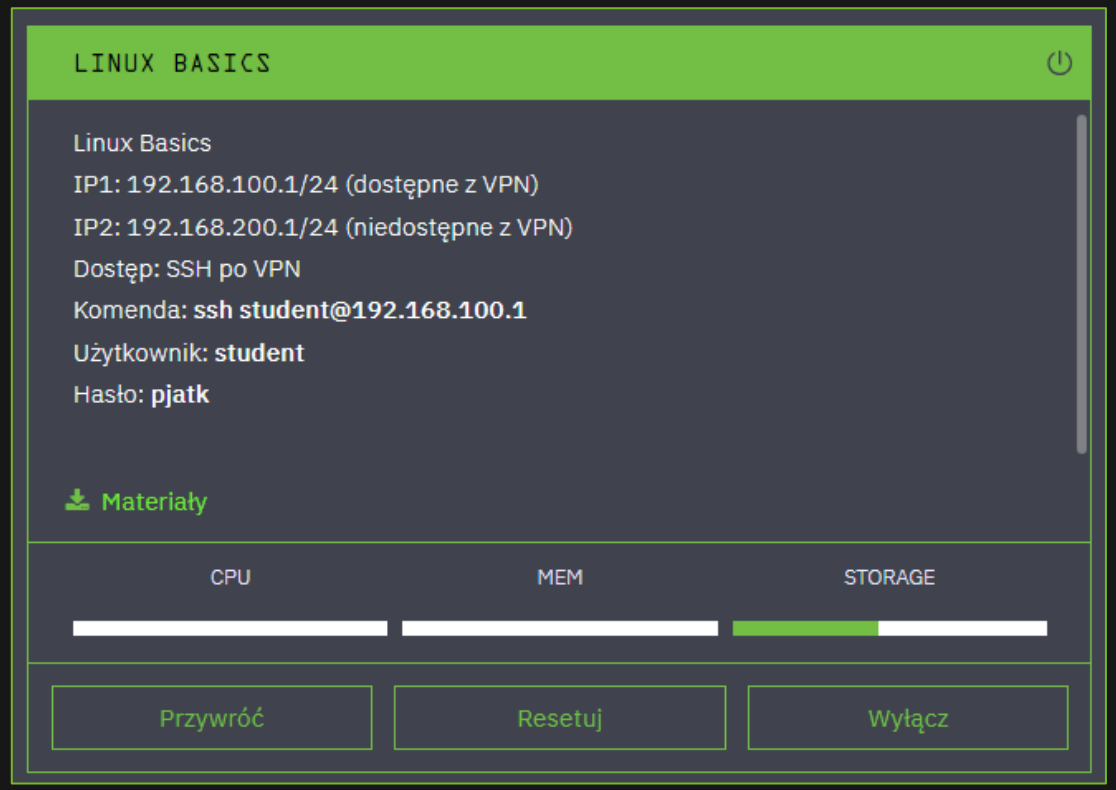
Student nauczy się podstaw obsługi systemu Linux, koncentrując się na najważniejszych komendach, które każdy użytkownik powinien znać. Linux, jako wszechstronny system operacyjny, oferuje bogaty zestaw narzędzi umożliwiających efektywne zarządzanie plikami, procesami i konfiguracją systemu.

System operacyjny Kali Linux ma już zainstalowane wszystkie niezbędne narzędzia (chyba, że zadanie mówi inaczej). Jeśli korzystasz z innego systemu operacyjnego, musisz zadbać o nie samodzielnie.

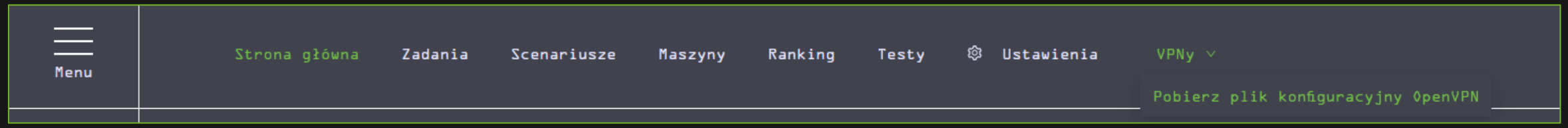
Przygotowanie środowiska

Uruchamianie maszyn

Przed rozpoczęciem upewnij się, że maszyna *Linux Basics* jest włączona. Jeżeli coś się zepsuje, w każdej chwili możesz ją przywrócić do stanu początkowego.



Połączenie OpenVPN



Połączenie OpenVPN jest niezbędne do rozwiązania zadań.
Aby połączyć się z siecią VPN, należy pobrać plik konfiguracyjny OpenVPN z rozwijanego menu „VPNy” na stronie głównej.
Następnie należy wykonać komendę:

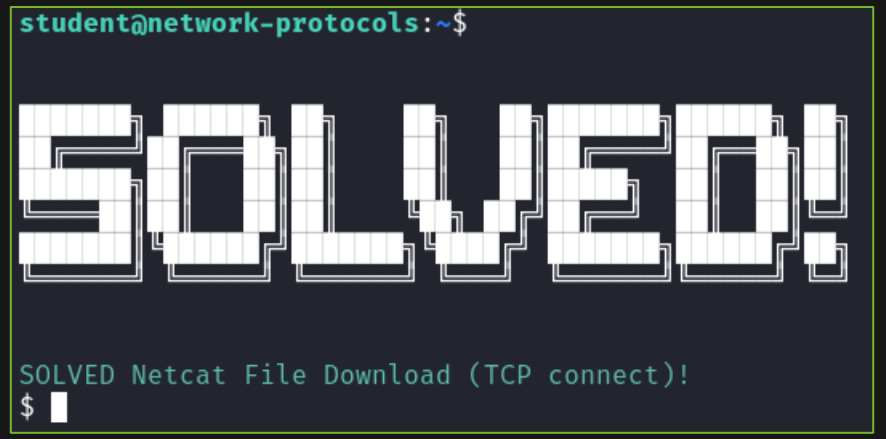
```
sudo openvpn /sciezka/do/pliku.ovpn
```

Śledzenie wyników w konsoli

Krok opcjonalny - w celu śledzenia aktualnych postępów w konsoli, istnieje możliwość połączenia się za maszyną z zadaniami, która w czasie rzeczywistym będzie wyświetlać wszystkie wykonane zadania:

```
ssh student@192.168.100.1 #hasło: pjatk
```

Przykładowy ekran po rozwiązaniu zadania:



Wyniki można śledzić również na platformie WWW z poziomu mapy na stronie głównej, zakładki Zadania lub Rankingu.

Legenda

cat /etc/passwd – komendę należy wykonać na maszynie Kali
cat /etc/passwd – komendę należy wykonać na maszynie Ćwiczenia
cat /etc/passwd – komendę należy wykonać na maszynie Pentest

Teoria

Połączenie SSH

SSH (Secure Shell) to protokół umożliwiający bezpieczne łączenie się zdalnie z serwerem. Istnieją dwa główne sposoby uwierzytelniania: hasłem oraz kluczem SSH. Logowanie hasłem jest proste, ale mniej bezpieczne – narażone na ataki siłowe i wycieki. Klucze SSH oferują wyższy poziom ochrony, ponieważ wymagają dopasowanej pary kluczy prywatnego i publicznego. Klucz publiczny umieszcza się na serwerze, a prywatny pozostaje na urządzeniu użytkownika, co eliminuje konieczność wpisywania hasła przy każdym logowaniu. Po skonfigurowaniu kluczy warto wyłączyć logowanie hasłem, by ograniczyć ryzyko nieautoryzowanego dostępu.

Połączenie SSH

Aby skorzystać z logowania kluczem SSH, najpierw należy wygenerować parę kluczy. Służy do tego polecenie ssh-keygen. Po jego wykonaniu powstają dwa pliki: klucz prywatny (id_rsa), który pozostaje na komputerze użytkownika, oraz klucz publiczny (id_rsa.pub), który należy umieścić na serwerze. Najprostszą metodą przesłania klucza jest użycie:

ssh-copy-iduser@hostname.example.com

Polecenie to automatycznie dodaje klucz publiczny do pliku ~/.ssh/authorized_keys na serwerze.

Połączenie SSH

Jeśli ssh-copy-id nie jest dostępne, można ręcznie przesłać klucz za pomocą SCP:
scp ~/.ssh/id_rsa.pub user@hostname.example.com:.ssh/authorized_keys
W tym przypadku plik authorized_keys zostanie nadpisany, więc jeśli wcześniej zawierał inne klucze, zostaną one usunięte. Katalog .ssh na serwerze musi już istnieć – jeśli go nie ma, należy go utworzyć i nadać mu odpowiednie uprawnienia (chmod 700 ~/.ssh). Po dodaniu klucza można logować się bez podawania hasła, wpisując ssh user@hostname.example.com.

