Linux – нет ничего проще! Часть 1

Основы работы с процессами в Linux!

Добрый день сегодня хочу с Вами поделиться своими воспоминаниями о том времени когда я только начинал заниматься администрирование операционной системы Linux, и как все начинающие администраторы которые раньше занимались только лишь «Окнами», и которые были просто в ужасе от того что опять необходимо возвращаться в те времена когда мы проводили все дни на пролет перед унылым и скучным досовским окном, с эти надоедливо мигающим подобием курсора от которого так и тянуло поспать :). Но высказывание «Админ спит — зарплата идет!» не про меня, и хотя я конечно и был практически на грани отчаяния, я все таки приступил к освоению этого детища **Линуса Торвальдса**.

Основная и в принципе первая проблема с которой я столкнулся это была работа с процессами в Linux. О них я и хочу сегодня в кратце Вам рассказать.

Итак начнем!

Немного определений!

1. Процесс в Linux это **– совокупность программного кода и данных, загруженных в память ЭВМ.**
2. Работающий процесс – в данный момент код процесса выполняется.
3. Спящий процесс – в данный момент код процесса не выполняется в ожидании какого либо события (нажатия клавиши на клавиатуре, поступление данных из сети и т.д.)
4. Процесс-зомби – сам процесс уже не существует, его код и данные выгружены из оперативной памяти, но запись в таблице процессов остается по тем или иным причинам.
5. **PID - Process Identifier** – идентификатор процесса.
6. **Демон** - Демуонами в мире Unix традиционно называются процессы, которые не взаимодействуют с пользователем напрямую.

Допусти что вы запускаете на своем Linux на выполнение какую то программку или команду и так как это очень умная операционная система она автоматически присваивает создает процесс который для удобства называется как правило так как и то что Вы запустили. И к тому же присваивает ему еще и PID идентификатор - числовые идентификаторы (личные номера) в диапазоне от 1 до 65535. PID является именем процесса, по которому мы можем адресовать процесс в операционной системе при использовании различных

средств просмотра и управления процессами.

Как же увидеть запущенные нами процессы и демоны!

Для этого в Linux существует встроенная утилита которая позволяет вывести на экран наши процессы.

Общий синтаксис: ps [PID][options].

Где: options может принимать след значения:

-a или -e – показать все процессы.

-f – полный листинг

-w - показать полные строки описания процессов. Если они превосходят длину экрана, то перенести описание на следующую строку.

-l – полный вывод.

Это лишь малая часть того что может эта замечательная команда. Подробнее о ней Вы можете узнать набрав **man ps** в Вашей консоле.

Все процессы в Linux выполняются и получают доступ к ресурсам системы на основании так называемой «Системы приоритетов». Что это такое система приоритетов указывает ядру ОС какой процесс необходимо выполнять в первую очередь. И этим Вы как администратор можете управлять через команду **nice -n command.**

nice -n command - позволяет изменять приоритет, с которым будет выполняться процесс после запуска. Без указания команды command выдает текущий приоритет работы. n по умолчанию равен 10. Диапазон приоритетов расположен от -20 (наивысший приоритет) до 19 (наименьший).

Приведу в пример еще несколько полезных команд:

nohup command – позволяет процессу продолжить выполнение даже при потере управляющего терминала (SIGHUP). Эту команду выгодно использовать когда необходимо выполнить команду продолжительного действия. Вы запускаете команду и закрываете терминальный сеанс, а она при этом продолжает выполняться. Вывод команды nohup сохранит в файл nohup.out в текущем каталоге.

kill -SIGNAL pid– посылает сигнал процессу с идентификатором pid. Если сигнал не указан, команда посылает процессу сигнал SIGTERM.

killall -s SIGNAL – посылает сигнал всем процессам с именем процесс. Если сигнал не указан, посылает SIGTERM.

Вот список всех сигналов, существующих в системе на сегодняшний день:

Название	Действие по умолчанию	Значение
SIGABRT	Завершить + core	Сигнал отправляется, если процесс вызывает системный вызов abort()
SIGTERM	Завершить	Сигнал обычно представляет своего рода предупреждение, что процесс вскоре будет уничтожен. Этот сигнал позволяет процессу соответствующим образом "подготовиться к смерти" – удалить временные файлы, завершить необходимые транзакции и т.д. Команда kill по умолчанию отправляет именно этот сигнал.
SIGTTIN	Остановить	Сигнал генерируется ядром (драйвером управляющего терминала) при попытке процесса фоновой группы осуществить чтение с управляющего терминала.
SIGTTOU	Остановить	Сигнал генерируется ядром (драйвером терминала) при попытке процесса фоновой группы осуществить запись на управляющий терминал.
SIGALRM	Завершить	Сигнал отправляется, когда срабатывает таймер, ранее установленный.
SIGBUS	Завершить + core	Сигнал свидетельствует о некоторой аппаратной ошибке. Обычно этот сигнал отправляется при обращении к недопустимому виртуальному адресу, для которого отсутствует соответствующая физическая страница.
SIGCHLD	Игнорировать	Сигнал, посылаемый родительскому процессу при завершении его потомка.
SIGSEGV	Завершить + core	Сигнал свидетельствует об обращении процесса к недопустимому адресу или области памяти, для которой у процесса недостаточно привилегий доступа.
SIGFPE	Завершить + core	Сигнал свидетельствует о возникновении особых ситуаций, таких как деление на 0 или переполнение операции с плавающей точкой.
SIGHUP	Завершить	Сигнал посылается лидеру сеанса, связанному с управляющим терминалом, что терминал отсоединился (потеря линии). Сигнал также посылается всем процессам текущей группы при завершении выполнения лидера.

Название	Действие по умолчанию	Значение
		Этот сигнал иногда используют в качестве простейшего средства межпроцессного взаимодействия. В частности, он применяется для сообщения демонам о необходимости обновить конфигурационную информацию. Причина выбора именно сигнала SIGHUP заключается в том, что демон по определению не имеет управляющего терминала и, соответственно, обычно не получает этого сигнала.
SIGILL	Завершить + core	Сигнал посылается ядром, если процесс попытается выполнить недопустимую инструкцию.
SIGINT	Завершить	Сигнал посылается ядром всем процессам при нажатии клавиши прерывания (<CTRL>+<C>)
SIGKILL	Завершить	Сигнал, при получении которого выполнение процесса прекращается. Этот сигнал нельзя не перехватить, не проигнорировать.
SIGPIPE	Завершить	Сигнал посылается при попытке записи в сокет, получатель данных которого завершил выполнение или закрыл файловый указатель на сокет.
SIGPOLL	Завершить	Сигнал отправляется при наступлении определенного события для устройства, которое является опрашиваемым (например, получен пакет по сети)
SIGPWR	Игнорировать	Сигнал генерируется при угрозе потери питания. Обычно он отправляется, когда питание системы переключается на источник бесперебойного питания (UPS).
SIGQUIT	Завершить	Сигнал посылается всем процессам текущей группы при нажатии клавиш <CTRL>+<\>.
SIGSTOP	Остановить	Сигнал отправляется всем процессам текущей группы при нажатии пользователем клавиш <CTRL>+<Z>. Получение сигнала вызывает останов выполнения процесса.
SIGSYS	Завершить + core	Сигнал отправляется ядром при попытке осуществления процессом недопустимого системного вызова.
SIGUSR1	Завершить	Сигнал предназначен для прикладных задач как простейшее средство

Название	Действие по умолчанию	Значение
		межпроцессного взаимодействия.
SIGUSR2	Завершить	Сигнал предназначен для прикладных задач как простейшее средство межпроцессного взаимодействия.