

Уровни

Linux использует концепцию различных уровней выполнения, чтобы определить, какие службы или процессы будут запущены. Хотя ядро Linux может распознавать значения уровня запуска от нуля до девяти, обычно используются только уровни запуска от нуля до шести. И традиционный `init`, и `Upstart` используют эти уровни запуска.

В `Systemd` есть что-то похожее на уровни выполнения, называемые целями, которые показаны в следующей таблице вместе с их эквивалентом уровня выполнения.

Уровень	Цель	Systemd Target
0	Остановить или отключить систему	<code>poweroff.target</code>
1	Однопользовательский режим для административных задач	<code>rescue.target</code>
2	Многопользовательский режим без настроенных сетевых интерфейсов или сетевых сервисов	<code>multi-user.target</code>
3	Нормальный запуск системы	<code>multi-user.target</code>
4	Определяемые пользователем	<code>multi-user.target</code>
5	Запустите систему в обычном режиме с помощью диспетчера графического отображения.	<code>graphical.target</code>
6	Перезагрузите систему	<code>reboot.target</code>

Хотя эти уровни выполнения считаются «стандартными», не все дистрибутивы используют их для одних и тех же целей. Фактически, для систем, которые больше не используют традиционный процесс инициализации, использование уровней выполнения может быть предоставлено только для поддержания совместимости с процессами, которые могут их ожидать.

Уровень запуска по умолчанию

Системы, использующие традиционный `init`, могут изменить уровень выполнения по умолчанию, изменив запись в файле `/etc/inittab`, которая выглядит следующим образом:

`id:5:initdefault:`

В этом примере указывается уровень запуска по умолчанию, при котором система переходит на уровень запуска 5, что типично для настольного компьютера или ноутбука, используемого конечным пользователем. Для большинства систем Linux уровень запуска 5 обеспечивает высочайший уровень функциональности, включая интерфейс с графическим интерфейсом.

Серверы обычно не предлагают интерфейс с графическим интерфейсом, поэтому запись `initdefault` может выглядеть следующим образом:

id:3:initdefault:

Как обсуждалось ранее, уровень запуска по умолчанию может быть переопределен во время загрузки, взаимодействуя с загрузчиком. В случае GRUB Legacy добавьте строку ядра или используйте функцию редактирования, чтобы добавить номер уровня выполнения. Например, добавление 5 приведет к тому, что система перейдет на уровень запуска 5, а слово **single** (или заглавные или строчные буквы **s**) приведет систему к уровню запуска для одного пользователя.

Если система использует Upstart вместо традиционного процесса инициализации, уровень запуска по умолчанию также может быть установлен в `/etc/inittab`, как в случае с дистрибутивами, производными от RedHat Enterprise Linux 6. С другой стороны, дистрибутивы, такие как Ubuntu (дистрибутив, который разработал Upstart), можно изменить, установив переменную окружения `DEFAULT_RUNLEVEL` в файле `/etc/init/rc-sysinit.conf`.

Systemd изначально не использует уровни выполнения, но имеет нечто похожее, называемое целью. Например, **graphical.target** аналогичен стандартному уровню выполнения 5, на котором работает GUI; **multi-user.target** аналогичен стандартному уровню выполнения 3, где система обычно работает без графического интерфейса.

Чтобы установить цель по умолчанию, создайте символическую ссылку из определения цели в файл `/etc/systemd/system/default.target`. Сначала удалите прежнюю цель по умолчанию, а затем создайте новую символическую ссылку, например:

```
rm -f /etc/systemd/system/default.target
ln -sf /lib/systemd/system/graphical.target /etc/systemd/system/default.target
```

В предыдущем примере задано целевое значение по умолчанию - **graphical.target**, поэтому система перейдет в состояние, подобное уровню запуска 5.

Просмотр текущего уровня запуска

Несмотря на наличие уровня запуска по умолчанию, система может иметь другой уровень запуска, либо потому, что она была переопределена во время загрузки, либо изменилась после загрузки системы.

