Практична робота № 15. Знайомство з процесами. Контроль ресурсів та планування задач

Поняття процесу

Операційна система Linux є багатозадачною (мультизадачною). Це означає, що одночасно в системі може бути присутня множина процесів, кожному з яких доступна певна кількість процесорного часу. Для користувача створюється «ілюзія» одночасного виконання процесів.

Процес – виконувана програма з її даними і контекстом.

Кожен процес має унікальний у будь-який момент часу іденти-фікатор у системі – ***PID***. Перший підготовлений до запуску в системі процес ***init*** має ***pid = 1***.

Для опису процесів в операційній системі є список структур – дескрипторів, що містять інформацію про ідентифікатор процесу, пріоритет, стан процесу, інформацію про належність користувачеві і групі, займаних процесом ресурсах та ін.

Кожен процес у системі Linux запускається будь-яким іншим процесом. Запускаючий процес – батьківський, новий процес – дочірній. Процеси, які виконують одну задачу, об'єднуються в групи, які мають власний ідентифікатор. Процес всередині групи, ідентифікатор якого збігається з ідентифікатором групи процесів, вважається лідером групи процесів.

Усі запущені процеси умовно (залежно від виконуваної ними функції) можна розділити на три типи:

1. **Системні процеси** є частиною ядра і завжди розташовані в оперативній пам'яті. Вони часто не мають відповідних їм програм у вигляді виконуваних файлів і завжди запускаються особливим чином під час завантаження ядра системи.

2. **Процеси-демони** – це неінтерактивні процеси, які виконуються у фоновому режимі.

3. До **прикладних** відносяться всі інші процеси, що виконуються в системі.

Інтерактивні процеси пов'язані з визначеним терміналом і через нього взаємодіють з користувачем. Фонові процеси виконуються незалежно від користувача і (псевдо) паралельно.

Кожен процес в операційній системі Linux може перебувати в одному з чотирьох станів: **працездатний**, **сплячий** (або очікування), **зупинений** і **той, що завершився**.

Додаткові утиліти

Для отримання інформації про запущені процеси часто використовується команда ***ps*.**

```
$ ps
```

Виведення запущеної без аргументів команди містить: інформацію про процеси поточного користувача і асоційовані з поточним терміналом, процесорний час, зайнятий цим процесом, і ім'я файлу. Управляти форматом виведення можна за допомогою додаткових опцій.

Таблиця 1.

Приклади опцій команди *ps*

Ключ	*Опис*
-a	видати всі процеси системи, включаючи лідерів сеансів
-d	видати всі процеси системи, виключаючи лідерів сеансів
-e	видати всі процеси системи
-x	видати процеси системи, що не мають контрольного термінала
-o	визначає формат виведення у вигляді списку полів, розділених символом «,»
-u	видати процеси, що належать зазначеному користувачеві

Наприклад, можна отримати вибіркову інформацію про всі процеси в системі:

```
$ ps -eo s,pid,tty,command
```

Альтернативним способом дізнатися про стан процесів в реальному часі є використання команди ***top***. Висновком команди можна керувати за допомогою спеціальних комбінацій клавіш. Довідкову інформацію можна отримати, натиснувши клавішу «***h***».

Щоб отримати інформацію про запущені в системі процеси у вигляді дерева, можна використовувати утиліту ***pstree*.**

Управління процесами

Щоб запустити програму, достатньо ввести її ім'я в командному рядку і натиснути «Enter». Однак не всі команди запускають єдиний процес.

Інтерактивні процеси, запущені в терміналі, займають термінальну сесію, і оболонка не виводить користувачеві рядок запрошення до тих пір, поки програма не завершиться.

