Linux: Procesy

Systemy Operacyjne

Mateusz Hołenko

26 marca 2013

Plan zajęć

- Co to jest sygnał?
- Polecenia systemowe
- Sygnały
- Zadania

Co to jest proces?

Co to jest proces?

Abstrakcja opisująca aplikację w trakcie **wykonywania**.

Proces obejmuje sobą:

- kod aplikacji
- dane
- stos wykonania
- status wykonywania
- przydzielone zasoby
- uprawnienia

Co to jest proces?

Abstrakcja opisująca aplikację w trakcie **wykonywania**.

Proces obejmuje sobą:

- kod aplikacji
- dane
- stos wykonania
- status wykonywania
- przydzielone zasoby
- uprawnienia

Hierarchia procesów

- procesy tworzą hierarchię
  - proces główny nazywa się **`init`**
- proces tworzący inne procesy staje się dla nich rodzicem
- procesy wykonują się w separacji od siebie i mogą kończyć się w dowolnej kolejności
- procesy **sieroty** (ang. *orphan*) - procesy, których proces macierzysty się zakończył
  - ich procesem macierzystym staje się **`init`**
- procesy **zombie** - proces się zakończył, ale nie przekazał jeszcze swojego statusu zakończenia
  - nie zajmuje procesora ani innych zasobów
  - występuje w tablicy procesów

Hierarchia procesów

- procesy tworzą hierarchię
  - proces główny nazywa się **`init`**
- proces tworzący inne procesy staje się dla nich rodzicem
- procesy wykonują się w separacji od siebie i mogą kończyć się w dowolnej kolejności
- procesy **sieroty** (ang. *orphan*) - procesy, których proces macierzysty się zakończył
  - ich procesem macierzystym staje się **`init`**
- procesy **zombie** - proces się zakończył, ale nie przekazał jeszcze swojego statusu zakończenia
  - nie zajmuje procesora ani innych zasobów
  - występuje w tablicy procesów

Hierarchia procesów

- procesy tworzą hierarchię
  - proces główny nazywa się **`init`**
- proces tworzący inne procesy staje się dla nich rodzicem
- procesy wykonują się w separacji od siebie i mogą kończyć się w dowolnej kolejności
- procesy **sieroty** (ang. *orphan*) - procesy, których proces macierzysty się zakończył
  - ich procesem macierzystym staje się **`init`**
- procesy **zombie** - proces się zakończył, ale nie przekazał jeszcze swojego statusu zakończenia
  - nie zajmuje procesora ani innych zasobów
  - występuje w tablicy procesów

Hierarchia procesów

- procesy tworzą hierarchię
  - proces główny nazywa się **`init`**
- proces tworzący inne procesy staje się dla nich rodzicem
- procesy wykonują się w separacji od siebie i mogą kończyć się w dowolnej kolejności
- procesy **sieroty** (ang. *orphan*) - procesy, których proces macierzysty się zakończył
  - ich procesem macierzystym staje się **`init`**
- procesy **zombie** - proces się zakończył, ale nie przekazał jeszcze swojego statusu zakończenia
  - nie zajmuje procesora ani innych zasobów
  - występuje w tablicy procesów

Hierarchia procesów

- procesy tworzą hierarchię
  - proces główny nazywa się **`init`**
- proces tworzący inne procesy staje się dla nich rodzicem
- procesy wykonują się w separacji od siebie i mogą kończyć się w dowolnej kolejności
- procesy **sieroty** (ang. *orphan*) - procesy, których proces macierzysty się zakończył
  - ich procesem macierzystym staje się **`init`**
- procesy **zombie** - proces się zakończył, ale nie przekazał jeszcze swojego statusu zakończenia
  - nie zajmuje procesora ani innych zasobów
  - występuje w tablicy procesów

Polecenia systemowe

Przeglądanie listy procesów

Wyświetlanie listy procesów w systemie:

```
houen@kashyyyk:~$ ps
```

- **a** wyświetl wszystkie procesy mające terminal
- **x** wyświetl wszystkie procesy bieżącego użytkownika
- **u** wyświetl w formie szczegółowej
- **r** wyświetl procesy uruchomione

Przeglądanie listy procesów II

Wyświetlanie zawartości katalogu w postaci drzewa w *ascii art*:

```
houen@kashyyyk:~$ pstree
```