Одной из команд, отображающих текущий уровень выполнения, является команда **runlevel**, которая сначала показывает предыдущий уровень выполнения, а затем текущий уровень выполнения. Если предыдущий уровень выполнения не был достигнут, он покажет N для предыдущего уровня выполнения. Это означает, что система была загружена в текущий уровень запуска напрямую и не переключалась.

В следующем примере демонстрируется выполнение команды **runlevel**, когда система только что загрузилась до уровня запуска 5:

```
root@localhost:~# runlevel
N 5
```

Команда **who -r** также отображает текущий уровень запуска системы. Одним из преимуществ этого метода является то, что он будет отображать дату и время, когда был достигнут текущий уровень выполнения:

who -r

```
root@localhost:~# who -r
run-level 5 2019-05-17 23:18
```

Хотя Systemd на самом деле не использует уровни запуска, он прозрачно переводит свою текущую цель как уровень запуска для совместимости с командами **runlevel** и **who -r**. Если система достигла **multi-user.target**, она будет переведена на уровень выполнения 3; если система достигла **graphical.target**, то это переводится на уровень выполнения 5.

Изменение уровней запуска

И традиционный **init**, и **Upstart** поддерживают передачу уровней запуска в ядро в качестве параметров из загрузчика для переопределения уровня запуска по умолчанию.

Чтобы указать другой уровень выполнения во время загрузки в системе, которая использует **Systemd**, добавьте к параметрам ядра параметр с синтаксисом «**systemd.unit = требуемый.target**», где **требуемый.target** является одной из целей **Systemd**.

Пользователь **root** также может изменять уровни выполнения во время работы операционной системы, используя несколько команд, включая команды **init** и **telinit**, которые позволяют указать желаемый уровень выполнения. Есть также несколько команд, которые напрямую не указывают номер уровня выполнения, но предназначены для того, чтобы система изменила уровни выполнения.

Команды **init** и **telinit**

Чтобы напрямую указать уровень запуска, используйте **init** или **telinit**. В некоторых дистрибутивах команда **telinit** имеет опцию **-t**, которая позволяет указывать задержку в секундах; в противном случае команды **init** и **telinit** идентичны. Фактически в некоторых системах команда **telinit** может быть просто ссылкой на команду **init**.

Чтобы использовать эти команды, просто укажите желаемый уровень выполнения в качестве аргумента. Например, чтобы перезагрузить систему, используйте команду **init 6** или команду **telinit 6**. Или, чтобы перейти на уровень запуска 5, используйте **init 5** или **telinit 5**.

С заменой **Systemd** **init**, команда **init** все еще может использоваться для изменения уровня выполнения; **Systemd** преобразует желаемый уровень выполнения в цель **Systemd**. Например, если выполняется **init 5**, то **Systemd** попытается перейти в состояние **graphical.target**.

Чтобы **Systemd** изначально переключился в целевое состояние, с привилегиями **root** выполните команду **systemctl isolate required.target**.

Например, чтобы перевести систему на уровень запуска 1, выполните команду **systemctl isolate rescue.target**. Аналогично, чтобы перейти на уровень запуска 5, выполните команду **systemctl isolate graphical.target**.

Команды остановки, выключения, перезагрузки и выключения

Чтобы привести систему к нулевому уровню выполнения, выполните команду **shutdown**, **halt** или **poweroff**.

Хотя команды **halt** и **poweroff** начнут немедленно выключать систему, команде **shutdown** требуется аргумент времени, указывающий, когда должно начаться завершение работы. Форматы этого временного аргумента могут быть словом **now**, временем обратного отсчета в формате **hh:mm** или количеством минут для задержки в формате **+m**. Сообщение, которое появится в терминалах всех пользователей, также можно указать с помощью команды выключения. Например:

```
shutdown now "System going down for repairs"
```

Сообщение «System going down for repairs» будет отображаться в окне терминала каждого пользователя, который в данный момент вошел в систему.

