Procesy

Podczas uruchamiania każdy proces w systemie ma nadawany unikalny numer PID (ang. *Process IDentifier).* Wszystkie procesy w systemie Linux są procesami potomnymi procesu *init,* który ma identyfikator 1. Jest on tworzony podczas startu systemu operacyjnego.

Do każdego procesu przypisany jest użytkownik, który go uruchomił. Na potrzeby usług takich jak serwer WWW czy serwer pocztowy tworzone są specjalne konta, które pozwalają uruchomić wybraną usługę. Usługi w systemach Linux zwane są demonami (ang. *daemori).*

DEFINICJA

W systemie Linux wszystkie uruchomione programy noszą miano procesu. Jądro systemu steruje procesami i zarządza czasem procesora, przydzielając go kolejnym procesom (wielozadaniowość). System wykonuje dany proces przez pewien czas, a następnie udostępnia procesor kolejnemu procesowi. Procesy mogą przyjmować następujące stany:

- Działający — aktualnie wykonujący jakąś operację.
- Uśpiony — proces czeka na jakieś zdarzenie systemowe, na przykład odczyt danych z dysku.
- Gotowy do wykonania — proces czeka na przydzielenie mu czasu procesora.
- Zombie — proces zakończył działanie, czeka na zakończenie go przez proces macierzysty.

Jako system wielozadaniowy Linux pozwala uruchamiać zadania w tle \v trybie tekstowym. Standardowo programy uruchamiane są na pierwszym planie (następuje interakcja z terminalem). Program, który pracuje w tle, nie ma interakcji z terminalem. Aby przenieść uruchomiony program do pracy w tle, należy użyć kombinacji klawiszy *Ctrl+Z.* Kombinacja *Ctrl+C* kończy bieżący proces uruchomiony na pierwszym planie.

Aby zakończyć działające procesy z poziomu systemu operacyjnego — bez konieczności przełączania się między nimi — należy użyć komendy kil l PID, gdzie ?rr jest identyfikatorem procesu. Zamknięcie wszystkich procesów danego typu (na podstawie ich nazwy, a nie identyfikatora) umożliwia polecenie killaii *nazwa_procesu.*

Aby wyświetlić procesy uruchomione na serwerze, można skorzystać z polecenia p s, które bez parametrów wyświetla programy wybranego użytkownika. Opcje tej komendy, które pozwolą wyświetlić więcej informacji, to:

- -A — wyświetla wszystkie procesy, także procesy innych użytkowników (rysunek 13.2);
- -a — wyświetla wszystkie procesy uruchomione w aktualnym oknie terminala;
- -l — wyświetla długą listę informacji o procesach (w tym czas utworzenia i pra wa dostępu);
- -m — wyświetla informację o pamięci;

Rysunek 13.2. Wynik działania polecenia ps -A

-u — wyświetla informację o procesach wybranego

Źródło: P. Brensel: Systemy i sieci komputerowe

Aby uruchomić program, który rozpocznie przetwarzanie w tle (np. kompilację kodu źródłowego), na końcu polecenia uruchamiającego należy wpisać znak &, na przykład:

```
gcc program.c &
```

To polecenie uruchomi w tle kompilację kodu źródłowego zapisanego w pliku *program.c*. Użytkownik ujrzy na monitorze informację o numerze uruchomionego procesu (PID). Po zakończeniu kompilacji system wyświetli komunikat o zamknięciu procesu.

