Содержание

7 Руководство администратора

Общие принципы работы VeOS

Процессы и файлы

VeOS является многопользовательской интегрированной системой, т.е. она разработана в расчете на одновременную работу нескольких пользователей. Пользователь может либо сам работать в системе, выполняя некоторую последовательность команд, либо от его имени могут выполняться прикладные процессы. Пользователь взаимодействует с системой через командный интерпретатор, например, bash. Командный интерпретатор представляет собой прикладную программу, которая принимает от пользователя команды или набор команд и переводит их в системные вызовы к ядру системы. Интерпретатор позволяет пользователю работать с файлами, передвигаться по дереву файловой системы, запускать прикладные процессы. Все командные интерпретаторы UNIX имеют развитый командный язык и позволяют писать сложные программы, упрощающие процесс администрирования системы и работы с ней.

Процессы функционирования ОС

Все программы, которые загружены в память сервера (т.е. выполняются) в текущий момент времени, называются процессами. Процессы можно разделить на два основных класса: системные процессы и пользовательские процессы. Системные процессы — программы, решающие внутренние задачи ОС, например, организацию виртуальной памяти на диске или предоставляющие пользователям те или иные сервисы (процессы-службы). Пользовательские процессы — процессы, запускаемые пользователем из командного интерпретатора для решения задач пользователя или управления системными процессами. VeOS изначально разрабатывался как многозадачная система. Он использует технологии, опробованные и отработанные другими реализациями UNIX - предшественниками VeOS. Фоновый режим работы процесса — режим, когда программа может работать без взаимодействия с пользователем. В случае необходимости интерактивной работы с пользователем (в общем случае) процесс будет «остановлен» ядром, и работа его продолжается только после переведения его в «нормальный» режим работы.

Файловая система ОС

В VeOS использована файловая система Linux, которая является единым деревом. Корень этого дерева — каталог, называемый root (рут) и обозначаемый /. Части дерева файловой системы могут физически располагаться в разных разделах разных дисков или вообще на других компьютерах — для пользователя это прозрачно. Процесс присоединения файловой системы раздела к дереву называется монтированием, удаление — размонтированием. Например, файловая система CD-ROM в дистрибутиве монтируется по умолчанию в каталог /media/cdrom (путь в дистрибутиве обозначается с использованием /, а не \, как в Windows). Текущий каталог обозначается ".", родительский каталог (уровнем выше) - "..", например, для перехода в каталог верхнего уровня:

```
> cd ..
```

Структура каталогов

Корневой каталог /:

- /bin — командные оболочки (shell), основные утилиты;
- /boot — содержит ядро системы;
- /dev — псевдофайлы устройств, позволяющие работать с устройствами напрямую. Файлы в /dev создаются сервисом udev
- /etc — общесистемные конфигурационные файлы для большинства программ в системе;
- /etc/rc?.d, /etc/init.d, /etc/rc.boot, /etc/rc.d — каталоги, где расположены командные файлы, выполняемые при запуске системы или при смене её режима работы;
- /etc/passwd — база данных пользователей, в которой содержится информация об имени пользователя, его настоящем имени, личном каталоге, его зашифрованный пароль и другие данные;
- /etc/shadow — теневая база данных пользователей. При этом информация из файла /etc/passwd перемещается в /etc/shadow, который недоступен для чтения всем, кроме пользователя root. В случае использования альтернативной схемы управления теневыми паролями (ТСВ), все теневые пароли для каждого пользователя располагаются в каталоге /etc/tcb/имя пользователя/shadow;
- /home — домашние каталоги пользователей;
- /lib — содержит файлы динамических библиотек, необходимых для работы большей части приложений, и подгружаемые модули ядра;
- /lost+found — восстановленные файлы;
- /media — подключаемые носители (каталоги для монтирования файловых систем сменных устройств);
- /mnt — точки временного монтирования;
- /opt — вспомогательные пакеты;
- /opt/vasexperts - родительский каталог продуктов компании ВАСЭкспертс
- /proc — виртуальная файловая система, хранящаяся в памяти компьютера при загруженной ОС. В данном каталоге расположены самые свежие сведения обо всех процессах, запущенных на компьютере.
- /root — домашний каталог администратора системы;
- /run — файлы состояния приложений;
- /sbin — набор программ для административной работы с системой (системные утилиты);
- /selinux — виртуальная файловая система SELinux;
- /srv — виртуальные данные сервисных служб;
- /sys — файловая система, содержащая информацию о текущем состоянии системы;
- /tmp — временные файлы.
- /usr — пользовательские двоичные файлы и данные, используемые только для чтения (программы и библиотеки);
- /var — файлы для хранения изменяющихся данных (рабочие файлы программ, очереди, журналы).