Как ни странно, если опция не указана, то команда **shutdown** фактически переводит систему на уровень запуска 1. Команда **shutdown**, используемая с параметром **-g**, аналогична команде **reboot** и заставит систему перейти на уровень выполнения 6. Команда **shutdown**, используемая с параметром **-h**, похожа на команду **halt** и заставит систему перейти на нулевой уровень выполнения.

Не всегда необходимо переходить на уровень выполнения 5 (в большинстве дистрибутивов), чтобы иметь возможность использовать графический интерфейс пользователя (GUI). Графический интерфейс, который обычно предоставляется

программным обеспечением, известным как X Windows, также может быть запущен командой **startx**.

После использования графической среды используйте графический выход для завершения сеанса X Windows.

Использование команды **startx** не требует привилегий **root**, как это делают команды **init 5** или **systemctl isolate graphical.target**.

Команда **wall**

Иногда системному администратору необходимо отправлять сообщения об ожидающем событии всем пользователям. Пока можно отправить сообщение с помощью команды выключения:

```
shutdown now "System going down for repairs"
```

Есть случаи, когда уведомление может не касаться неизбежного завершения работы системы. Для этого используется команда **wall**. Следующее сообщение передается на стену из команды **echo**:

```
echo -e "The server offline " | wall
```

Команда **wall** принимает стандартный ввод или имя файла. Чтобы отобразить файл, команда **wall** либо требует, чтобы пользователь имел привилегии **root**, либо содержимое должно передаваться из другой команды, например, **cat**. Без них команда **wall** будет отображать сообщение об ошибке:

```
nano attention.txt
```

```
Time expired
```

```
Ctl+x
```

```
Y
```

```
wall attention.txt
```

```
wall: will not read attention.txt - use stdin
```

```
su
```

```
Password:
```

```
root@localhost:/home/sysadmin# wall attention.txt  
  
Broadcast message from sysadmin@localhost (tty2) (Fri May 17 23:49:38 2019):  
  
time expired
```

Опция **-n** может использоваться командой **wall** для подавления ведущего баннера:

```
cat attention.txt | wall -n
```

Управление системными службами

Как администратор системы, вы можете контролировать, какие услуги будут предоставляться различными демонами (процессами, которые выполняются в фоновом режиме системы). Если вы хотите протестировать сервисы или временно включить или отключить их, вы можете управлять ими вручную.

Как правило, администраторы хотят автоматизировать управление службами, поэтому, когда система переводится на определенный уровень выполнения или целевое состояние, они будут знать, какие службы должны автоматически быть доступны.

Ручное управление услугами

Если система использует традиционный процесс **init** для управления системными службами, то сценарии в каталоге **/etc/rc.d/init.d** используются для управления состоянием этих системных служб. Для удобства этот каталог обычно имеет символическую ссылку из файла **/etc/init.d**. Сценарии в этом каталоге часто называются сценариями инициализации.

Чтобы вручную управлять состоянием службы, такой как веб-сервер, используйте соответствующий сценарий в каталоге **/etc/rc.d/init.d**, чтобы запустить, остановить или иным образом изменить состояние веб-сервера. Чтобы управлять службой с помощью этих сценариев, запустите сценарий с аргументом, который указывает, что сценарий должен делать.