Aby sprawdzić zadania wykonywane w tle, należy skorzystać z polecenia jobs (rysunek 13.3), które wyświetla numer *zadania* w tle, nazwę procesu, jego status — działający (ang. *running),* zatrzymany (ang. *stopped)* lub zakończony (ang. *done).*

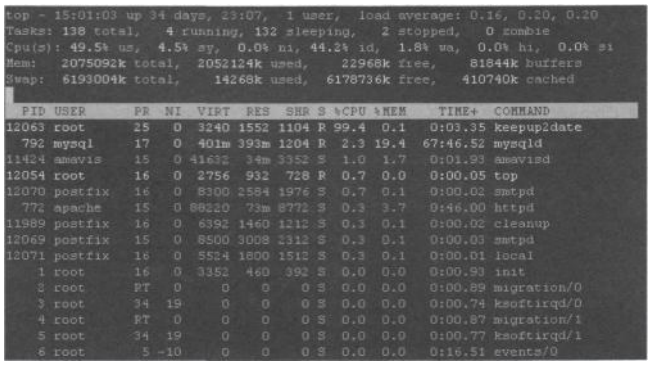
Rysunek 13.3. Wynik działania polecenia jobs

Aby umieścić wybrane polecenie ponownie na pierwszym planie, należy użyć polecenia f g, podając jako parametr numer *zadania* w tle, wyświetlany przez polecenie jobs. Polecenie f g bez parametru przeniesie na pierwszy plan zadanie, które zostało umieszczone w tle jako ostatnie.

Aby zmienić status *zadania* wykonywanego w tle, należy użyć komendy bg z numerem *zadania.* Polecenie to powoduje, że status *zadania* w tle zmieni się z zatrzymanego na działający.

Do zbadania wydajności pracy komputera służy w systemie Linux polecenie top (rysunek 13.4). Wyświetla ono w czasie rzeczywistym listę *zadań* najbardziej obciążających procesor oraz podsumowanie pracy serwera (średnie obciążenie procesora,

DEFINICJA



Rysunek 13.4. Wynik działania polecenia top

Z każdym procesem w systemie Linux związane jest pojęcie strumienia, czyli danych przekazywanych do programu i danych, które generuje dany proces. Zwykle występują trzy strumienie:

- *stdin* — strumień wejściowy domyślnie związany z klawiaturą, z której wpro wadzane są dane;
- *stdout* — strumień wyjściowy domyślnie związany z ekranem, na którym wy świetlane są wyniki pracy programu;
- *stderr* — strumień wyjściowy domyślnie związany z ekranem, na którym wy świetlane są btędy generowane przez dany proces.

Zależności pomiędzy strumieniami przedstawione zostały na rysunku 13.5.

Źródło: P. Brensel: Systemy i sieci komputerowe

Rysunek 13.5.
Przepływy strumieni danych

Jedną z cech systemu Linux jest możliwość przekierowania strumieni do plików. Operatory przekierowań to:

- > — przekierowuje strumień wyjściowy do zwykłego pliku podanego jako pa
  rametr. Jeśli plik nie istnieje, zostanie utworzony, jeśli istnieje, cała zawartość zostanie zastąpiona.
- » — przekierowuje strumień wyjściowy do pliku, dopisując dane na końcu pliku.
- < — przekierowuje na strumień wejściowy dane zawarte we wskazanym pliku.

Aby przekierować wyniki pracy wybranego programu do pliku, należy użyć konstrukcji *nazwa_polecenia [parametry] > plik_z_wynikami,* na przykład:

```
Is  -la  >  moje_dane.txt
```

Polecenie to zapisze w pliku *moje_dane.txt* zawartość bieżącego katalogu (wynik działania polecenia Is -la). Użycie konstrukcji *nazwa_polecenia \pararr.e-ry] clik_z_wynikami*$_y$ na przykład:

```
Is  -la  »  moje_dane.txt
```

spowoduje, że wyniki działania programu zostaną dopisane na końcu wybranego pliku.