Каталог /usr:

- /usr/bin — дополнительные программы для всех учетных записей;
- /usr/sbin — команды, используемые при администрировании системы и не предназначенные для размещения в файловой системе root;

- /usr/local — место, где рекомендуется размещать файлы, установленные без использования пакетных менеджеров, внутренняя организация каталогов практически такая же, как и корневого каталога;
- /usr/man — каталог, где хранятся файлы справочного руководства man;
- /usr/share — каталог для размещения общедоступных файлов большей части приложений.

Каталог /var:

- /var/log — место, где хранятся файлы аудита работы системы и приложений;
- /var/spool — каталог для хранения файлов, находящихся в очереди на обработку для того или иного процесса (очереди печати, непрочитанные или не отправленные письма, задачи cron т.д.).

Организация файловой структуры

Система домашних каталогов пользователей помогает организовывать безопасную работу пользователей в многопользовательской системе. Вне своего домашнего каталога пользователь обладает минимальными правами (обычно чтение и выполнение файлов) и не может нанести ущерб системе, например, удалив или изменив файл. Кроме файлов, созданных пользователем, в его домашнем каталоге обычно содержатся персональные конфигурационные файлы некоторых программ. Маршрут (путь) — это последовательность имён каталогов, представляющая собой путь в файловой системе к данному файлу, где каждое следующее имя отделяется от предыдущего наклонной чертой (слешем). Если название маршрута начинается со слеша, то путь в искомый файл начинается от корневого каталога всего дерева системы. В обратном случае, если название маршрута начинается непосредственно с имени файла, то путь к искомому файлу должен начаться от текущего каталога (рабочего каталога). Имя файла может содержать любые символы за исключением косой черты (/). Однако следует избегать применения в именах файлов большинства знаков препинания и непечатаемых символов. При выборе имен файлов рекомендуется ограничиться следующими символами:

- строчные и ПРОПИСНЫЕ буквы.
- символ подчеркивания (_);
- точка (.).

Имена файлов и каталогов - регистрозависимы, т.е. команды touch test.txt и touch TEST.TXT создадут два разных файла с соответствующими именами

Для удобства работы точку можно использовать для отделения имени файла от расширения файла. Данная возможность может быть необходима пользователям или некоторым программам, но не имеет значение для shell.

Имена дисков и разделов

Все физические устройства вашего компьютера отображаются в каталог /dev файловой системы дистрибутива (об этом — ниже). Диски (в том числе IDE/SATA/SCSI/SAS жёсткие диски, USB-диски) имеют имена:

- /dev/sda — первый диск;
- /dev/sdb — второй диск;

и т.д. Диски обозначаются /dev/sdX, где X — a, b, c, d, e, … в зависимости от порядкового номера диска на шине. Раздел диска обозначается числом после его имени. Например, /dev/sdb4 — четвертый раздел второго диска.

Разделы, необходимые для работы VeOS

Для работы ОС на жестком диске (дисках) должны быть созданы, по крайней мере, два раздела: корневой (то есть тот, который будет содержать каталог /) и раздел подкачки (swap). Размер последнего, как правило, составляет от однократной до двукратной величины оперативной памяти компьютера. Если на диске много свободного места, то можно создать отдельные разделы для каталогов /usr, /home, /var.