Różne dystrybucje Linuxa i ich zastosowania

System operacyjny Linux występuje w wielu dystrybucjach, które różnią się pod względem zastosowania, struktury oraz domyślnie zainstalowanego oprogramowania. Dystrybucje takie jak Ubuntu, Fedora czy openSUSE są popularne wśród użytkowników domowych i biurowych, oferując przyjazne środowisko oraz wsparcie dla szerokiej gamy aplikacji. Z kolei CentOS oraz Red Hat Enterprise Linux (RHEL) znajdują zastosowanie w środowiskach korporacyjnych i serwerowych, zapewniając stabilność oraz długoterminowe wsparcie. Istnieją również specjalistyczne dystrybucje, takie jak Arch Linux, skierowany do zaawansowanych użytkowników, czy Kali Linux, przeznaczony do testów penetracyjnych i cyberbezpieczeństwa.

Kali Linux

Kali Linux jest jedną z najpopularniejszych dystrybucji wykorzystywanych w testach penetracyjnych i badaniach nad cyberbezpieczeństwem. Jest to system bazujący na Debianie, co oznacza, że korzysta z jego struktury pakietów oraz repozytorium. Debian jest znany z wysokiej stabilności i elastyczności, co czyni go doskonałą bazą. Główną cechą tej dystrybucji jest jej bogaty zestaw narzędzi przeznaczonych do testowania zabezpieczeń, takich jak:

- Metasploit Framework - platforma do testów penetracyjnych i wykorzystywania podatności,
- Nmap - zaawansowany skaner sieciowy,
- Wireshark - narzędzie do analizy ruchu sieciowego,
- John the Ripper - program do łamania haseł,
- Burp Suite - zestaw narzędzi do testowania bezpieczeństwa aplikacji webowych.

Kali Linux

Dzięki preinstalowanym narzędziom oraz zoptymalizowanej konfiguracji Kali Linux pozwala pentesterom szybko przystąpić do audytowania systemów i aplikacji. Ważnym aspektem tej dystrybucji jest również możliwość uruchamiania jej w trybie Live, co oznacza, że można ją wykorzystać bez konieczności instalacji na dysku twardym.

Dlaczego Linux jest popularniejszy w pentestach?

Linux, a szczególnie jego specjalistyczne dystrybucje jak Kali Linux, jest preferowany przez pentesterów z kilku powodów:

- Dostęp do narzędzi open-source - Większość narzędzi do testów penetracyjnych jest rozwijana pod Linuxem, co sprawia, że działają w nim bardziej stabilnie i wydajnie.
- Elastyczność i kontrola - Linux pozwala na lepszą i wygodniejszą kontrolę nad systemem, łącznie z konfiguracją sieci, uprawnieniami i skryptami automatyzującymi pracę.
- Bezpieczeństwo - Większość wirusów i złośliwego oprogramowania jest dedykowana przeciw użytkownikom systemów firmy Microsoft, co sprawia, że praca na Linuxie jest bezpieczniejsza.
- Kompaktowość - Linux może działać na starszym sprzęcie, a jego wersje Live pozwalają na szybkie uruchamianie i testowanie różnych środowisk.

Kiedy Windows jest konieczny?

Mimo że Linux oferuje szerokie możliwości w zakresie pentestingu, w niektórych przypadkach Windows jest niezbędny. Oto kilka powodów:

- Testowanie aplikacji Windowsowych - Wiele organizacji korzysta z aplikacji natywnych dla Windowsa.
- Integracja z systemami Active Directory - Windows jest dominującym systemem w środowiskach korporacyjnych.
- Analiza malware dla Windows - Większość złośliwego oprogramowania jest tworzona z myślą o systemie Windows.
- Użycie specyficznych narzędzi - Niektóre komercyjne aplikacje do testów zabezpieczeń mają lepsze wsparcie lub wyłącznie wersję na Windows.

W praktyce wielu specjalistów korzysta z obu systemów, dobierając odpowiednie narzędzia do konkretnego zadania.

Najważniejsze katalogi

W katalogu głównym znajdziesz kilka podkatalogów, które mają swoje konkretne zastosowania. Oto niektóre z nich:

- **/bin** – zawiera podstawowe polecenia systemowe i wykonywalne pliki, takie jak ls, cp, mv, które są niezbędne do pracy systemu w trybie użytkownika i podczas uruchamiania.
- **/dev** – przechowuje pliki urządzeń (device files), które reprezentują sprzęt i urządzenia systemowe, np. dyski twarde (/dev/sda), terminale (/dev/tty) czy pamięć USB.
- **/etc** – zawiera pliki konfiguracyjne systemu i aplikacji, takie jak passwd, fstab czy hosts, odpowiadające za ustawienia systemowe.
- **/home** – przechowuje katalogi domowe użytkowników, gdzie znajdują się ich pliki osobiste, konfiguracje i dane (/home/użytkownik).
- **/media** – służy do automatycznego montowania nośników zewnętrznych, takich jak płyty CD, DVD, pendrive’y czy dyski przenośne.
- **/tmp** – tymczasowy katalog do przechowywania plików tworzonych przez aplikacje i procesy, które są usuwane po restarcie systemu lub po określonym czasie.
- **/var** – zawiera dane zmienne, takie jak logi systemowe (/var/log), pliki tymczasowe usług, cache, kolejkowane e-maile czy dane baz danych.

Podstawy obsługi terminala

Kiedy zaczynamy przygodę z Linuxem, najwięcej czasu spędzamy w terminalu. Dla kogoś przyzwyczajonego do klikania w okienka i ikony może to brzmieć jak cofnięcie się do epoki komputerów sprzed 30 lat. Ale prawda jest taka, że terminal to najpotężniejsze narzędzie w systemie. Z czasem odkrywamy, że zamiast pięciu kliknięć wystarczy jedno polecenie – a jeśli dodamy do tego znajomość skrótów klawiaturowych, praca staje się naprawdę szybka i wygodna. Przejdźmy więc krok po kroku przez podstawowe skróty, które każdy użytkownik Kali Linux (i nie tylko) powinien znać.

Podstawy obsługi terminala

Autouzupełnianie – [TAB]

To prawdopodobnie pierwszy skrót, którego używa każdy początkujący. Wpisujesz fragment polecenia albo nazwę pliku, naciskasz [TAB], i Linux sam dokańcza resztę.Jeśli jest kilka możliwości – na przykład wpiszesz cd Do i w katalogu są „Documents” i „Downloads” – wystarczy nacisnąć [TAB] dwa razy, a terminal pokaże listę pasujących opcji. Dzięki temu nie tylko oszczędzasz czas, ale też unikasz literówek.

Poruszanie się po poleceniu

Zdarza Ci się napisać długą komendę i zauważyć, że literówka jest na samym początku? Nie musisz kasować wszystkiego.

- [Ctrl] + A – przenosi kursor na początek linii.
- [Ctrl] + E – przesuwa go na koniec.
- [Ctrl] + F / [Ctrl] + B – przesuwają o jeden znak.
- [Alt] + F / [Alt] + B – skaczą po całych słowach.

Te skróty są mało efektowne, ale w praktyce używa się ich ciągle. To taka mała zmiana, która nagle sprawia, że praca jest o wiele wygodniejsza.

Podstawy obsługi terminala

Usuwanie tekstu

Kiedy już pomylimy się w środku komendy, można szybko pozbyć się fragmentu zamiast kasować znak po znaku:

- [Ctrl] + U – kasuje wszystko od kursora do początku linii.
- [Ctrl] + K – usuwa od kursora do końca.
- [Ctrl] + W – wymazuje całe poprzednie słowo.
- I tu przychodzi najlepsze: usunięte kawałki można „wkleić” z powrotem przy pomocy [Ctrl] + Y. To taki mały podręczny schowek terminala.

Kontrola nad procesami

W Linuxie nie musisz ograniczać się do jednej rzeczy na raz – terminal może obsługiwać kilka procesów równolegle. I tu wchodzą w grę skróty:

- [Ctrl] + C – zatrzymuje działający program (np. skan w nmapie, który trwa zbyt długo).
- [Ctrl] + Z – wstrzymuje proces i przenosi go w tło.
- Potem możesz użyć fg (foreground), żeby go wznowić, albo bg, żeby dalej działał w tle.

To bardzo praktyczne, bo nie trzeba otwierać kilku terminali tylko po to, żeby odpalić kilka programów.

Podstawy obsługi terminala

Inne skróty, które warto znać

- [Ctrl] + D – zamyka sesję terminala (tak jakbyś wpisał exit).
- [Ctrl] + L – czyści ekran; to odpowiednik clear.
- [Ctrl] + R – pozwala przeszukiwać historię komend. Wpisujesz kilka liter i terminal sam podpowiada wcześniejsze polecenia.
- Strzałki [↑]/[↓] – klasyka: przewijanie historii poleceń.

