ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 6. «ПРОЦЕССЫ. ДОСТУП ПРОЦЕССОВ К ФАЙЛОВОЙ СИСТЕМЕ»

Теоретическая часть

Под процессами во всех Unix-подобных системах подразумевается любая независимо выполняющаяся программа со всеми используемыми ей ресурсами. Процесс тесно связан с такими понятиями, как учетные записи, права доступа и файловая система. Эта связь неразрывна и рекурсивна:

— пользователь (реальный или виртуальный) запускает процессы, порождающие файлы;

— права доступа процесса соответствуют правам доступа запустившего его пользователя;

— порожденные процессом файлы получают права, соответствующие правам процесса;

— права пользователя определены параметрами его учетной записи.

Каждый процесс уникален. Для идентификации процесса используется числовое значение - идентификатор процесса (PID, Process Identificator). Для каждого процесса известен владелец (пользователь, запустивший процесс). Если процесс создан реальным пользователем, то он привязан к терминалу, из которого был запушен. Для виртуальных (системных) пользователей такой ассоциации не предусмотрено. Также каждый процесс связан с родительским процессом по его идентификатору (PPID, Parent PID). В каждый момент времени системе известно состояние процесса - степень его исполнения. Процесс может быть:

— выполняемым в текущий момент (R, Runned);

— находящимся в режиме ожидания (S, Suspended);

— прерванным (T, Terminated), например, при использовании клавиш Ctrl+Z;

— "зомби" (Z, Zombied) - завершившимся, но от которого родительский процесс еще не принял сигнала завершения. Спустя некоторое время "зомби" завершаются окончательно и освобождают ресурсы;

— зависшим, или в состоянии непрерывного ожидания (uninterruptible sleep). Такой процесс не реагирует на какие-либо сигналы и может быть снят только перезагрузкой системы.

Еще одной характеристикой процесса является уровень приоритета (NI, NIce value, "степень дружественности"). Он влия-

ет на количество системных ресурсов, выделяемых процессу.

Основными командами для получения сведений о выполняемых процессах являются ps и top. Фрагмент вывода сведений командой ps с параметрами a (расширенный вывод), u (с указанием UID), x (в т.ч для виртуальных пользователей):

```
aag@stilo:~> ps aux
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.0 744 284 ? Ss 14:27 0:00 init [5]
root 2 0.0 0.0 0 0 ? S< 14:27 0:00 [kthreadd]
root 3 0.0 0.0 0 0 ? S< 14:27 0:00 [migration/0]
root 4 0.0 0.0 0 0 ? SN 14:27 0:00 [ksoftirqd/0]
root 5 0.0 0.0 0 0 ? S< 14:27 0:00 [events/0]
root 6 0.0 0.0 0 0 ? S< 14:27 0:00 [khelper]
root 25 0.0 0.0 0 0 ? S< 14:27 0:00 [kblockd/0]
...
root 141 0.0 0.0 0 0 ? S< 14:27 0:00 [kswapd0]
root 142 0.0 0.0 0 0 ? S< 14:27 0:00 [aio/0]
root 367 0.0 0.0 0 0 ? S< 14:27 0:00 [kpsmoused]
root 377 0.0 0.0 0 0 ? S< 14:27 0:00 [kondemand/0]
...
root 991 0.0 0.0 0 0 ? D< 14:28 0:01 [kjournald]
root 1682 0.0 0.0 0 0 ? S< 14:28 0:01 [ipw2200/0]
root 1694 0.0 0.0 0 0 ? S< 14:28 0:00 [khpsbpkt]

wwwrun 3323 0.0 2.4 104548 12804 ? S 14:28 0:00 /usr/sbin/httpd
wwwrun 3324 0.0 2.4 104540 12816 ? S 14:28 0:00 /usr/sbin/httpd
```

Большей гибкостью и универсальностью по сравнению с командой ps обладает команда top. Она позволяет не только получить информацию о процессах, но и выполнять мониторинг через заданные интервалы времени. Так же эта команда позволяет управлять процессами, объединяя возможности команд jobs, nice, fg и kill.

Управление процессами

Пользователь может управлять только теми процессами, владельцем которых является. Но суперпользователь может управлять всеми процессами.