Работа с наиболее часто используемыми компонентами

Виртуальная консоль

VeOS предоставляет доступ к виртуальным консолям, с которых можно осуществлять одновременно несколько сеансов работы в системе (сессий). Переключение между первыми шести виртуальными консолями производится нажатием комбинации клавиш Alt+F1 — Alt+F6 (Ctrl+Alt+F1 — Ctrl+Alt+F6).

Командные оболочки

Для управления VeOS используются командные интерпретаторы (shell). Зайдя в систему, Вы увидите приглашение — строку, содержащую символ «$» (далее этот символ будет обозначать командную строку). Программа ожидает ваших команд. Роль командного интерпретатора — передавать ваши команды операционной системе. При помощи командных интерпретаторов можно писать небольшие программы — сценарии (скрипты). В VeOS доступны следующие командные оболочки:

- bash — Bourne Again Shell, самая распространенная оболочка
- ksh — Korn Shell, хорошо известная оболочка в UNIX™ системах.

Проверить, какая оболочка используется в данный момент можно, выполнив команду:

```
$ echo $SHELL
```

Оболочкой по умолчанию является Bash — самая распространённая оболочка под Linux, которая ведет историю команд и предоставляет возможность их редактирования.

Командная оболочка Bash

В Bash имеется несколько приемов для работы со строкой команд. Например, можно использовать следующие сочетания:

- Ctrl+A — перейти на начало строки;
- Ctrl+U — удалить текущую строку;
- Ctrl+C — остановить текущую задачу.

Для ввода нескольких команд одной строкой можно использовать разделитель «;». По истории команд можно перемещаться с помощью клавиш ↑ («вверх») и ↓ («вниз»). Чтобы найти конкретную команду в списке набранных, не пролистывая всю историю, можно нажать Ctrl+R и начать вводить символы ранее введенной команды. Дуля просмотра истории команд можно воспользоваться командой histor. Команды, присутствующие в истории, отображаются в списке пронумерованными. Чтобы запустить конкретную команду необходимо набрать:

```
> !номер команды
```

Если ввести:

```
> !!
```

запустится последняя из набранных команд.

В Bash имеется возможность подстановки имен команд из общего списка команд, что существенно облегчает работу с системой. При нажатии клавиши Tab, Bash завершает имя команды, программы или каталога, либо выводит несколько альтернативных вариантов, предлагаю пользователю ввести следующий символ и нажать клавишу Tab для уточнения выбора. Например, чтобы использовать программу декомпрессии gunzip, можно набрать следующую команду:

```
> gu
```

Затем нажать клавишу Tab. Так как в данном случае существует несколько возможных вариантов завершения команды, то необходимо повторно нажать клавишу Tab, чтобы получить список имен, начинающихся с gu. В предложенном примере можно получить следующий список:

```
> gu
guile gunzip gupnp-binding-tool
```

Если набрать: n (gunzip — это единственное имя, третьей буквой которого является «n»), а затем нажать клавишу Tab, то оболочка самостоятельно дополнит имя. Чтобы запустить команду нужно нажать Enter. Программы, вызываемые из командной строки, Bash ищет в каталогах, определяемых в системной переменной $PATH. По умолчанию в этот перечень каталогов не входит текущий каталог, обозначаемый ./ (точка слеш). Поэтому, для запуска программы из текущего каталога, необходимо использовать команду (в примере запускается команда test):

```
> ./test
```

Команда

Простейшая команда состоит из одного «слова» и может быть как встроенной в интерпретатор, так и отдельной командой, находящейся в исполняемом бинарном файле на диске. Текст, что следует за командой называется параметрами (или аргументами) и они вводятся для изменения поведения команды. В большинстве случаев, первое слово считается именем команды, а остальные — её параметрами. Поведением команд можно управлять, например, для изменения формата вывода команды ls можно использовать ключ -l.

```
> ls
> ls -l
```