Drobne wygody

- [Ctrl] + + / [Ctrl] + - – powiększanie i pomniejszanie tekstu. Idealne, gdy pracujesz na laptopie i chcesz lepiej widzieć wpisywane komendy.
- [Ctrl] + Shift + C / [Ctrl] + Shift + V – kopiowanie i wklejanie w terminalu. Standardowe [Ctrl] + C/V nie działają, bo [Ctrl] + C zatrzymuje procesy.

Podstawy obsługi terminala

Trochę sprytu z historią

Linux zapamiętuje Twoje komendy. To ogromne ułatwienie:

- !! – powtarza ostatnią komendę.
- !nazwa – uruchamia ostatnie polecenie zaczynające się od „nazwa”.
- history – pokazuje całą listę poleceń, które wpisywałeś wcześniej.

To szczególnie ważne w Kali Linux, gdzie często uruchamia się długie i skomplikowane skrypty albo zestawy parametrów.

Aktualizacja systemu

W systemie Kali Linux do zarządzania aktualizacjami oprogramowania używa się polecenia apt, które obsługuje pakiety w formacie .deb. Oto podstawowe komendy i ich znaczenie:

- **apt update**
  - Odświeża listę dostępnych pakietów, pobierając najnowsze informacje z repozytoriów.
- **apt upgrade**
  - Aktualizuje wszystkie zainstalowane pakiety do najnowszych wersji dostępnych w repozytoriach.
  - Instaluje aktualizacje tylko wtedy, gdy nie wymagają one usunięcia ani zmiany zależności innych pakietów.
- **apt full-upgrade** lub **apt dist-upgrade**
  - Oprócz aktualizacji pakietów, potrafi dodawać nowe zależności i usuwać pakiety, jeśli jest to konieczne do przeprowadzenia pełnej aktualizacji.

Aktualizacja systemu

- **apt autoremove**
  - Usuwa pakiety, które zostały zainstalowane automatycznie jako zależności, ale nie są już potrzebne po odinstalowaniu głównych programów. Gdy usuniesz program, jego zależności mogą zostać w systemie. apt autoremove je wyczyści.
- Inne przydatne polecenia
  - Sprawdzenie dostępnych aktualizacji: **apt list --upgradable**
  - Czyszczenie niepotrzebnych plików cache: **apt clean**

Aktualizacja systemu

Regularne aktualizacje: W systemie Kali warto aktualizować pakiety często, aby mieć dostęp do najnowszych narzędzi bezpieczeństwa.

Backup przed dużymi zmianami: Przed użyciem full-upgrade w środowisku produkcyjnym warto wykonać kopię zapasową.

Uwaga na zależności: full-upgrade może zmieniać konfigurację systemu, dlatego zaleca się dokładne sprawdzenie listy zmian przed zatwierdzeniem.

Dodatkowe informacje

Polecenie **man** w Linux służy do wyświetlania stron podręcznika (manuali) z dokumentacją dotyczącą poleceń systemowych, funkcji, konfiguracji i innych elementów systemu. Wpisując **man [nazwa_polecenia]**, np. **man ls**, użytkownik otrzymuje szczegółowe informacje o tym poleceniu, takie jak opis, składnia i dostępne opcje.

Dodatkowe informacje

W systemie Linux kombinacja **Ctrl+C** służy do przerywania wykonywania bieżącego polecenia lub procesu uruchomionego w terminalu. Wysyła ona sygnał SIGINT (interrupt), który informuje proces o konieczności zakończenia działania.

Dodatkowe informacje

Kombinacja **Ctrl+R** w terminalu Linux uruchamia interaktywne wyszukiwanie wsteczne (reverse search) w historii poleceń.

- Po naciśnięciu **Ctrl+R** możesz zacząć wpisywać fragment wcześniej używanego polecenia, a terminal automatycznie wyszuka i wyświetli najbliższe pasujące polecenie z historii.
- Ponowne naciśnięcie **Ctrl+R** przeszukuje historię dalej, znajdując starsze dopasowania.
- Aby zaakceptować znalezione polecenie, wystarczy nacisnąć Enter, a jeśli chcesz je edytować przed wykonaniem, użyj strzałek w lewo/prawo.
- Możesz anulować wyszukiwanie, naciskając Ctrl+C lub Esc.

Zadania

Zadanie 1.1: Połączenie SSH

SSH (Secure Shell) to protokół sieciowy służący do bezpiecznego zdalnego logowania oraz przesyłania danych między komputerami. Zapewnia szyfrowanie komunikacji, co chroni przed podsłuchiwaniem, przechwytywaniem sesji i innymi zagrożeniami związanymi z bezpieczeństwem. SSH umożliwia zdalne zarządzanie serwerami, wykonywanie komend, transfer plików i tunelowanie innych protokołów, co czyni go niezastąpionym narzędziem w administracji systemami oraz programowaniu. Dzięki mechanizmom uwierzytelniania, takim jak klucze publiczne, SSH oferuje również wysoką elastyczność i bezpieczeństwo dostępu.

Zadanie 1.1: Połączenie SSH

Aby połączyć się za pomocą ssh z maszyną „Ćwiczenia” wykonaj komendę: `ssh student@192.168.100.1`. Jeśli łączysz się po raz pierwszy, otrzymasz prośbę o potwierdzenie. Wpisz „`yes`”, a po pojawieniu się pytania o hasło wpisz „`pjatk`”.

```
┌──(kali㉿kali)-[~]
└─$ ssh student@192.168.100.1
The authenticity of host '192.168.100.1 (192.168.100.1)' can't be established.
ED25519 key fingerprint is SHA256:jeAF0Twljj9oLSJYwWGetHe5×0zJZDc0tQEgnhPnHdg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.100.1' (ED25519) to the list of known hosts.
student@192.168.100.1's password:
Linux linux-basics 6.1.0-22-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.94-1 (2024-06-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
student@linux-basics:~$
```

Zadanie 1.2: Użytkownik

Na początku poznamy kilka sposób poznania nazwy użytkownika. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.2: Użytkownik

`whoami` - wyświetla nazwę bieżącego użytkownika zalogowanego w systemie.

```
student@linux-basics:~$ whoami
student
student@linux-basics:~$
Solved (1/4) Current User: Who am I?!
$
```

Zadanie 1.2: Użytkownik

`echo $USER` - wyświetla nazwę użytkownika przypisaną do zmiennej środowiskowej $USER.

```
student@linux-basics:~$ echo $USER
student
student@linux-basics:~$
Solved (2/4) Current User: $USER variable!
$
```

Zadanie 1.2: Użytkownik

env - wyświetla listę wszystkich zmiennych środowiskowych i ich wartości.

```
student@linux-basics:~$ env
SHELL=/bin/bash
PWD=/home/student
LOGNAME=student
XDG_SESSION_TYPE=tty
MOTD_SHOWN=pam
HOME=/home/student
LANG=C.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00
:su=37;41:sg=30;43:ca=00:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;3
1:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01
;31:*.zip=01;31:*.z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*
.tzst=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:
*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31
:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.avif=01;35:*.jpg=01;35:*.jpeg=01;
35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm
=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.
mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.webp=01;35:*.ogm=01;35:*.mp4=01;3
5:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*.rm=01;35:*.rmvb=01;
35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;3
5:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;
36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=0
0;36:*.spx=00;36:*.xspf=00;36:*~=00;90:*#=00;90:*.bak=00;90:*.old=00;90:*.orig=00;90:*.part=00;90:*.rej=00;
90:*.swp=00;90:*.tmp=00;90:*.dpkg-dist=00;90:*.dpkg-old=00;90:*.ucf-dist=00;90:*.ucf-new=00;90:*.ucf-old=00
;90:*.rpmnew=00;90:*.rpmorig=00;90:*.rpmsave=00;90:
SSH_CONNECTION=10.65.0.6 60558 192.168.100.1 22
XDG_SESSION_CLASS=user
TERM=xterm-256color
USER=student
SHLVL=1
XDG_SESSION_ID=9
XDG_RUNTIME_DIR=/run/user/1001
SSH_CLIENT=10.65.0.6 60558 22
PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1001/bus
SSH_TTY=/dev/pts/0
```

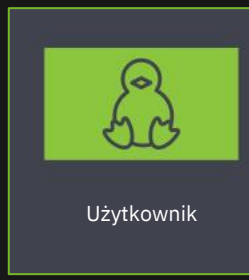
Zadanie 1.2: Użytkownik

`id` - wyświetla informacje o bieżącym użytkowniku, takie jak identyfikator użytkownika (UID), identyfikator grupy (GID) oraz członkostwo w grupach.

```
student@linux-basics:~$ id
uid=1001(student) gid=1001(student) groups=1001(student)
student@linux-basics:~$
Solved (4/4) Current User: List ids and groups of current user!
$
```