Например, в дистрибутиве Red Hat Enterprise Linux сценарий для управления веб-сервером имеет путь **/etc/rc.d/init.d/httpd**. Итак, чтобы вручную запустить веб-сервер, выполните следующую команду от имени пользователя root:

```
/etc/rc.d/init.d/httpd start
```

Чтобы вручную остановить работающий веб-сервер, выполните:

```
/etc/rc.d/init.d/httpd stop
```

Вместо того, чтобы вводить полный путь к сценарию, многие системы предоставляют служебный сценарий, который позволяет выполнять сценарий инициализации без необходимости вводить полный путь к сценарию; вместо этого просто укажите имя программы инициализации, так как скрипт должен делать это в качестве аргументов. Например, чтобы запустить и остановить использование веб-сервера:

```
service httpd start
```

```
service httpd stop
```

Также можно остановить и запустить демон веб-сервера, набрав:

```
service httpd restart
```

Различные сценарии имеют разные возможности или функции, которые они могут выполнять. Чтобы узнать, что может делать скрипт, запустите скрипт без каких-либо аргументов. Например:

```
root@localhost:/home/sysadmin# /etc/init.d/cron
[info] Usage: /etc/init.d/cron {start|stop|status|restart|reload|force-reload}.
```

В следующей таблице объясняется назначение этих функций сценария **httpd**, которые также могут реализовывать многие другие сценарии:

Argument	Function
start	Если служба не запущена, попробуйте запустить ее.
stop	Если служба запущена, попытайтесь остановить ее.
restart	Остановите, а затем начните обслуживание заново. Если в службу вносится серьезное изменение конфигурации, возможно, ее придется перезапустить, чтобы изменения вступили в силу.
condrestart	Перезапустите службу при условии, что она в данный момент работает.
try-restart	Такой же как condrestart
reload	Прочитать и загрузить конфигурацию для сервиса. Перезагрузка файла конфигурации службы, как правило, является менее разрушительным способом сделать изменения конфигурации службы эффективными, но может не принести успеха при серьезных изменениях.

Argument	Function
status	Показать, остановлена ли служба или идентификатор процесса (PID), если служба запущена. Примечание. Также можно использовать команду <code>service --status-all</code> , чтобы увидеть состояние всех демонов.
fullstatus	Для веб-сервера apache отображает URL / статус сервера.
graceful	Для веб-сервера apache он корректно перезапускает сервер. Если сервер не работает, он запускается. В отличие от обычного перезапуска, открытые соединения не прерываются.
help	Отображает использование скрипта.
configtest	Проверяет файлы конфигурации на правильность. Для некоторых служб, если файл конфигурации изменен, это можно использовать для проверки того, что изменения не содержат синтаксических ошибок.

Каталоги уровня запуска

Несмотря на то, что можно запускать службы вручную, большинство служб автоматически запускаются любым процессом инициализации, который использует система.

При традиционном процессе инициализации определенные каталоги используются для управления тем, какие службы будут автоматически запускаться или останавливаться на разных уровнях выполнения. Все эти каталоги существуют в каталоге **/etc/rc.d** и имеют следующие имена путей: **rc0.d**, **rc1.d**, **rc2.d**, **rc3.d**, **rc4.d**, **rc5.d** и **rc6.d**. Число в имени каталога представляет уровень запуска, которым он управляет; например, **rc0.d** для нулевого уровня выполнения, а **rc1.d** для первого уровня выполнения.

Чтобы служба запускалась на уровне выполнения, в соответствующем каталоге уровня запуска можно создать символическую ссылку на сценарий инициализации в каталоге **/etc/rc.d/init.d**. Это имя ссылки должно начинаться с буквы **S**, за которой следует число от **1** до **99** и имя сценария инициализации, с которым он связан.

Например, когда **dns** настроен на запуск на пятом уровне выполнения, в каталоге **/etc/rc.d/rc5.d** имеется символическая ссылка с именем **S01bind9**, которая связана с **/etc/rc.d/init.d/httpd** скрипт:

```
root@localhost:/home/sysadmin# ls -l /etc/rc5.d/S01bind9
lrwxrwxrwx 1 root root 15 Jan 24 09:13 /etc/rc5.d/S01bind9 -> ../init.d/bind9
```