Такого рода параметры называются ключами или модификаторами выполнения. Ключ принадлежит данной конкретной команде и сам по себе смысла не имеет. Этим он отличается от других параметров (например, имён файлов, чисел), имеющих собственный смысл, не зависящий ни от какой команды. Каждая команда может распознавать некоторый набор ключей и соответственно изменять своё поведение. Один и тот же ключ может определять для разных команд совершенно разные значения. Для формата ключей нет жёсткого стандарта, однако существуют договорённости:

- Если ключ начинается на -, то это простой ключ. За -, как правило, следует один символ, чаще всего буква, обозначающая действие или свойство, которое этот ключ придаёт команде. Так проще отличать ключи от других параметров.
- Если ключ начинается на --, то он называется расширенным ключом. Расширенный формат ключа начинается на два знака --, за которыми следует полное имя обозначаемого этим ключом содержания.

Некоторые ключи имеют и простой, и расширенный формат, а некоторые — только один из форматов. Информацию о ресурсах каждой команды можно получить, используя ключ --help или -h. К примеру, получить подсказку о том, что делает команда touch, можно, набрав в терминале touch --help.

Стыкование команд в системе VeOS

Стандартный ввод и стандартный вывод

Многие команды системы имеют так называемые стандартный ввод (standard input) и стандартный вывод (standard output), часто сокращаемые до stdin и stdout. Ввод и вывод — это входная и выходная информация для данной команды. Программная оболочка делает так, что стандартным вводом является клавиатура, а стандартным выводом — экран монитора. Пример с использованием команды cat. По умолчанию команда cat читает данные из всех файлов, которые указаны в командной строке, и посылает эту информацию непосредственно в стандартный вывод (stdout). Следовательно, команда: cat /etc/os-release выведет на экран сначала содержимое файла. Если имя файла не указано, программа cat читает входные данные из stdin и возвращает их в stdout. Пример: cat Привет. Привет. Пока. Пока. Ctrl-D Каждую строку, вводимую с клавиатуры, программа cat возвращает на экран. При вводе информации со стандартного ввода конец текста сигнализируется вводом специальной комбинации клавиш, как правило, Ctrl+D. Сокращённое название сигнала конца текста — EOT

(end of text).

Перенаправление ввода и вывода

При необходимости можно перенаправить стандартный вывод, используя символ >, и стандартный ввод, используя символ <. Фильтр (filter) — программа, которая читает данные из стандартного ввода, некоторым образом их обрабатывает и результат направляет на стандартный вывод. Когда применяется перенаправление, в качестве стандартного ввода и вывода могут выступать файлы. Как указывалось выше, по умолчанию, stdin и stdout относятся к клавиатуре и к экрану соответственно. Программа sort является простым фильтром — она сортирует входные данные и посылает результат на стандартный вывод. Совсем простым фильтром является программа cat — она ничего не делает с входными данными, а просто пересылает их на выход.

Использование состыкованных команд

Стыковку команд (pipelines) осуществляет командная оболочка, которая stdout первой команды направляет на stdin второй команды. Для стыковки используется символ |. Направить stdout команды ls на stdin команды sort:

```
> ls -la /etc| sort -r
-rw-r--r--.  1 root root        9 июн  7  2013 host.conf
-rw-r--r--.  1 root root      970 авг  8  2019 yum.conf
-rw-r--r--.  1 root root      966 авг  9  2019 rwtab
-rw-r--r--.  1 root root       94 мар 24  2017 GREP_COLORS
```

Вывод списка файлов частями:

```
> ls /usr/bin | more
```

Если необходимо вывести на экран последнее по алфавиту имя файла в текущем каталоге, можно использовать следующую команду:

```
> ls | sort -r | head -1 notes
```

где команда head -1 выводит на экран первую строку получаемого ей входного потока строк (в примере поток состоит из данных от команды ls), отсортированных в обратном алфавитном порядке.