SOLVED!

```
SOLVED Current User!
$
```



Zadanie 1.3: Inni użytkownicy

Kolejny zestaw komend pozwoli nam sprawdzić nazwy innych użytkowników. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.3: Inni użytkownicy

`who` - wyświetla listę użytkowników aktualnie zalogowanych do systemu.

```
student@linux-basics:~$ who
root     tty1         2024-07-25 14:04
student  pts/0        2024-07-26 07:14 (10.65.0.6)
student@linux-basics:~$
```

Zadanie 1.3: Inni Użytkownicy

`cat /etc/passwd` - wyświetla zawartość pliku /etc/passwd, który zawiera podstawowe informacje o wszystkich użytkownikach systemu, takie jak nazwa użytkownika, numer identyfikacyjny użytkownika (UID), numer identyfikacyjny grupy (GID), pełna nazwa użytkownika, katalog domowy oraz powłoka logowania.

```
student@linux-basics:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
systemd-timesync:x:997:997:systemd Time Synchronization:/:/usr/sbin/nologin
uuidd:x:100:105::/run/uuidd:/usr/sbin/nologin
messagebus:x:101:106::/nonexistent:/usr/sbin/nologin
systemd-resolve:x:996:996:systemd Resolver:/:/usr/sbin/nologin
tcpdump:x:102:107::/nonexistent:/usr/sbin/nologin
sshd:x:103:65534::/run/sshd:/usr/sbin/nologin
polkitd:x:995:995:polkit:/nonexistent:/usr/sbin/nologin
Debian-exim:x:104:111::/var/spool/exim4:/usr/sbin/nologin
student:x:1001:1001::/home/student:/bin/bash
student@linux-basics:~$
Solved (2/3) Other Users: passwd file!
$
```

Zadanie 1.3: Inni użytkownicy

`last` - wyświetla listę ostatnich logowań użytkowników do systemu, wraz z informacjami o czasie i dacie logowania oraz trwaniu sesji.

```
student@linux-basics:~$ last
student  pts/0        10.65.0.6        Fri Jul 26 07:14   still logged in
root     tty1                          Thu Jul 25 14:04   still logged in
reboot   system boot  6.1.0-22-amd64   Thu Jul 25 14:04   still running
reboot   system boot  6.1.0-22-amd64   Thu Jul 25 14:03   still running
root     tty1                          Thu Jul 25 14:00 - crash  (00:03)
reboot   system boot  6.1.0-22-amd64   Thu Jul 25 14:00   still running
root     tty1                          Thu Jul 25 13:59 - crash  (00:00)
reboot   system boot  6.1.0-22-amd64   Thu Jul 25 13:59   still running

wtmp begins Thu Jul 25 13:55:21 2024
student@linux-basics:~$
Solved (3/3) Other Users: Who was there in the past?!
$

 ____   ___  _     _   _ _____ ____  _
/ ___| / _ \| |   | | | | ____|  _ \| |
\___ \| | | | |   | | | |  _| | | | | |
 ___) | |_| | |___| |_| | |___| |_| |_|
|____/ \___/|_____|\___/|_____|____/(_)

SOLVED Other Users!
$
```

Zadanie 1.4: System operacyjny

Dzięki kolejnemu zestawowi komend dowiemy się możliwie najwięcej o używanym systemie operacyjnym. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.4: System operacyjny

`cat /etc/issue` - wyświetla zawartość pliku /etc/issue, który zawiera krótki komunikat powitalny lub informacje o wersji systemu operacyjnego wyświetlane przed logowaniem.

```
student@linux-basics:~$ cat /etc/issue
Debian GNU/Linux 12 \n \l

student@linux-basics:~$
```

Zadanie 1.4: System operacyjny

`cat /etc/os-release` - wyświetla szczegółowe informacje o systemie operacyjnym, takie jak jego nazwa, wersja i identyfikator, zapisane w pliku /etc/os-release.

```
student@linux-basics:~$ cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 12 (bookworm)"
NAME="Debian GNU/Linux"
VERSION_ID="12"
VERSION="12 (bookworm)"
VERSION_CODENAME=bookworm
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
student@linux-basics:~$
Solved (2/4) Operating System: More detailed distribution info!
$
```

Zadanie 1.4: System operacyjny

`uname -a` - wyświetla pełne informacje o systemie, w tym nazwę jądra, nazwę hosta, wersję jądra, datę kompilacji oraz architekturę systemu.

```
student@linux-basics:~$ uname -a
Linux linux-basics 6.1.0-22-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.94-1 (2024-06-21) x86_64 GNU/Linux
student@linux-basics:~$
Solved (3/4) Operating System: Architecture and kernel version!
$
```

Zadanie 1.4: System operacyjny

`cat /proc/version` - wyświetla szczegóły dotyczące wersji jądra Linux, w tym wersję jądra, datę kompilacji oraz używaną wersję kompilatora.

```
student@linux-basics:~$ cat /proc/version
Linux version 6.1.0-22-amd64 (debian-kernel@lists.debian.org) (gcc-12 (Debian 12.2.0-14) 12.2.0, GNU ld (GN
U Binutils for Debian) 2.40) #1 SMP PREEMPT_DYNAMIC Debian 6.1.94-1 (2024-06-21)
student@linux-basics:~$
Solved (4/4) Operating System: Architecture and kernel version alternative method!
$

 ____   ___  _     _   _____ ____  _
/ ___| / _ \| |   | | | ____|  _ \| |
\___ \| | | | |   | | |  _| | | | | |
 ___) | |_| | |___| |_| |___| |_| |_|
|____/ \___/|_____|\___/|_____|____/(_)

SOLVED Operating System!
$
```

Zadanie 1.5: CPU i pamięć

Tym razem postaramy się lepiej poznać sprzęt, z jakiego korzystamy. W tej części skupimy się na procesorze i pamięci RAM. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.5: CPU i pamięć

`cat /proc/cpuinfo` - wyświetla szczegółowe informacje o procesorze, takie jak model, liczba rdzeni, częstotliwość taktowania oraz cechy procesora.

```
student@linux-basics:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 106
model name      : Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz
stepping        : 6
microcode       : 0×d0003b9
cpu MHz         : 2394.374
cache size      : 24576 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 27
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr
sse sse2 ss syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc
cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer
 aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_single ssbd ibrs ibpb stibp ibrs_en
hanced fsgsbase tsc_adjust bmi1 avx2 smep bmi2 invpcid avx512f avx512dq rdseed adx smap clflushopt clwb avx
512cd avx512bw avx512vl xsaveopt xsavec xsaves arat pku ospke md_clear flush_l1d arch_capabilities
bugs            : spectre_v1 spectre_v2 spec_store_bypass swapgs itlb_multihit mmio_stale_data eibrs_pbrsb
gds bhi
bogomips        : 4788.74
clflush size    : 64
cache_alignment : 64
address sizes   : 43 bits physical, 48 bits virtual
power management:
```

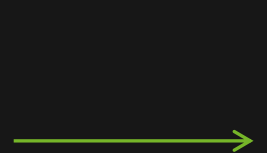
Zadanie 1.5: CPU i pamięć

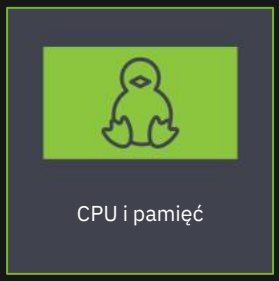
`cat /proc/meminfo` - wyświetla szczegółowe informacje o pamięci systemowej, w tym całkowitą ilość pamięci, używaną pamięć, wolną pamięć oraz inne statystyki dotyczące pamięci.

```
student@linux-basics:~$ cat /proc/meminfo
MemTotal:        2014196 kB
MemFree:         1324840 kB
MemAvailable:    1647808 kB
Buffers:           44780 kB
Cached:           393700 kB
SwapCached:            0 kB
Active:           229676 kB
Inactive:         324288 kB
Active(anon):        556 kB
Inactive(anon):   115684 kB
Active(file):     229120 kB
Inactive(file):   208604 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:             0 kB
SwapFree:              0 kB
Zswap:                 0 kB
Zswapped:              0 kB
Dirty:                12 kB
Writeback:             0 kB
AnonPages:        115608 kB
Mapped:           130416 kB
Shmem:               756 kB
KReclaimable:      33556 kB
Slab:              68416 kB
SReclaimable:      33556 kB
SUnreclaim:        34860 kB
KernelStack:        3296 kB
PageTables:         2292 kB
SecPageTables:         0 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     1007096 kB
Committed_AS:     460848 kB
VmallocTotal:   34359738367 kB
VmallocUsed:       22960 kB
VmallocChunk:          0 kB
Percpu:              888 kB
HardwareCorrupted:     0 kB
AnonHugePages:     36864 kB
ShmemHugePages:        0 kB
ShmemPmdMapped:        0 kB
FileHugePages:         0 kB
FilePmdMapped:         0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
Hugetlb:               0 kB
DirectMap4k:      100224 kB
DirectMap2M:     1996800 kB
DirectMap1G:           0 kB
student@linux-basics:~$
Solved (2/2) CPU and Memory: Memory Information!
$

SOLVED!

SOLVED CPU and Memory!
```