Так же, как ссылки на **S**-файлы в каталоге уровня запуска будут указывать, что служба должна быть запущена, так и ссылки на **K**-файлы в каталоге уровня выполнения будут указывать, что служба должна быть остановлена. Используя **dhcpcd**-сервер в качестве примера, чтобы **dhcpcd**-сервер остановился на пятом уровне запуска, имеется символическая ссылка в каталоге **/etc/rc.d/rc5.d**, которая начинается с буквы **K**, за которой следует число от одного до девяноста девяти и имени сценария инициализации, с которым он связан:

```
root@localhost:/home/sysadmin# ls -l /etc/rc5.d/K01isc-dhcp-server
lrwxrwxrwx 1 root root 25 Jan 24 07:44 /etc/rc5.d/K01isc-dhcp-server -> ../init.d/isc-dhcp-server
```

Если ссылка была создана вручную, то перед созданием ссылки остановки необходимо удалить начальную ссылку:

```
rm /etc/rc.d/rc5.d/S85httpd
ln -s /etc/rc.d/init.d/httpd /etc/rc.d/rc5.d/K15httpd
```

Скрипты К иногда путают начинающих администраторов. Они часто задаются вопросом «почему я должен останавливать службу, когда система выводится на рабочий уровень?». Эта путаница обычно возникает из-за того, что уровни выполнения являются лишь «загрузочными объектами», но, как уже упоминалось, администратор может изменить систему с одного уровня выполнения на другой.

Представьте, что служба httpd доступна на уровне запуска 5, но не на уровне запуска 3. Когда система переводится с уровня запуска 5 на уровень запуска 3, службу httpd следует остановить. Отсюда и К сценарии

Причина того, что ссылки «старт» и «стоп» имеют номер после буквы S или K, заключается в том, что службы запускаются или останавливаются в правильной последовательности. Сценарии запускаются (или останавливаются) по порядку, поэтому K15httpd будет выполняться до K35vncserver.

Если службы не запускаются или останавливаются в правильном порядке, они могут работать неправильно, поскольку некоторые службы зависят от других служб. Например, чтобы веб-сервер работал так, как он должен работать, сетевая служба уже должна быть запущена. Если сетевая служба имеет более низкий начальный номер, чем веб-сервер, это означает, что init запустит сетевую службу до того, как попытается запустить веб-службу.

Команда chkconfig

Команда **chkconfig** может использоваться для просмотра того, какие службы будут запущены для разных уровней выполнения. Эту команду также можно использовать для включения или отключения службы для определенных уровней выполнения. В дистрибутивах Linux, которые не являются производными от Red Hat, этот инструмент может быть недоступен.

Чтобы просмотреть все службы, которые настроены на автоматический запуск или остановку, администратор может выполнить команду **chkconfig --list**, и выходные данные будут выглядеть примерно так (хотя будет много выходных строк):

```
[root@localhost ~]# chkconfig --list
```

```
Note: This output shows SysV services only and does not include native
systemd services. SysV configuration data might be overridden by native
systemd configuration.
```

```
If you want to list systemd services use 'systemctl list-unit-files'.
To see services enabled on particular target use
'systemctl list-dependencies [target]'.
```

netconsole	0:off	1:off	2:off	3:off	4:off	5:off	6:off
network	0:off	1:off	2:on	3:on	4:on	5:on	6:off

Выходные данные для каждой строки показывают имя файла сценария, найденного в каталоге **/etc/rc.d/init.d**, за которым следуют каждый номер уровня выполнения, двоеточие и указывается, включена или отключена служба. Например, основываясь на выводе предыдущего графика, служба network запускается на уровнях выполнения 2, 3, 4 и 5.