Роботу деяких запущених у терміналі програм можна перервати за допомогою поєднання клавіш «**Ctrl + C**» у вікні термінала. У цей момент програмі надсилається сигнал **INT (Interrupt).**

Щоб запустити програму у фоновому режимі, необхідно завершити команду символом амперсанд «**&**». Після цього в термінал виводиться

інформація про запущений процес, включаючи номер завдання термінал, і запрошення користувачеві на введення нової команди.

```
$ top &
```

Використовуючи команду ***jobs***, ми можемо отримати список завдань, які запущені через термінал.

```
$ jobs
```

Щоб повернути запущений у фоні процес на перший план, використовується команда ***fg*** із зазначенням номера завдання у списку завдань.

```
$ fg %2
```

Якщо ми хочемо перевести процес у стан «**зупинений**», використовується поєднання клавіш «**Ctrl + z**». У цей момент програмі надсилається сигнал **TSTP (Terminal Stop)**.

Після цього ми можемо або перемістити завдання на перший план командою ***fg***, або продовжити його виконання у фоновому режимі командою ***bg***.

```
$ bg %2
```

Ще одним способом управляти виконанням процесів є використання утиліти ***kill***. Ця команда дозволяє послати певний сигнал процесу. Можливе завершення процесу як за іменем, так і за номером завдання або за ідентифікатором процесу (PID).

```
$ kill -SIGINT 124672
```

Отримати список сигналів можна за допомогою опції -l.

```
$ kill -l
```

Існує більше двадцяти різних сигналів. Перелічимо основні з них:

– **SIGCHLD** – сигнал про завершення дочірнього процесу;

– **SIGHUP** – сигнал звільнення лінії. Посилається всім процесам, підключеним до керуючого терміналу у разі відключення терміналу. Багато демонів під час отримання такого сигналу заново переглядають файли конфігурації і перезапускаються;

– **SIGINT** – сигнал посилається всім процесам сеансу, що пов'язані з терміналом у разі натискання користувачем клавіші переривання (CTRL-C);

– **SIGTERM, SIGKILL** – сигнали призводять до негайного припинення роботи процесу, що отримав сигнал. На відміну від сигналу SIGTERM, процес не може блокувати і перехоплювати сигнал SIGKILL;

– **SIGSEGV** – сигнал посилається процесу, якщо той намагається звернутися за неправильною адресою пам'яті;

– **SIGSTOP** – сигнал приводить до зупинки процесу. Для відправки сигналу SIGSTOP активного процесу поточного термінала можна скористатися комбінацією клавіш (CTRL-Z);
– **SIGCONT** – сигнал відновлює роботу зупиненого процесу;
– **SIGUSR1, SIGUSR2** – сигнали, що визначаються користувачем.

Послати сигнал декільком процесам можна за допомогою команди ***killall***.

```
$ killall gedit
```

Наведена вище команда завершить всі процеси поточного користувача з ім'ям **gedit**. За замовчуванням команда відправляє сигнал **TERM** (software termination signal).

Отримання інформації про систему

Однією з утиліт для відстеження інформації про продуктивність є ***vmstat***. Виведенням команди є звіт про стан системи, що отримується із заданим інтервалом часу. Наприклад, отримати звіт, що складається з 8 рядків, котрі містять статистику, зібрану з інтервалом у 2 секунди, можна командою:

```
$ vmstat 2 8
```

Виведення містить інформацію про готові до виконання і сплячі процеси, використання оперативної пам'яті, підкачки, дискові операції, статистику використання центрального процесора. Детальніше про формат виведення можна дізнатися на сторінках довідкового керівництва.

Інформацію про середню завантаженість системи за період 5 с, 10 с і 15 с, а також час безперервної роботи системи можна отримати за допомогою утиліти ***uptime***.

```
$ uptime
```

Отримати загальну інформацію про наявну фізичну і віртуальну пам'ять у системі можна з допомогу команди ***free***. Крім цього, виведення містить інформацію про розмір буферів системи.

```
$ free -h
```

Отримати інформацію про використання дискового простору в системі можна за допомогою утиліти ***df***. Виведення команди містить дані за усіма змонтованими на поточний момент файловими системами із зазначенням відсотка використаного простору і точки монтування кожної з них. Опція ***-h*** призводить форматування виведення до зручного для користувача вигляду.

```
$ df -h
```

Щоб дізнатися розмір не всього обсягу в цілому, а будь-якої директорії, можна застосувати утиліту ***du***. Початкове виведення команди містить розміри всіх вкладених директорій, є можливість управляти глибиною вкладеності. Також, як і у випадку з командою ***df***, може застосовуватися опція ***-h***.

```
$ du –h
```

Планування повторюваних завдань

Для більшості завдань, що стоять перед системними адміністраторами, характерне періодичне виконання. Для зручності складання розкладу для користувача завдань в операційній системі є служба **cron**.