Перенаправление вывода в режиме добавления

ффект от использования символа > для перенаправления вывода файла является деструктивным; т.е, команда

```
> ls > file-list
```

уничтожит содержимое файла file-list, если этот файл ранее существовал, и создаст на его месте новый файл. Если вместо этого перенаправление будет сделано с помощью символов », то вывод будет приписан в конец указанного файла, при этом исходное содержимое файла не будет уничтожено.

Перенаправление ввода и вывода и стыкование команд осуществляется командными оболочками, которые поддерживают использование символов >, » и |. Сами команды не способны воспринимать и интерпретировать эти символы.

Режим суперпользователя

VeOS — система многопользовательская, а потому пользователь — ключевое понятие для организации всей системы доступа в . Файлы всех пользователей в VeOS хранятся раздельно, у каждого пользователя есть собственный домашний каталог, в котором он может хранить свои данные. Доступ других пользователей к домашнему каталогу пользователя может быть ограничен. Суперпользователь в VeOS — это выделенный пользователь системы, на которого не распространяются ограничения прав доступа. Именно суперпользователь имеет возможность произвольно изменять владельца и группу файла. Ему открыт доступ на чтение и запись к любому файлу или каталогу системы. Среди учётных записей VeOS всегда есть учётная запись суперпользователя — root. Поэтому вместо «суперпользователь» часто говорят «root». Множество системных файлов принадлежат root, множество файлов только ему доступны для чтения или записи. Пароль этой учётной записи — одна из самых больших драгоценностей системы. Именно с её помощью системные администраторы выполняют самую ответственную работу.

Системные утилиты требуют для своей работы привилегий суперпользователя, потому что они вносят изменения в системные файлы. При их запуске выводится предупреждение о недостаточных правах текущего пользователя.

Для опытных пользователей, умеющих работать с командной строкой, существует два различных способа получить права суперпользователя. Первый — это войти в систему под именем root. Второй способ — воспользоваться специальной утилитой su (shell of user), которая позволяет выполнить одну или несколько команд от лица другого пользователя. По умолчанию эта утилита выполняет команду sh от пользователя root, то есть запускает командный интерпретатор. Отличие от предыдущего способа в том, что всегда известно, кто именно запускал su, а значит, ясно, кто выполнил определённое административное действие. В некоторых случаях удобнее использовать не su, а утилиту sudo, которая позволяет выполнять только заранее заданные команды.

Для того чтобы воспользоваться командами su и sudo, необходимо быть членом группы wheel. Пользователь, созданный при установке системы, по умолчанию уже включён в эту группу.

Для перехода в режим суперпользователя наберите в терминале команду su -. Если воспользоваться командой su без ключа, то происходит вызов командного интерпретатора с правами root. При этом значение переменных окружения, в частности $PATH, остаётся таким

же, как у пользователя: в переменной $PATH не окажется каталогов /sbin, /usr/sbin, без указания полного имени будут недоступны команды route, shutdown, mkswap и другие. Более того, переменная $HOME будет указывать на каталог пользователя, все программы, запущенные в режиме суперпользователя, сохранят свои настройки с правами root в каталоге пользователя, что в дальнейшем может вызвать проблемы. Чтобы избежать этого, следует использовать su -. В этом режиме su запустит командный интерпретатор в качестве login shell, и он будет вести себя в точности так, как если бы в системе зарегистрировался root.

Управление пользователями

Пользователи и группы внутри системы обозначаются цифровыми идентификаторами — UID и GID, соответственно. Пользователь может входить в одну или несколько групп. По умолчанию он входит в группу, совпадающую с его именем. Чтобы узнать, в какие еще группы входит пользователь, введите команду id, вывод её может быть примерно следующим:

```
uid=500(test) gid=500(test) группы=500(test),16(rpm)
```

Такая запись означает, что пользователь test (цифровой идентификатор 500) входит в группы test и rpm. Разные группы могут иметь разный уровень доступа к тем или иным каталогам; чем в большее количество групп входит пользователь, тем больше прав он имеет в системе.