Чтобы просмотреть настройки отдельной службы, используйте команду сценария **chkconfig --list**, где script - это имя файла сценария, найденного в каталоге **/etc/rc.d/init.d**. Например, чтобы просмотреть скрипт веб-сервера, выполните:

```
chkconfig --list network
```

Чтобы включить запуск службы для большинства уровней выполнения, используйте сценарий **chkconfig** в команде, где сценарий - это имя файла сценария,

найденного в каталоге **/etc/rc.d/init.d**. Таким образом, чтобы разрешить запуск сетевой службы на большинстве уровней выполнения, выполните команду **chkconfig network on**:

Для большинства служб, если для включения службы используется команда **chkconfig**, она будет автоматически запускаться на уровнях выполнения со 2 по 5 и автоматически останавливаться на уровнях выполнения 0, 1 и 6. Это может немного отличаться в зависимости от содержимого самого скрипта.

В скрипте **etc / rc.d / init.d / network** есть строка, которая содержит следующее:

```
# chkconfig: 2345 10 90
```

2345 означает, что **network** по умолчанию включено на уровнях запуска три, четыре и пять.

Чтобы включить или отключить службы для уровня не по умолчанию, можно использовать параметр **--level** с командой **chkconfig**. Например, следующие две команды гарантируют, что служба **atd** будет доступна на уровнях запуска два и четыре, но недоступна на уровнях запуска три и пять:

```
chkconfig --level 24 network on
```

```
chkconfig --level 35 network off
```

Если в выводе есть **-**, то он указывает на то, что служба не включается ни на одном из уровней запуска автоматически при первом добавлении в управление **chkconfig**. Другими словами, эта служба не настроена на автоматический запуск, если только администратор не использует команду **chkconfig httpd on**.

Также имеются две другие опции **chkconfig**, хотя они редко используются напрямую. Параметры **chkconfig --add** и **--del** можно использовать вручную, но обычно они используются автоматически, когда администратор устанавливает новый пакет служебного программного обеспечения или удаляет пакет служебного программного обеспечения.

Если администратор должен был создать сценарий инициализации с именем **serviced** и сохранить его в каталоге **/etc/rc.d/init.d**, то перед использованием команды **chkconfig serviced on** или **chkconfig service off** команда **chkconfig --add serviced** должна быть выполнена первой. Команда, подобная этой, выполняется, когда установлен новый программный пакет, содержащий службу.

Команда **chkconfig --del serviced** удалит все ссылки в каталогах уровня выполнения, и службы не будут запускаться или останавливаться автоматически на любом уровне выполнения. Команда, подобная этой, выполняется при удалении существующего программного пакета для службы.

Каталог /etc/init

Для пользователей дистрибутивов Linux, производных от Debian, каталог **/etc/init** используется для хранения сценариев Upstart. Эти сценарии запускают или останавливают разные службы на основе разных событий, включая переход на определенный уровень выполнения.

Для Debian и его производных (таких как Ubuntu) знайте, что используемые уровни выполнения немного отличаются от тех, которые определены в Linux Standard Base 4.1. Нулевой уровень запуска, один и шесть соответствуют стандарту. Тем не менее, второй уровень запуска считается уровнем запуска по умолчанию; этот уровень запуска настроен для нескольких пользователей с работающим графическим интерфейсом, очень похож на стандартный уровень запуска пять. Уровни запуска три, четыре и пять изначально совпадают с уровнем запуска два.

Если администратор хочет изменить уровни выполнения службы, файл конфигурации для этой службы можно изменить в каталоге **/etc/init**. Например, в установке Ubuntu, которая включает веб-сервер Apache, этот каталог обычно содержит файл конфигурации **/etc/init/apache2.conf** Upstart. В файле **/etc/init/apache2.conf** должно быть две строки, которые определяют уровни запуска для запуска и остановки сервера:

```
start on runlevel [2345]
```



```
stop on runlevel [!2345]
```

В этом случае служба будет запущена на уровнях выполнения со второго по пять и будет остановлена на уровнях выполнения, отличных от двух до пяти, потому что! означает "не это". Чтобы служба была доступна только на уровнях запуска два и три, измените строки следующим образом:

```
start on runlevel [23]
```

```
stop on runlevel [!23]
```

Чтобы отключить службу без ее удаления, можно создать файл переопределения в каталоге / etc / init. Этот файл должен иметь то же имя, что и файл конфигурации службы, но заканчиваться на .override вместо .conf. Этот метод предпочтительнее, чем комментирование строк «начало».