Завдання планувальника **cron** за рядками перераховані в спеціальному **crontab**-файлі. Записи у файлі мають такий вигляд:

10 15 * * * /home/user/my_script.sh

Де п'ять полів, розділених проміжками, означають числові представлення хвилин, годин, днів місяця, місяців на рік і дня тижня відповідно. Символ «*****» відповідає будь-якому значенню. Символ «**/**» служить для вказівки додаткової періодичності завдання. Наприклад, «***/3**» в першому полі означає «кожні 3 хвилини». У наведеному вище прикладі для користувача скрипт **my_script.sh** буде виконуватися кожен день о 15 годині і 10 хвилинах.

Кожен користувач може мати свій файл **crontab**, щоб повідомити системі ім'я файлу, необхідно виконати команду:

```
$ crontab filename
```

Виведення наявних завдань виконується цією утилітою з опцією ***-l***. Очищення списку завдань виконується командою ***crontab*** з опцією ***-r***. Редагування наявного файлу завдань можливо текстовим редактором з використанням опції ***-e***, наприклад:

```
$ crontab -e
```

Після додавання файлу завдання або зміни наявного файлу відбувається перевірка синтаксису.

Завдання

1. Ознайомтеся з роботою команд, які наведені у лабораторній роботі, за допомогою довідкового керівництва.
2. Створіть файл ***proc1***, що містить список процесів користувача **root**, відсортоване за ідентифікатором батьківського процесу. Використовуйте команду ***ps*** і вивчені раніше команди.
3. Отримайте інформацію про процеси вашого користувача, що мають статус «працездатний».

4. Додайте до файлу ***proc1*** відомості про процес, що в цей момент споживає більшість ресурсів центрального процесора.
5. Запустіть утиліту ***top***. Вивчіть вміст інформаційних полів, що надаються утилітою. Отримайте інформацію про ступінь використання ресурсів системи, кількості користувачів, часу роботи системи.
6. Отримайте список завдань поточної сесії термінала.
7. Використовуючи команди ***fg*** і ***bg*** і поєднання клавіш «***Ctrl+Z***» і «***Ctrl+C***» навчитеся переміщувати завдання з фону на перший план і навпаки.
8. Отримайте список сигналів для команди ***kill***. Завершіть запущені процеси за допомогою сигналів ***SIGKILL*** і ***SIGTERM***.
9. Протягом 30 с з інтервалом в 3 с збирайте статистику про використання ресурсів системи. Отримайте інформацію про середню завантаженість процесора протягом останніх 15 с.
10. Опишіть поточний стан сторінок пам'яті та твердих дисків, доступних у вашій системі.
11. Отримайте інформацію про розмір вашого домашнього каталогу, отримайте список 3 найбільших каталогів у вашій домашній директорії за допомогою команд, вивчених раніше.
12. Створіть завдання, згідно з яким:
    – кожну хвилину у файл **~/memory/stat** буде додаватися інформація про поточний стан пам'яті, без урахування розміру підкачки і заголовка;
    – кожні 3 хвилини файл ***~/memory/stat*** буде упаковуватися в архів.
13. Після виконання роботи видаліть всі записи з вашого crontab-файлу.

Контрольні питання

1. Які способи отримання інформації про процеси в системі ви знаєте?
2. Як можна керувати виведенням утиліти ***top***?
3. Які сигнали відправляються поєднаннями клавіш «***Ctrl+Z***» і «***Ctrl+C***»?
4. Які типи процесів ви знаєте?
5. Як отримати інформацію про стан пам'яті?
6. Як отримати інформацію про доступний дисковий простір?
7. Як відбувається робота з файлами завдань планувальника **cron**?