Zadanie 1.6: Sprzęt

Oczywiście procesor i pamięć RAM to nie wszystko. Kolejne polecenia pozwolą nam lepiej poznać hardware. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.6: Sprzęt

`lspci` - wyświetla listę wszystkich urządzeń podłączonych do magistrali PCI w systemie, wraz z ich identyfikatorami i podstawowymi informacjami o sprzęcie.

```
student@linux-basics:~$ lspci
00:00.0 Host bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX Host bridge (rev 01)
00:01.0 PCI bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX AGP bridge (rev 01)
00:07.0 ISA bridge: Intel Corporation 82371AB/EB/MB PIIX4 ISA (rev 08)
00:07.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01)
00:07.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 08)
00:07.7 System peripheral: VMware Virtual Machine Communication Interface (rev 10)
00:0f.0 VGA compatible controller: VMware SVGA II Adapter
00:10.0 SCSI storage controller: Broadcom / LSI 53c1030 PCI-X Fusion-MPT Dual Ultra320 SCSI (rev 01)
00:11.0 PCI bridge: VMware PCI bridge (rev 02)
00:15.0 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.1 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.2 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.3 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.4 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.5 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.6 PCI bridge: VMware PCI Express Root Port (rev 01)
00:15.7 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.0 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.1 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.2 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.3 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.4 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.5 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.6 PCI bridge: VMware PCI Express Root Port (rev 01)
00:16.7 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.0 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.1 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.2 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.3 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.4 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.5 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.6 PCI bridge: VMware PCI Express Root Port (rev 01)
00:17.7 PCI bridge: VMware PCI Express Root Port (rev 01)
00:18.0 PCI bridge: VMware PCI Express Root Port (rev 01)
```

Zadanie 1.6: Sprzęt

`lsusb` - wyświetla listę wszystkich urządzeń USB podłączonych do systemu, w tym informacje o identyfikatorach producenta i produktu, a także numerze magistrali i porcie, do którego urządzenie jest podłączone.

```
student@linux-basics:~$ lsusb
student@linux-basics:~$
Solved (2/5) Hardware: List USB devices!
$
```

Zadanie 1.6: Sprzęt

`lsscsi` - wyświetla listę wszystkich urządzeń SCSI (w tym dysków twardych, napędów optycznych i innych urządzeń) podłączonych do systemu, wraz z ich identyfikatorami i informacjami o urządzeniach.

```
student@linux-basics:~$ lsscsi
[32:0:0:0]   disk    VMware   Virtual disk     2.0   /dev/sda
student@linux-basics:~$
Solved (3/5) Hardware: List drives!
$
```

Zadanie 1.6: Sprzęt

`lshw` - wyświetla szczegółowe informacje o sprzęcie systemu, w tym o procesorze, pamięci, dyskach, urządzeniach peryferyjnych oraz ich konfiguracji.

```
student@linux-basics:~$ lshw
WARNING: you should run this program as super-user.
linux-basics
    description: Computer
    width: 64 bits
    capabilities: smp vsyscall32
  *-core
       description: Motherboard
       physical id: 0
     *-memory
          description: System memory
          physical id: 0
          size: 2GiB
     *-cpu:0
          product: Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz
          vendor: Intel Corp.
          physical id: 1
          bus info: cpu@0
          version: 6.106.6
          width: 64 bits
          capabilities: fpu fpu_exception wp vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp x86-64 constant_tsc arch_perfmon nopl xtopolog
y tsc_reliable nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movb
e popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid_single s
sbd ibrs ibpb stibp ibrs_enhanced fsgsbase tsc_adjust bmi1 avx2 smep bmi2 invpcid avx512f avx512dq rdseed a
dx smap clflushopt clwb avx512cd avx512bw avx512vl xsaveopt xsavec xsaves arat pku ospke md_clear flush_l1d
 arch_capabilities
          configuration: microcode=218104761
     *-cpu:1
          product: Intel(R) Xeon(R) Silver 4314 CPU @ 2.40GHz
          vendor: Intel Corp.
          physical id: 2
          bus info: cpu@1
          version: 6.106.6
          width: 64 bits
          capabilities: fpu fpu_exception wp vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
```

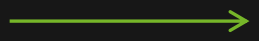
Zadanie 1.6: Sprzęt

`lsblk` - wyświetla informacje o wszystkich dostępnych urządzeniach blokowych w systemie, takich jak dyski twarde i partycje, wraz z ich nazwami, rozmiarami, typem (np. dysk, partycja), oraz punktami montowania.

```
student@linux-basics:~$ lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0    6G  0 disk
├─sda1    8:1    0  5.8G  0 part /
├─sda14   8:14   0    3M  0 part
└─sda15   8:15   0  124M  0 part /boot/efi
student@linux-basics:~$
Solved (5/5) Hardware: List block devices!
$
```

SOLVED!

```
SOLVED Hardware!
$
```



Zadanie 1.7: Zasoby

Dobrze wiedzieć, jakimi zasobami dysponujemy. Następne polecenia dadzą nam informacje m.in. o użyciu pamięci dyskowej, czy uruchomionych procesach. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.7: Zasoby

`df -h` - wyświetla informacje o użyciu przestrzeni dyskowej na zamontowanych systemach plików, prezentując wyniki w formacie czytelnym dla ludzi (z jednostkami takimi jak GB i MB).

```
student@linux-basics:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            965M     0  965M   0% /dev
tmpfs           197M  756K  196M   1% /run
/dev/sda1       5.8G  2.3G  3.2G  42% /
tmpfs           984M     0  984M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
/dev/sda15      124M   12M  113M  10% /boot/efi
tmpfs           197M     0  197M   0% /run/user/0
tmpfs           197M     0  197M   0% /run/user/1001
student@linux-basics:~$
Solved (1/3) Resources: Show disk usage!
$
```

Zadanie 1.7: Zasoby

`free -h` - wyświetla informacje o użyciu pamięci systemowej, w tym o całkowitej, używanej, wolnej i dostępnej pamięci RAM oraz pamięci wymiany, prezentując wyniki w formacie czytelnym dla ludzi.

```
student@linux-basics:~$ free -h
               total        used        free      shared  buff/cache   available
Mem:           1.9Gi       372Mi       1.2Gi       756Ki       465Mi       1.6Gi
Swap:             0B          0B          0B
student@linux-basics:~$
Solved (2/3) Resources: Show memory usage!
$
```

Zadanie 1.7: Zasoby

`top` - wyświetla dynamiczny, na bieżąco aktualizowany widok procesów systemowych, pokazując zużycie CPU, pamięci oraz inne statystyki dotyczące aktywnych procesów.

Uwaga! Po zaliczeniu zadania przerwij jego wykonywanie kombinacją **ctrl+c**.

```
top - 07:42:26 up 17:38,  2 users,  load average: 0.00, 0.00, 0.00
Tasks: 187 total,   1 running, 186 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us, 50.0 sy,  0.0 ni, 50.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1967.0 total,   1275.6 free,    371.9 used,    465.4 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.   1595.1 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
      1 root      20   0  102240  12456   9296 S   0.0   0.6   0:04.01 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.03 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
top - 07:42:35 up 17:38,  2 users,  load average: 0.00, 0.00, 0.00
     11 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_kthread
%Cpu(s):  0.0 us,  0.3 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   1967.0 total,   1275.6 free,    371.8 used,    465.5 buff/cache
MiB Swap:      0.0 total,      0.0 free,      0.0 used.   1595.2 avail Mem

     16 root      rt   0       0      0      0 S   0.0   0.0   0:00.58 migration/0
   2216 root      20   0       0      0      0 I   0.3   0.0   0:02.96 kworker/1:2-events
      1 root      20   0  102240  12456   9296 S   0.0   0.6   0:04.01 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.03 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
```


Zadanie 1.8: Sieć

Oczywiście nasza maszyna musi być podpięta do sieci. Jeśli chcemy dowiedzieć się o niej więcej, potrzebujemy następnego zestawu narzędzi. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.8: Sieć

`ip a` - wyświetla szczegółowe informacje o interfejsach sieciowych, w tym ich adresach IP, maskach sieciowych i innych konfiguracjach związanych z siecią.

```
student@linux-basics:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:50:56:bf:40:33 brd ff:ff:ff:ff:ff:ff
    altname enp3s0
    altname ens160
    inet 192.168.100.1/24 brd 192.168.100.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:febf:4033/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:bf:05:b5 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    altname ens192
    inet 192.168.200.1/24 brd 192.168.200.255 scope global eth1
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:febf:5b5/64 scope link
       valid_lft forever preferred_lft forever
4: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:ad:53:b5:ce brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
student@linux-basics:~$
Solved (1/6) Network: List IP addresses!
```