В связи с тем, что большинство привилегированных системных утилит в VeOS имеют не SUID-, а SGID-бит, будьте предельно внимательны и осторожны в переназначении групповых прав на системные каталоги.

Команда passwd

Команда passwd поддерживает традиционные опции passwd и утилит shadow. Синтаксис:

```
passwd [ПАРАМЕТРЫ...] <пользователь>
-k, --keep-tokens       сохранить неустаревшие данные авторизации (пароли)
-d, --delete            удалить пароль пользователя (только root)
-l, --lock              заблокировать пароль пользователя (только root)
-u, --unlock            разблокировать пароль пользователя (только root)
-e, --expire            просрочить пароль пользователя (только root)
-f, --force             принудительное выполнение
-x, --maximum=DAYS      максимальный срок действия пароля (только root)
-n, --minimum=DAYS      минимальный срок действия пароля (только root)
-w, --warning=DAYS      за сколько дней до истечения пароля начать предупреждать
пользователя (только root)
-i, --inactive=DAYS     через сколько дней после истечения пароля заблокировать
учетную запись (только root)
-S, --status            сообщить состояние пароля для пользователя (только root)
--stdin                 получить новое значение из stdin (только root)
```

Код выхода: при успешном завершении passwd заканчивает работу с кодом выхода 0. Код

выхода 1 означает, что произошла ошибка. Текстовое описание ошибки выводится на стандартный поток ошибок. Пользователь может в любой момент поменять свой пароль. Единственное,что требуется для смены пароля знать текущий пароль. Только суперпользователь может обновить пароль другого пользователя.

Добавления нового пользователя

Для добавления нового пользователя используйте команды useradd и passwd:

```
> useradd test
> passwd test
Изменяется пароль пользователя test.
Новый пароль :
```

В результате в системе появился пользователь test с заданным паролем. Если пароль оказался слишком слабым с точки зрения системы, она об этом предупредит. Пользователь в дальнейшем может поменять свой пароль при помощи команды passwd — но если он попытается поставить слабый пароль, система откажет ему (в отличие от root) в изменении. Программа useradd имеет множество параметров, которые позволяют менять её поведение по умолчанию. Например, можно принудительно указать, какой будет UID или какой группе будет принадлежать пользователь.

Модификация пользовательских записей

Для модификации пользовательских записей применяется утилита usermod:

```
> usermod -G wheel,test test
```

Такая команда изменит список групп, в которые входит пользователь test — теперь это wheel, test.

```
> usermod -l test1 test
```

Будет произведена смена имени пользователя с test на test1. Команды usermod -L test1 и usermod -U test1 соответственно временно блокируют возможность входа в систему пользователю test1 и возвращают всё на свои места. Изменения вступят в силу только при следующем входе пользователя в систему. При неинтерактивной смене или задании паролей для целой группы пользователей используйте утилиту chpasswd. На стандартный вход ей следует подавать список, каждая строка которого будет выглядеть как имя:пароль.

Удаление пользователей

Для удаления пользователей используйте userdel. Команда userdel test1 удалит пользователя test1 из системы. Если будет дополнительно задан параметр -r, то будет уничтожен и домашний каталог пользователя. Нельзя удалить пользователя, если в данный момент он еще работает в системе

Система инициализации systemd

Запуск операционной системы

Алгоритм запуска компьютера приблизительно такой:

1. BIOS компьютера.
2. Загрузчик системы (например, LILO, GRUB или другой). В загрузчике вы можете задать параметры запуска системы или выбрать систему для запуска.
3. Загружается ядро Linux.
4. Запускается на выполнение первый процесс в системе — init.
5. Ядром запускается самая первая программа в системе init. Её задачей является запуск новых процессов и повторный запуск завершившихся. Вы можете посмотреть, где расположился init в иерархии процессов вашей системы, введя команду pstree.

От конфигурации init зависит, какая система инициализации будет использована. Система инициализации — это набор скриптов, которые будут выполнены при старте системы.