Wyświetlanie listy procesów w postaci interaktywnej:

```
houen@kashyyyk:~$ top
```

Skróty klawiszowe:

- **k** wyślij sygnał do procesu
- **u** wyświetlaj procesy danego użytkownika
- **h** wyświetl pomoc
- **f** wybierz kolumny, sortuj
- **H** pokaż wątki

Przeglądanie listy procesów II

Wyświetlanie zawartości katalogu w postaci drzewa w *ascii art*:

```
houen@kashyyyk:~$ pstree
```

Wyświetlanie listy procesów w postaci interaktywnej:

```
houen@kashyyyk:~$ top
```

Skróty klawiszowe:

- **k** wyślij sygnał do procesu
- **u** wyświetlaj procesy danego użytkownika
- **h** wyświetl pomoc
- **f** wybierz kolumny, sortuj
- **H** pokaż wątki

Przeglądanie listy procesów III

Jeszcze lepsze przeglądanie listy procesów w postaci interaktywnej:

```
houen@kashyyyk:~$ htop
Klawiszologia:
                F1 pomoc
```

Sygnały

Sygnały są mechanizmem komunikacji między procesami udostępnianym przez jądro systemu Linux. Ze względu na swój specyficzny charakter określane są często mianem **przerwań programowych**. Stosuje się je do sygnalizowania wystąpienia sytuacji wyjątkowych. Obsługiwane są **asynchronicznie**.

Sygnały generowane mogą być przez:

- jądro systemy operacyjnego
- inne procesy w systemie
- użytkownika systemu

Sygnały są mechanizmem komunikacji między procesami udostępnianym przez jądro systemu Linux. Ze względu na swój specyficzny charakter określane są często mianem **przerwań programowych**. Stosuje się je do sygnalizowania wystąpienia sytuacji wyjątkowych. Obsługiwane są **asynchronicznie**.

Sygnały generowane mogą być przez:

- jądro systemy operacyjnego
- inne procesy w systemie
- użytkownika systemu

O sygnałach

Generowanie sygnału za pomocą klawiatury:

```
houen@kashyyyk:~$ <CTRL-C>
houen@kashyyyk:~$ <CTRL-D>
houen@kashyyyk:~$ <CTRL-\>
```

Praca samodzielna

```
houen@kashyyyk:~$ kill -l
houen@kashyyyk:~$ man 7 signal
```

O sygnałach

Generowanie sygnału za pomocą klawiatury:

```
houen@kashyyyk:~$ <CTRL-C>
houen@kashyyyk:~$ <CTRL-D>
houen@kashyyyk:~$ <CTRL-\>
```

Praca samodzielna

```
houen@kashyyyk:~$ kill -l
houen@kashyyyk:~$ man 7 signal
```

Wysyłanie sygnałów

Wysyłanie sygnałów do procesów:

```
houen@kashyyyk:~$ kill 46387
houen@kashyyyk:~$ killall firefox
```

-l wypisuje listę sygnałów

-N wyślij sygnał o numerze N

Domyślnie wysyłany jest sygnał TERM o numerze 15. Kombinacja klawiszy <CTRL–C> domyślnie wysyła ten sam sygnał do aktywnego działającego procesu. Sygnał ten może zostać zignorowany!

Wysyłanie sygnałów

Wysyłanie sygnałów do procesów:

```
houen@kashyyyk:~$ kill 46387
houen@kashyyyk:~$ killall firefox
```

`-l` wypisuje listę sygnałów
`-N` wyślij sygnał o numerze `N`

Domyślnie wysyłany jest sygnał `TERM` o numerze 15. Kombinacja klawiszy <CTRL–C> domyślnie wysyła ten sam sygnał do aktywnego działającego procesu. Sygnał ten może zostać zignorowany!

Umilanie procesów

Procesy można uruchomić z innym niż domyślnie priorytetem:

```
houen@kashyyyk:~$ nice -n 13 vim
```

Można też zmieniać priorytet dla działających procesów:

```
houen@kashyyyk:~$ renice +1 34223
```

Zwiększanie priorytetu wymaga uprawnień administratora systemu! Zmniejszać priorytet może zaś każdy z użytkowników (oczywiście tylko w kontekście swoich procesów).