W celu przekierowania strumienia we j ściowego używa się konstrukc j i *na z wa_poi e cen i a parametry] < plik_z_danymi,* **na przykład:**

```
mаił uczenl999@wp.pl < informacja.txt
```

Powyższa linijka spowoduje wysłanie zawartości pliku *informacja.txt* na adres pocztowy *uczenl999@wp.pl.*

Kolejnym przykładem funkcjonalności rozwiązań związanych z pracą w trybie tekstowym są potoki danych. Są to strumienie wyjściowe jednego procesu przekazywar.i jako dane wejściowe do innego procesu. W przypadku potoków operatorem póz łającym na przekazanie jest symbol |. Obowiązuje składnia: *program__pierwszy program drugi,* **na przykład:**

```
Is  -la  |  grep  uczeń
```

W przytoczonym przykładzie wyniki działania polecenia l s zostają przekazane ni wejście dla programu grep, który ma za zadanie wypisanie tylko tych linii, w których znajduje się słowo *uczeń.* Danymi wejściowymi programu drugiego (grep) jest lista plików będąca wynikiem działania programu pierwszego (ls).

Podczas przekazywania potoków między procesami bardzo często używana jest wspomniana komenda grep. Służy ona do wyświetlania tylko tych linii, które pasują (lub nie pasują) do określonego wzorca. Uproszczona składnia wygląda w sposób następujący: grep [-v] *wzorzec [plik],* gdzie

- -v — oznacza opcję negacji wzorca;
- *wzorzec* — oznacza treść do wyszukania;
- *plik* — oznacza plik, którego zawartość ma być sprawdzona (gdy nie używamy potoków).

Wzorce tworzone są z wykorzystaniem wyrażeń regularnych. W tabeli 13.2 przedstawiono znaki specjalne, pozwalające na tworzenie dowolnych wyrażeń.

Tabela 13.2. Znaki specjalne wykorzystywane w wyrażeniach regularnych

Znak	Opis
	Dopasuj dowolny znak.

Źródło: P. Brensel: Systemy i sieci komputerowe

Dopasuj poprzedzające wyrażenie do końca wiersza.

Dopasuj występujące po operatorze wyrażenie do początku wiersza.

Dopasuj zero lub więcej wystąpień znaku poprzedzającego operator.

Dopasuj dowolny znak ujęty w nawiasy, na przykład [abc012].

Dopasuj dowolny znak z przedziału, na przykład [O - 9] — wszystkie cyfry; [a- z] — wszystkie małe litery; [O - 9a- zA-Z] — wszystkie litery i cyfry.

[^] Dopasuj znak, który nie znajduje się w nawiasach.

Aby lepiej zrozumieć mechanizm tworzenia wyrażeń regularnych, w tabeli 13.3 przedstawione zostały przykłady zastosowań.

Tabela 13.3. Wykorzystanie wyrażeń regularnych w programie grep

Polecenie	**Opis**
`grep 'Ala^1 plik grep`	Wypisze linie zawierające wyraz Ala.
`'A.a^1 plik`	Wypisze linie zawierające wyrazy takie jak Ala. A: . A+a itp.
`grep 'A[lg]a^? plik`	Wypisze linie zawierające wyrazy Ala i Aga.
`grep ^IAAla' plik`	Wypisze linie rozpoczynające się od słowa Ala.
`grep 'Go*gle^f plik`	Wypisze linie zawierające wyrazy rozpoczynające się na literę G, kończące się na g l e, które między nimi zawierają dowolną liczbę wystąpień litery o.

Innym poleceniem wykorzystywanym podczas przekazywania potoków **jest morę,** które wyświetla dane z podziałem na strony. Gdy użytkownik zapełni ekran danymi, polecenie to pozwala kontynuować wyświetlanie. Wystarczy, że użytkownik naciśnie klawisz.

Ćwiczenia:

W tabeli przedstaw różnicę między wyświetlaniem procesów w systemie MS Microsoft i Linux zwracając uwagę na dostęp do procesów, ich rodzaje i sposób zakończenia. Możesz użyć aplikacji Word, Excel lub PowerPoint. Gotowy plik o nazwie procesy_nazwisko umieszczasz w folderze Wspolny/KlasaIII na dysku wspólnym. Podobieństwa i różnice zilustruj screenami.