Содержимое файла .override должно быть просто словом manual, что означает, что служба будет игнорировать любые строки «start on» из файла конфигурации. Например, чтобы переопределить файл конфигурации apache2 и отключить веб-сервер, выполните следующую команду:

```
echo manual > /etc/init/apache2.override
```

Команда systemctl

Команда systemctl используется в системах, в которых Systemd заменяет традиционный процесс инициализации. Эта одна команда может использоваться для ручного управления состоянием служб, включения или отключения автоматического запуска служб, а также для изменения системных целей.

Чтобы вручную контролировать состояние службы, используйте команду systemctl для запуска, остановки или проверки состояния этой службы. Например, чтобы запустить сервис, такой как веб-сервер, выполните следующее:

```
systemctl start httpd.service
```

Чтобы закрыть сервис:

```
[root @ localhost ~] # systemctl stop httpd.service
```

Чтобы проверить состояние услуги:

```
[root @ localhost ~] # systemctl status httpd.service
```

Для просмотра статуса всех услуг:

```
[root @ localhost ~] # systemctl -a
```

или же:

```
[root @ localhost ~] # systemctl --all
```

Чтобы настроить автоматический запуск службы, выполните следующие действия:

```
[root @ localhost ~] # systemctl включить httpd.service
```

Чтобы настроить службу на автоматический запуск, выполните следующее:

```
[root @ localhost ~] # systemctl отключить httpd.service
```

Как упоминалось ранее, можно перейти на другой уровень выполнения с помощью команды **systemctl isolate graphical.target**. Команда **systemctl** также может управлять состояниями системы с низким или нулевым энергопотреблением с помощью таких командных строк, как: **systemctl hibernate**, **systemctl suspend**, **systemctl poweroff** и **systemctl reboot**.

При включении службы с помощью **Systemd** путем выполнения команды, такой как:

```
systemctl enable named.service
```

.... на целевом уровне создается символическая ссылка, которая «хочет» запустить эту службу. В этом примере предыдущая команда `systemctl` выполняет следующую команду:

```
ln -s /usr/lib/systemd/system/named.service  
/etc/systemd/system/multi-user.target.wants/
```

Причина, по которой `multi-user.target` «wants» запустить `named.service`, основана на строке в файле `named.service`, которая содержит следующее:

```
WantedBy = multi-user.target
```

По сути, эта строка устанавливает единственную цель, которая обычно запускает эту службу автоматически. Если администратор изменяет цель, для которой установлен `WantedBy`, то в следующий раз, когда эта служба будет включена, будет установлена символическая ссылка, которая включает службу, на новую цель, которая «wants» эту службу.

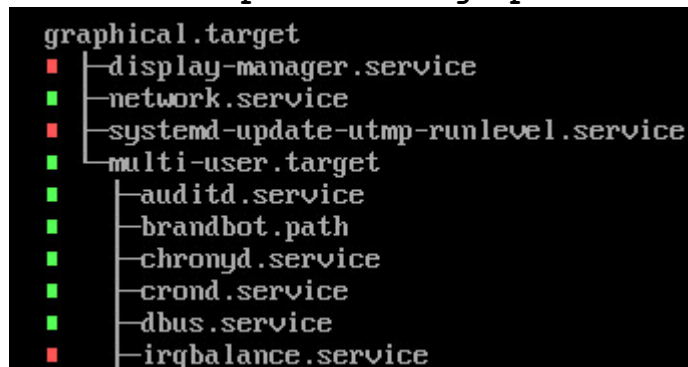
Например, если строка для `named.service` в файле `/usr/lib/systemd/system/named.service` обновится и будет выглядеть следующим образом:

```
WantedBy = graphical.target
```

Затем, после выполнения команд `systemctl disable named.service` и `systemctl enable named.service`, создается ссылка для запуска именованной службы в каталог `/etc/systemd/system/graphical.target.wants`, и служба запускается, когда Система переходит к `graphical.target` вместо `multi-user.target`.