Управление запущенными процессами сводится к приоста-

новке выполнения, изменению приоритета и принудительному завершению.

Приостановить выполнение активного процесса можно сочетанием клавиш Ctrl+Z. Для продолжения его работы следует использовать команду fg. Если имеется несколько приостановленных процессов, то в качестве параметра команды fg необходимо указать порядковый номер задания в текущей оболочке, (не путать с PID), работу которого нужно продолжить. Узнать номер задания можно командой jobs.

Изменение приоритета процесса возникает при необходимости перераспределения ресурсов системы. Значения уровня приоритета (nice value) изменяется от -20 (наименьшая "дружественность", высший приоритет) до +20 (низший приоритет). Все пользовательские (и большинство системных) процессы запускаются с равным приоритетом (nice value = 0).

Для понижения приоритета ранее запущенной задачи используется команда renice с указанием уровня и PID задачи:

[aag@localhost ~]$ renice --7 20117.

Пользователь имеет право *понижать* приоритет *собственных* задач. Повышать уровень приоритета *любой задачи* может только суперпользователь.

Гораздо чаще, чем изменение приоритета, возникает необходимость принудительного завершения (снятия) процесса. Такая ситуация возникает, например, когда процесс "зависает", то есть перестает воспринимать нажатия клавиш и не отвечает на системные события. Для снятия "зависшей" программы предназначена команда kill, которая передает ей один из сигналов завершения. Список сигналов доступен по команде kill -l, а их подробное описание - по команде man 7 signal. Здесь же отметим, что без явного указания имени (или номера), процессу будет передан сигнал SIGTERM (номер 15), предписывающий *по возможности* корректно, с сохранением информации, завершить работу.

Примеры использования команды:

Вызов со значением сигнала по умолчанию (SIGTERM):

```
[aag@localhost ~]$ find / *.html
[aag@localhost ~]$ ps
PID TTY TIME CMD
2663 pts/1 00:00:00 bash
20712 pts/1 00:00:00 find
20762 pts/1 00:00:00 ps
[aag@localhost ~]$ kill 20712
```

Явное указание номера сигнала:

[aag@localhost ~]$ kill -15 20712

Явное указание имени сигнала (номер 9, SIGKILL, требующий немедленного завершения работы программы):

[aag@localhost ~]$ kill -SIGKILL 20712

Задания для самостоятельной работы

1. Войти в систему с собственной учетной записью.
2. Получить справку о команде ps.
3. Командой ps вывести краткую информацию о выполняющихся процессах в текущем терминале и определить PID текущей оболочки.
4. Получить подробную информацию о загруженных процессах и выяснить, какой из них использует максимальный объем памяти, а какой - максимально загружает процессор.
5. Из таблицы, полученной в п.4, выяснить, какой PID имеет процесс init и от чьего имени он запущен.
6. Открыть новый сеанс с собственной учетной записью в tty2 и запустить в нем файловый менеджер MC.
7. Вернуться в tty1 и снова просмотреть список процессов. Определить PID MC, запущенного от вашего имени.
8. Повторить п.6 для пользователей root и stud соответственно в tty3 и tty4.
9. Вернуться в tty1 и определить PID MC, запущенного от имени root и stud.
10. Командой kill снять все процессы MC.
11. Перейти в tty3 (сеанс root) и повторить п.10. Чем можно объяснить различия в результатах выполнения?
12. В tty1 выполнить команду top. Сравнить ее возможности с возможностями ps.
13. Используя top или ps определить, какие процессы порождены (поле PPID) процессом init (PID=1).
14. Завершить сеансы в tty3 и tty4.
15. В tty1 запустить поиск всех файлов .html от каталога /. Приостановить этот процесс (Ctrl+Z). Запустить команду man bash и приостановить ее выполнение.
16. Командой jobs определить номера задач, запущенных в п. 15.
17. Командой fg продолжить выполнение man bash.
18. Принудительно (kill) завершить команду find.
19. Завершить все открытые сеансы.

Контрольные вопросы

1. Что подразумевается под процессами в Unix-подобных системах?
2. Каким может быть процесс?
3. На что влияет уровень приоритета процесса?
4. Основными командами для получения сведений о выполняемых процессах являются…
5. Что такое изменение приоритета процесса?