Zadanie 1.8: Sieć

`ip r` - wyświetla tabelę routingu systemu, która pokazuje dostępne trasy sieciowe oraz przypisane im bramy, interfejsy sieciowe i inne parametry, co jest kluczowe dla zrozumienia, jak ruch sieciowy jest kierowany przez system.

```
student@linux-basics:~$ ip r
default via 192.168.100.254 dev eth0 onlink
10.65.0.0/24 via 192.168.100.250 dev eth0
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1 linkdown
192.168.100.0/24 dev eth0 proto kernel scope link src 192.168.100.1
192.168.200.0/24 dev eth1 proto kernel scope link src 192.168.200.1
student@linux-basics:~$
Solved (2/6) Network: List routing table!
$
```

Zadanie 1.8: Sieć

`ip n` - służy do wyświetlania tabeli sąsiadów sieciowych, która zawiera informacje o odwzorowaniach adresów IP na adresy MAC dla urządzeń w tej samej sieci lokalnej. Jest to narzędzie do zarządzania sąsiadami warstwy 3 (adresy IP) i warstwy 2 (adresy MAC) w sieci, często używane do diagnozowania problemów z połączeniami sieciowymi i analizowania stanu połączeń.

```
student@linux-basics:~$ ip n
192.168.100.254 dev eth1 lladdr 00:50:56:bf:75:b8 STALE
192.168.100.250 dev eth0 lladdr 00:50:56:bf:e8:13 DELAY
192.168.100.254 dev eth0 lladdr 00:50:56:bf:75:b8 REACHABLE
student@linux-basics:~$
```

Zadanie 1.8: Sieć

`cat /etc/resolv.conf` - wyświetla zawartość pliku konfiguracyjnego /etc/resolv.conf, który zawiera informacje o serwerach DNS używanych do rozwiązywania nazw domenowych.

```
student@linux-basics:~$ cat /etc/resolv.conf
# This is /run/systemd/resolve/resolv.conf managed by man:systemd-resolved(8).
# Do not edit.
#
# This file might be symlinked as /etc/resolv.conf. If you're looking at
# /etc/resolv.conf and seeing this text, you have followed the symlink.
#
# This is a dynamic resolv.conf file for connecting local clients directly to
# all known uplink DNS servers. This file lists all configured search domains.
#
# Third party programs should typically not access this file directly, but only
# through the symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a
# different way, replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 8.8.8.8
search .
student@linux-basics:~$
Solved (3/6) Network: Show DNS servers!
$
```

Zadanie 1.8: Sieć

`ss -tp` - wyświetla listę aktywnych połączeń TCP wraz z dodatkowymi informacjami o każdym połączeniu, takimi jak adresy IP i porty źródłowe oraz docelowe.

```
student@linux-basics:~$ ss -tp
State      Recv-Q      Send-Q           Local Address:Port            Peer Address:Port      Process
ESTAB      0           0                192.168.100.1:ssh              10.65.0.6:47956
student@linux-basics:~$
Solved (4/6) Network: List TCP connections!
$
```

Zadanie 1.8: Sieć

`ss -tlp` - wyświetla listę aktywnych nasłuchujących połączeń TCP wraz z informacjami o identyfikatorze procesu (PID) i nazwie programu, który jest odpowiedzialny za każde nasłuchujące gniazdo.

```
student@linux-basics:~$ ss -tlp
State       Recv-Q      Send-Q           Local Address:Port           Peer Address:Port     Process
LISTEN      0           20                   127.0.0.1:smtp                0.0.0.0:*
LISTEN      0           4096              127.0.0.54:domain                0.0.0.0:*
LISTEN      0           4096           127.0.0.53%lo:domain                0.0.0.0:*
LISTEN      0           4096                 0.0.0.0:5355                0.0.0.0:*
LISTEN      0           128                  0.0.0.0:ssh                 0.0.0.0:*
LISTEN      0           20                     [::1]:smtp                   [::]:*
LISTEN      0           4096                    [::]:5355                   [::]:*
LISTEN      0           128                     [::]:ssh                    [::]:*
student@linux-basics:~$
Solved (5/6) Network: List TCP listeners!
$
```

Zadanie 1.9: Procesy

Czy na pewno wiesz, co dzieje się wewnątrz systemu? Jednocześnie uruchomionych jest wiele procesów. Postarajmy się dowiedzieć o nich czegoś więcej. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.9: Procesy

`ps` - wyświetla listę aktualnie działających procesów w systemie, pokazując podstawowe informacje takie jak identyfikator procesu (PID), użytkownik oraz stan procesu.

```
student@linux-basics:~$ ps

Solved (1/3) Processes: List local processes!
$     PID TTY          TIME CMD
   2232 pts/0    00:00:00 bash
   2491 pts/0    00:00:00 ps
student@linux-basics:~$
```

Zadanie 1.9: Procesy

`ps aux` - wyświetla szczegółową listę wszystkich uruchomionych procesów w systemie, niezależnie od użytkownika, z dodatkowymi informacjami takimi jak użycie CPU, pamięci, czas działania i inne atrybuty procesu.

```
student@linux-basics:~$ ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.6 102240 12456 ?        Ss   Jul25   0:04 /sbin/init
root           2  0.0  0.0      0     0 ?        S    Jul25   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   Jul25   0:00 [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   Jul25   0:00 [rcu_par_gp]
root           5  0.0  0.0      0     0 ?        I<   Jul25   0:00 [slub_flushwq]
root           6  0.0  0.0      0     0 ?        I<   Jul25   0:00 [netns]
root           8  0.0  0.0      0     0 ?        I<   Jul25   0:00 [kworker/0:0H-events_highpri]
root          10  0.0  0.0      0     0 ?        I<   Jul25   0:00 [mm_percpu_wq]
root          11  0.0  0.0      0     0 ?        I    Jul25   0:00 [rcu_tasks_kthread]
root          12  0.0  0.0      0     0 ?        I    Jul25   0:00 [rcu_tasks_rude_kthread]
root          13  0.0  0.0      0     0 ?        I    Jul25   0:00 [rcu_tasks_trace_kthread]
root          14  0.0  0.0      0     0 ?        S    Jul25   0:00 [ksoftirqd/0]
root          15  0.0  0.0      0     0 ?        I    Jul25   0:04 [rcu_preempt]
root          16  0.0  0.0      0     0 ?        S    Jul25   0:00 [migration/0]
root          18  0.0  0.0      0     0 ?        S    Jul25   0:00 [cpuhp/0]
root          19  0.0  0.0      0     0 ?        S    Jul25   0:00 [cpuhp/1]
root          20  0.0  0.0      0     0 ?        S    Jul25   0:00 [migration/1]
root          21  0.0  0.0      0     0 ?        S    Jul25   0:00 [ksoftirqd/1]
root          23  0.0  0.0      0     0 ?        I<   Jul25   0:00 [kworker/1:0H-events_highpri]
root          26  0.0  0.0      0     0 ?        S    Jul25   0:00 [kdevtmpfs]
root          27  0.0  0.0      0     0 ?        I<   Jul25   0:00 [inet_frag_wq]
root          28  0.0  0.0      0     0 ?        S    Jul25   0:00 [kauditd]
root          29  0.0  0.0      0     0 ?        S    Jul25   0:00 [khungtaskd]
root          30  0.0  0.0      0     0 ?        S    Jul25   0:00 [oom_reaper]
root          31  0.0  0.0      0     0 ?        I<   Jul25   0:00 [writeback]
root          32  0.0  0.0      0     0 ?        S    Jul25   0:04 [kcompactd0]
root          33  0.0  0.0      0     0 ?        SN   Jul25   0:00 [ksmd]
root          34  0.0  0.0      0     0 ?        SN   Jul25   0:00 [khugepaged]
root          35  0.0  0.0      0     0 ?        I<   Jul25   0:00 [kintegrityd]
root          36  0.0  0.0      0     0 ?        I<   Jul25   0:00 [kblockd]
```