Подобно команде `chkconfig --list`, все службы, которые должны быть включены для конкретной цели в Systemd, можно просмотреть с помощью команды `systemctl list-dependencies` для этой цели, такой как:

```
systemctl list-dependencies graphical.target
```



```
graphical.target
├─display-manager.service
├─network.service
├─systemd-update-utmp-runlevel.service
├─multi-user.target
│   └─auditd.service
│       └─brandbot.path
│           └─chronyd.service
│               └─crond.service
│                   └─dbus.service
└─irqbalance.service
```

Приведенный выше частичный вывод показывает каждый уровень требуемых сервисов ниже цели и зависимости между каждой целью с отступом. Вы видите, что такие сервисы, как `display-manager.service` и `network.service`, нужны графическому объекту. Также `graphical.target` зависит от `multi-user.target`, а `multi-user.target` хочет, чтобы сервисы `auditd.service` и `crond.service`. Фактический результат показывает гораздо больше услуг и целей.

Поскольку существует три разных типа загрузочных систем: традиционный **init**, **Upstart** и **Systemd**, логический вопрос: «какой из них использует моя система?». Простой ответ на этот вопрос - проверить наличие двух каталогов: `/etc/init` и `/etc/systemd`.

Если в вашей системе есть каталог `/etc/init`, то ваша система использует Upstart. Если в вашей системе есть каталог `/etc/systemd`, то ваша система использует Systemd. В противном случае ваша система использует традиционный `init`.

systemd — подсистема инициализации Linux — демон для запуска других демонов в Linux и управления ими в процессе работы системы, разработанная взамен используемого ранее демона `init` в стиле System V. Его особенностью является интенсивное распараллеливание запуска служб в процессе загрузки системы, что

позволяет существенно ускорить запуск операционной системы. Название происходит от принятого в Unix добавления суффикса «-d» к демонам.

Upstart — система инициализации ОС, которая управляет запуском демонов в течение загрузки системы, их остановку, а также управляет ими во время работы системы. Основанная на событиях замена системы инициализации `init` в Unix и Linux системах.

Первоначально была разработана для дистрибутива Ubuntu, но затем стала использоваться и в других дистрибутивах Linux, например в Fedora (впоследствии была заменена `systemd`), как замена SysV `init`.

init (сокращение от англ. initialization — инициализация) — подсистема инициализации в Unix и ряде Unix-подобных систем, которая запускает все остальные процессы. Работает как демон и обычно имеет PID 1. Обычно (согласно Filesystem Hierarchy Standard) располагается по пути `/sbin/init`. Существуют отличия в организации работы подсистемы в операционных системах, ведущих родословную от System V и систем в стиле BSD.

Длительное время была основной подсистемой инициализации в Linux, пока не была в большинстве дистрибутивов заменена `systemd`. В Solaris 10 вместо `init` применяется Service Management Facility. В ряде Unix-систем применяются другие альтернативы `init`: Upstart, Runit, Daemontools, Launchd, Initng, OpenRC.

В процессе загрузки после инициализации ядра как правило запускается `/sbin/init` как первый процесс пользовательского режима, и `init` отвечает за дальнейшую загрузку системы. Для этого запускаются стартовые сценарии, которые выполняют проверку и монтирование файловых систем, запуск необходимых демонов, настройку ядра (в том числе загрузку модулей ядра согласно установленному оборудованию, настройку IP-адресов, таблиц маршрутизации и другие задачи), запуск графической оболочки. Основная информация для загрузки как правило размещается в `/etc/inittab`.