Система инициализации systemd является основной системой инициализации VeOS, вобравшей в себя достоинства классического System V init и более современных launchd (OS X), SMF (Solaris) и Upstart (Ubuntu, Fedora), но при этом лишенной многих их недостатков. Он разрабатывался для обеспечения лучшего выражения зависимостей между службами, что позволяет делать одновременно больше работы при загрузке системы, и уменьшить время загрузки системы. systemd (system daemon) реализует принципиально новый подход к инициализации и контролю работы системы. Одним из ключевых новшеств этого подхода является высокая степень параллелизации запуска служб при инициализации системы, что в перспективе позволяет добиться гораздо более высокой скорости, чем традиционный подход с последовательным запуском взаимозависимых служб. Другим важным моментом является контроль над точками монтирования (не-жизненно-важные файловые системы можно монтировать только при первом обращении к ним, не тратя на это время при инициализации системы) и устройствами (можно запускать и останавливать определенные службы и при появлении или удалении заданных устройств). Для отслеживания групп процессов используется механизм cgroups, который также может быть использован для ограничения потребляемых ими системных ресурсов. Удобство systemd особенно заметно на компьютерах для домашнего пользования — когда пользователи включают и перезагружают компьютер ежедневно. В отличие от sysvinit, подвисание при запуске одного сервиса не приведет к остановке всего процесса загрузки.

Примеры команд управления службами, журнал в systemd

Команды service и chkconfig продолжат работать в мире systemd практически без изменений. Тем не менее, в этой таблице показано как выполнить те же действия с помощью встроенных утилит systemctl.

Команды Sysvinit	Команды Systemd	Примечания
service fastdpi start	systemctl start fastdpi.service	Используется для запуска службы (не перезагружает постоянные
service fastdpi stop	systemctl stop fastdpi.service	Используется для остановки службы (не перезагружает постоянные)
service fastdpi restart	systemctl restart fastdpi.service	Используется для остановки и последующего запуска службы
service fastdpi reload	systemctl reload fastdpi.service	Если поддерживается, перезагружает файлы конфигурации без прерывания незаконченных операций
service fastdpi condrestart	systemctl condrestart fastdpi.service	Перезапускает службу, если она уже работает
service fastdpi status	systemctl status fastdpi.service	Сообщает, запущена ли уже служба
ls /etc/rc.d/init.d/	systemctl list-unit-files --type=service (preferred)	Используется для отображения списка служб, которые можно запустить или остановить.
chkconfig fastdpi on	systemctl enable fastdpi.service	Включает службу во время следующей перезагрузки, или любой другой триггер
chkconfig fastdpi off	systemctl disable fastdpi.service	Выключает службу во время следующей перезагрузки, или любой другой триггер
chkconfig fastdpi	systemctl is-enabled fastdpi.service	Используется для проверки, сконфигурирована ли служба для запуска в текущем окружении
chkconfig --list	systemctl list-unit-files --type=service(preferred)	Выводит таблицу служб. В ней видно, на каких уровнях загрузки они (не)запускаются
chkconfig fastdpi --list	ls /etc/systemd/system/*.wants/fastdpi.service	Используется, для отображения на каких уровнях служба (не)запускается
chkconfig fastdpi --add	systemctl daemon-reload	Используется, когда вы создаете новую службу или модифицируете любую конфигурацию

Журнал в systemd

В systemd включена возможность ведения системного журнала. Для чтения журнала следует использовать команду journalctl. По умолчанию, больше не требуется запуск службы syslog. Вы можете запускать journalctl с разными ключами:

`> journalctl -b` – покажет сообщения только с текущей загрузки;

`> journalctl -f` – покажет только последние сообщения.

Так же вы можете посмотреть сообщения определенного процесса:

`> journalctl _PID=1` – покажет сообщения первого процесса `(init).`

Для ознакомления с прочими возможностями, читайте руководство по journalctl. Для этого используйте команду man journalctl.