Zadanie 1.9: Procesy

`ps auxf` - wyświetla szczegółową listę wszystkich uruchomionych procesów, prezentując je w formie drzewa hierarchii procesów, co ułatwia zrozumienie relacji między procesami i ich rodzicami.

```
student@linux-basics:~$ ps auxf
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           2  0.0  0.0      0     0 ?        S    Jul25   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [rcu_gp]
root           4  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [rcu_par_gp]
root           5  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [slub_flushwq]
root           6  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [netns]
root           8  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [kworker/0:0H-events_highpri]
root          10  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [mm_percpu_wq]
root          11  0.0  0.0      0     0 ?        I    Jul25   0:00  \_ [rcu_tasks_kthread]
root          12  0.0  0.0      0     0 ?        I    Jul25   0:00  \_ [rcu_tasks_rude_kthread]
root          13  0.0  0.0      0     0 ?        I    Jul25   0:00  \_ [rcu_tasks_trace_kthread]
root          14  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [ksoftirqd/0]
root          15  0.0  0.0      0     0 ?        I    Jul25   0:04  \_ [rcu_preempt]
root          16  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [migration/0]
root          18  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [cpuhp/0]
root          19  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [cpuhp/1]
root          20  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [migration/1]
root          21  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [ksoftirqd/1]
root          23  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [kworker/1:0H-events_highpri]
root          26  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [kdevtmpfs]
root          27  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [inet_frag_wq]
root          28  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [kauditd]
root          29  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [khungtaskd]
root          30  0.0  0.0      0     0 ?        S    Jul25   0:00  \_ [oom_reaper]
root          31  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [writeback]
root          32  0.0  0.0      0     0 ?        S    Jul25   0:04  \_ [kcompactd0]
root          33  0.0  0.0      0     0 ?        SN   Jul25   0:00  \_ [ksmd]
root          34  0.0  0.0      0     0 ?        SN   Jul25   0:00  \_ [khugepaged]
root          35  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [kintegrityd]
root          36  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [kblockd]
root          37  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [blkcg_punt_bio]
root          38  0.0  0.0      0     0 ?        I<   Jul25   0:00  \_ [tpm_dev_wq]
```

Zadanie 1.10: Listowanie plików

Wyświetlenie zawartości katalogu nie jest może zadaniem trudnym, ale i na to znajdziemy kilka sposobów, a wybór odpowiedniego zależy od tego, jakich efektów oczekujemy. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.10: Listowanie plików

`ls` - wyświetla listę plików i katalogów znajdujących się w bieżącym katalogu roboczym.

```
student@linux-basics:~$ ls
student@linux-basics:~$
Solved (1/3) Directory Listing: List current directory!
$
```

Zadanie 1.10: Listowanie plików

`ls -l` - wyświetla szczegółową listę plików i katalogów w bieżącym katalogu roboczym, prezentując dodatkowe informacje takie jak uprawnienia, właściciel, grupa, rozmiar i data ostatniej modyfikacji.

```
student@linux-basics:~$ ls -l
total 0
student@linux-basics:~$
Solved (2/3) Directory Listing: List currect directory using long version!
$
```

Zadanie 1.10: Listowanie plików

`ls -a` - wyświetla wszystkie pliki i katalogi w bieżącym katalogu roboczym, w tym ukryte pliki i katalogi, których nazwy zaczynają się od kropki (.).

```
student@linux-basics:~$ ls -a
.  ..  .bash_logout  .bashrc  .config  .profile  .ssh
student@linux-basics:~$
Solved (3/3) Directory Listing: List hidden files in currect directory!
$
```

SOLVED!

```
SOLVED Directory Listing!
$
```


Zadanie 1.11: Operacje na plikach

Nadszedł czas na kilka kolejnych poleceń, które związane są z operacjami na plikach. Jak utworzyć nowy plik? Skopiować go? Usunąć? Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.11: Operacje na plikach

`touch pjatk1` - tworzy nowy, pusty plik o podanej nazwie lub aktualizuje datę ostatniego dostępu i modyfikacji istniejącego pliku.

```
student@linux-basics:~$ touch pjatk1
student@linux-basics:~$
```

```
student@linux-basics:~$ touch pjatk1
student@linux-basics:~$ ls -l
total 0
-rw-r--r-- 1 student student 0 Jul 26 07:51 pjatk1
student@linux-basics:~$
```

Zadanie 1.11: Operacje na plikach

`echo Hello > pjatk2` - zapisuje tekst "Hello" do pliku o nazwie pjatk2, tworząc nowy plik lub nadpisując zawartość istniejącego.

```
student@linux-basics:~$ echo Hello > pjatk2
student@linux-basics:~$
```

```
student@linux-basics:~$ echo Hello > pjatk2
student@linux-basics:~$ ls -l
total 4
-rw-r--r-- 1 student student 0 Jul 26 07:51 pjatk1
-rw-r--r-- 1 student student 6 Jul 26 07:52 pjatk2
student@linux-basics:~$
```

```
student@linux-basics:~$ cat pjatk2
Hello
student@linux-basics:~$
```

Zadanie 1.11: Operacje na plikach

`cp pjatk2 pjatk3` - kopiuje zawartość pliku pjatk2 do nowego pliku o nazwie pjatk3.

```
student@linux-basics:~$ cp pjatk2 pjatk3
student@linux-basics:~$
```

```
student@linux-basics:~$ cp pjatk2 pjatk3
student@linux-basics:~$ ls -l
total 8
-rw-r--r-- 1 student student 0 Jul 26 07:51 pjatk1
-rw-r--r-- 1 student student 6 Jul 26 07:52 pjatk2
-rw-r--r-- 1 student student 6 Jul 26 07:53 pjatk3
student@linux-basics:~$ cat pjatk3
Hello
student@linux-basics:~$
```

Zadanie 1.11: Operacje na plikach

`mv pjatk2 pjatk4` - przenosi plik pjatk2 do nowej lokalizacji lub zmienia jego nazwę na pjatk4.

```
student@linux-basics:~$ mv pjatk2 pjatk4
student@linux-basics:~$
```

```
student@linux-basics:~$ mv pjatk2 pjatk4
student@linux-basics:~$ ls -l
total 8
-rw-r--r-- 1 student student 0 Jul 26 07:51 pjatk1
-rw-r--r-- 1 student student 6 Jul 26 07:53 pjatk3
-rw-r--r-- 1 student student 6 Jul 26 07:52 pjatk4
student@linux-basics:~$ cat pjatk4
Hello
student@linux-basics:~$
```

Zadanie 1.11: Operacje na plikach

`rm pjatk3` - usuwa plik o nazwie pjatk3 z systemu.

```
student@linux-basics:~$ rm pjatk3
student@linux-basics:~$
```

```
student@linux-basics:~$ rm pjatk3
student@linux-basics:~$ ls -l
total 4
-rw-r--r-- 1 student student 0 Jul 26 07:51 pjatk1
-rw-r--r-- 1 student student 6 Jul 26 07:52 pjatk4
student@linux-basics:~$
```

Zadanie 1.11: Operacje na plikach

`mkdir pjatk` - tworzy nowy katalog o nazwie pjatk w bieżącym katalogu roboczym.

```
student@linux-basics:~$ mkdir pjatk
student@linux-basics:~$
```

```
student@linux-basics:~$ mkdir pjatk
student@linux-basics:~$ ls -l
total 8
drwxr-xr-x 2 student student 4096 Jul 26 07:55 pjatk
-rw-r--r-- 1 student student    0 Jul 26 07:51 pjatk1
-rw-r--r-- 1 student student    6 Jul 26 07:52 pjatk4
student@linux-basics:~$
```

Zadanie 1.11: Operacje na plikach

`rmdir pjatk` - usuwa pusty katalog o nazwie pjatk.

```
student@linux-basics:~$ rmdir pjatk
student@linux-basics:~$
```

```
student@linux-basics:~$ rmdir pjatk
student@linux-basics:~$ ls -l
total 4
-rw-r--r-- 1 student student 0 Jul 26 07:51 pjatk1
-rw-r--r-- 1 student student 6 Jul 26 07:52 pjatk4
student@linux-basics:~$
```

Zadanie 1.12: Wyszukiwanie

Wyszukanie plików o danej nazwie jest zadaniem, które wykonywać będziemy często. Poniższe polecenia pozwolą nam wykonać to zadanie. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.12: Wyszukiwanie

`locate .conf` - wyszukuje i wyświetla ścieżki do plików, których nazwy zawierają .conf, bazując na zaktualizowanej bazie danych plików.

```
student@linux-basics:~$ locate .conf
/etc/.resolv.conf.systemd-resolved.bak
/etc/adduser.conf
/etc/ca-certificates.conf
/etc/debconf.conf
/etc/deluser.conf
/etc/e2scrub.conf
/etc/fuse.conf
/etc/gai.conf
/etc/host.conf
/etc/kernel-img.conf
/etc/ld.so.conf
/etc/ld.so.conf.d
/etc/libaudit.conf
/etc/manpath.config
/etc/mke2fs.conf
/etc/nsswitch.conf
/etc/pam.conf
/etc/reportbug.conf
/etc/resolv.conf
/etc/sudo.conf
/etc/sudo_logsrvd.conf
/etc/sysctl.conf
/etc/ucf.conf
/etc/updatedb.conf
/etc/xattr.conf
/etc/apparmor/parser.conf
/etc/apt/apt.conf.d
/etc/apt/auth.conf.d
```