Umilanie procesów

Procesy można uruchomić z innym niż domyślnie priorytetem:

```
houen@kashyyyk:~$ nice -n 13 vim
```

Można też zmieniać priorytet dla działających procesów:

```
houen@kashyyyk:~$ renice +1 34223
```

Zwiększanie priorytetu wymaga uprawnień administratora systemu! Zmniejszać priorytet może zaś każdy z użytkowników (oczywiście tylko w kontekście swoich procesów).

Umilanie procesów

Procesy można uruchomić z innym niż domyślnie priorytetem:

```
houen@kashyyyk:~$ nice -n 13 vim
```

Można też zmieniać priorytet dla działających procesów:

```
houen@kashyyyk:~$ renice +1 34223
```

Zwiększanie priorytetu wymaga uprawnień administratora systemu! Zmniejszać priorytet może zaś każdy z użytkowników (oczywiście tylko w kontekście swoich procesów).

Uruchamianie wielu procesów / Zadania

Zadania

- mechanizm pozwala na zawieszanie oraz wznawianie działania procesów.
- w celu zawieszenia aktualnego procesu użyć należy klawiszy: **<CTRL-Z>**

Kontrolować stan procesu można również przy użyciu procesów:

SIGSTOP zawieszanie procesu

SIGCONT uaktywnianie procesu

Zadania

- mechanizm pozwala na zawieszanie oraz wznawianie działania procesów.
- w celu zawieszenia aktualnego procesu użyć należy klawiszy: **<CTRL-Z>**

Kontrolować stan procesu można również przy użyciu procesów:

SIGSTOP zawieszanie procesu

SIGCONT uaktywnianie procesu

Kontrola zadań

Listowanie aktywnych zadań:

```
houen@kashyyyk:~$ jobs
```

Uaktywnianie procesu:

```
houen@kashyyyk:~$ fg vim
houen@kashyyyk:~$ fg 1
```

Uaktywnianie procesu w tle:

```
houen@kashyyyk:~$ bg vim
houen@kashyyyk:~$ bg 1
```

Kontrola zadań

Listowanie aktywnych zadań:

```
houen@kashyyyk:~$ jobs
```

Uaktywnianie procesu:

```
houen@kashyyyk:~$ fg vim
houen@kashyyyk:~$ fg 1
```

Uaktywnianie procesu w tle:

```
houen@kashyyyk:~$ bg vim
houen@kashyyyk:~$ bg 1
```

Kontrola zadań

Listowanie aktywnych zadań:

```
houen@kashyyyk:~$ jobs
```

Uaktywnianie procesu:

```
houen@kashyyyk:~$ fg vim
houen@kashyyyk:~$ fg 1
```

Uaktywnianie procesu w tle:

```
houen@kashyyyk:~$ bg vim
houen@kashyyyk:~$ bg 1
```

Kontrola uruchamiania procesów

Uruchamianie programu w tle:

```
houen@kashyyyk:~$ gvim &
```

Sekwencyjne uruchamianie procesów:

```
houen@kashyyyk:~$ appa; appb; appc
```

Warunkowe uruchamianie procesów:

```
houen@kashyyyk:~$ appa || appb
houen@kashyyyk:~$ appa && appb
houen@kashyyyk:~$ (appa || appc) && appb
```

Kontrola uruchamiania procesów

Uruchamianie programu w tle:

```
houen@kashyyyk:~$ gvim &
```

Sekwencyjne uruchamianie procesów:

```
houen@kashyyyk:~$ appa; appb; appc
```

Warunkowe uruchamianie procesów:

```
houen@kashyyyk:~$ appa || appb
houen@kashyyyk:~$ appa && appb
houen@kashyyyk:~$ (appa || appc) && appb
```

Kontrola uruchamiania procesów

Uruchamianie programu w tle:

```
houen@kashyyyk:~$ gvim &
```

Sekwencyjne uruchamianie procesów:

```
houen@kashyyyk:~$ appa; appb; appc
```

Warunkowe uruchamianie procesów:

```
houen@kashyyyk:~$ appa || appb
houen@kashyyyk:~$ appa && appb
houen@kashyyyk:~$ (appa || appc) && appb
```