Zadanie 1.12: Wyszukiwanie

`find / -name '*.conf' 2>/dev/null` - wyszukuje pliki o rozszerzeniu .conf w całym systemie, ignorując komunikaty o błędach dostępu do niektórych katalogów.

```
student@linux-basics:~$ find / -name '*.conf' 2>/dev/null
/var/lib/ucf/cache/:etc:apt:listchanges.conf
/etc/apt/listchanges.conf
/etc/kernel-img.conf
/etc/sudo.conf
/etc/pam.conf
/etc/sudo_logsrvd.conf
/etc/sysctl.conf
/etc/reportbug.conf
/etc/ucf.conf
/etc/selinux/semanage.conf
/etc/fuse.conf
/etc/modules-load.d/modules.conf
/etc/libaudit.conf
/etc/systemd/sleep.conf
/etc/systemd/resolved.conf
/etc/systemd/journald.conf
/etc/systemd/logind.conf
/etc/systemd/networkd.conf
/etc/systemd/timesyncd.conf
/etc/systemd/pstore.conf
/etc/systemd/system.conf
/etc/systemd/system/getty@tty1.service.d/override.conf
/etc/systemd/system/sshd-keygen@.service.d/disable-sshd-keygen-if-cloud-init-active.conf
/etc/systemd/user.conf
/etc/ca-certificates.conf
/etc/ld.so.conf
/etc/mke2fs.conf
```


Zadanie 1.13: Grep

Wyszukanie plików mamy już opanowane. Co jednak, gdy nie znamy nazwy, a wiemy tylko, co dany plik powinien zawierać? Tu z pomocą przyjdzie nam polecenie grep, które teraz poznamy. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.13: Grep

`grep sh /etc/passwd` - wyszukuje w pliku /etc/passwd linie zawierające ciąg znaków sh, co może pomóc w znalezieniu użytkowników z powłoką logowania zawierającą "sh" (np. /bin/sh).

```
student@linux-basics:~$ grep sh /etc/passwd
root:x:0:0:root:/root:/bin/bash
sshd:x:103:65534::/run/sshd:/usr/sbin/nologin
student:x:1001:1001::/home/student:/bin/bash
student@linux-basics:~$
Solved (1/2) Grep: Find all shell users in /etc/passwd!
$
```

Zadanie 1.13: Grep

`grep -v nologin /etc/passwd` - wyświetla linie z pliku /etc/passwd, które nie zawierają ciągu nologin, pomagając w identyfikacji użytkowników, którzy mają przypisaną aktywną powłokę logowania.

```
student@linux-basics:~$ grep -v nologin /etc/passwd
root:x:0:0:root:/root:/bin/bash
sync:x:4:65534:sync:/bin:/bin/sync
student:x:1001:1001::/home/student:/bin/bash
student@linux-basics:~$
Solved (2/2) Grep: Find all users that can login!
$
```

```
 ____   ___  _     _   _ _____ ____  _
/ ___| / _ \| |   | | | | ____|  _ \| |
\___ \| | | | |   | | | |  _| | | | | |
 ___) | |_| | |___| |_| | |___| |_| |_|
|____/ \___/|_____|\___/|_____|____/(_)
```

```
SOLVED Grep!
$
```

Zadanie 1.14: Usługi

Podobnie jak w systemach z rodziny Windows, tak i w dystrybucjach Linuxa niektóre procesy uruchomione są jako tzw. usługi. Dowiedzmy się o nich czegoś więcej. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.14: Usługi

`systemctl status ssh` - wyświetla aktualny stan usługi SSH, w tym informacje o jej statusie, działających procesach oraz ewentualnych błędach lub problemach z jej uruchomieniem.

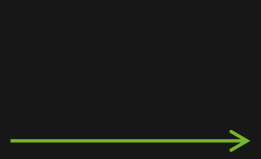
```
student@linux-basics:~$ systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Thu 2024-07-25 14:04:06 UTC; 17h ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 766 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 777 (sshd)
      Tasks: 1 (limit: 2314)
     Memory: 4.5M
        CPU: 191ms
     CGroup: /system.slice/ssh.service
             └─777 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Warning: some journal files were not opened due to insufficient permissions.
student@linux-basics:~$
Solved (1/2) Services: Display ssh service status!
$
```

Zadanie 1.14: Usługi

`systemctl status` - wyświetla status usługi systemowej, ale wymaga podania nazwy usługi, której status ma być sprawdzony. Jeśli użyta bez dodatkowych argumentów, może wyświetlić ogólny stan systemu oraz informacje o działających jednostkach i błędach systemowych.

```
student@linux-basics:~$ systemctl status
● linux-basics
    State: degraded
    Units: 257 loaded (incl. loaded aliases)
     Jobs: 0 queued
   Failed: 1 units
    Since: Thu 2024-07-25 14:04:04 UTC; 18h ago
  systemd: 252.26-1~deb12u2
   CGroup: /
           ├─init.scope
           │ └─1 /sbin/init
           ├─system.slice
           │ ├─autosolver.service
           │ │ └─763 /usr/bin/python3 /root/.autosolver.py
           │ ├─containerd.service
           │ │ └─773 /usr/bin/containerd
           │ ├─cron.service
           │ │ └─640 /usr/sbin/cron -f
           │ ├─dbus.service
           │ │ └─641 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activ>
           │ ├─docker.service
           │ │ └─901 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
           │ ├─exim4.service
           │ │ └─1157 /usr/sbin/exim4 -bd -q30m
           │ ├─lb.service
           │ │ └─765 /usr/bin/python3 /root/.lb.py
           │ ├─open-vm-tools.service
           │ │ └─531 /usr/bin/vmtoolsd
           │ ├─ssh.service
           │ │ └─777 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
           │ ├─systemd-journald.service
           │ │ └─357 /lib/systemd/systemd-journald
           │ ├─systemd-logind.service
           │ │ └─643 /lib/systemd/systemd-logind
           │ ├─systemd-networkd.service
           │ │ └─511 /lib/systemd/systemd-networkd
```



Zadanie 1.15: Zadania Cron

Ostatnie zagadnienie tej części to tzw. Cronjobs. Czasami chcemy, aby nasz system uruchomił jakieś polecenie samodzielnie. Codziennie? Co godzinę? Co minutę? Cron jest odpowiedzią! Dowiedzmy się o nich czegoś więcej. Celem zadania jest uruchomienie tych komend na maszynie z ćwiczeniami.

Zadanie 1.15: Zadania Cron

`cat /etc/crontab` - wyświetla zawartość pliku /etc/crontab, który zawiera globalne zadania cron, definiujące harmonogram uruchamiania zadań systemowych na poziomie całego systemu.

```
student@linux-basics:~$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.daily; }
47 6    * * 7   root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.weekly; }
52 6    1 * *   root    test -x /usr/sbin/anacron || { cd / && run-parts --report /etc/cron.monthly; }
#
student@linux-basics:~$
Solved (1/2) Cronjobs: Display /etc/crontab contents!
$
```

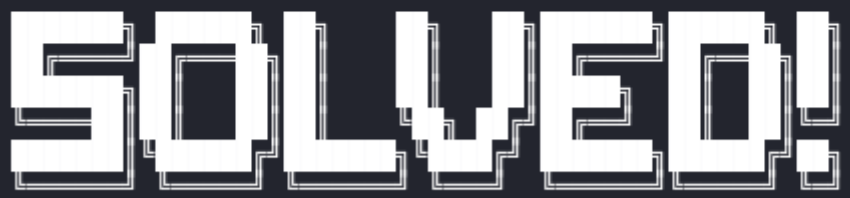
Zadanie 1.15: Zadania Cron

`crontab -e` - otwiera edytor tekstu, umożliwiając edytowanie osobistych zadań cron dla bieżącego użytkownika, które będą uruchamiane zgodnie z określonym harmonogramem.

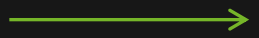
```
student@linux-basics:~$ crontab -e
no crontab for student - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny

Choose 1-3 [1]:
Solved (2/2) Cronjobs: Edit your private crontab!
$
```




```
SOLVED Cronjobs!
$
```



Referencje

https://man.linux.pl/
polski manual dla systemu Linux

https://www.man7.org/linux/man-pages/man1/man.1.html
manual Linux

https://ubuntu.com/tutorials/command-line-for-beginners#1-overview
podstawowe komendy Linux (tu w wersji na dystrybucję Ubuntu, jednak są one takie same w innych dystrybucjach)

https://www.onworks.net/programs/ubuntu-emulator-online
emulator Linuxa

https://www.geeksforgeeks.org/difference-between-linux-and-windows/
porównanie Linuxa z Windowsem

Koniec ćwiczeń