

Все темы



 Space Police 21 янв в 2021  29K

[Подписаться на автора](#)

Cron: что это такое и как его правильно использовать



Обсудить



Администрирование

#Серверы #Настройка #Разбор #OpenSource



14 мин. чтения

Разбираемся с тем, что представляет собой Cron, как он работает и зачем нужен. Пытаемся обуздать его параметры, чтобы планировать задачи на своем сервере без лишних затрат сил и времени.

Что такое Cron и crontab?

Если в двух словах, то Cron – это планировщик задач. Если подробнее, то это утилита, позволяющая выполнять скрипты на сервере в назначенное время с заранее определенной периодичностью.

К примеру, у вас есть скрипт, который собирает какие-либо статистические данные каждый день в 6 часов вечера. Такие скрипты называют «заданиями», а их логика описывается в специальных файлах под названием crontab.

crontab – это таблица с расписанием запуска скриптов и программ, оформленная в специальном формате, который умеет считывать компьютер. Для каждого пользователя системы создается отдельный crontab-файл со своим расписанием. Эта встроенная в Linux утилита доступна на низком уровне в каждом дистрибутиве.

В Linux-дистрибутивах с поддержкой systemd Cron считается устаревшим решением, его заменили утилитой systemd.timer. Ее предназначение и функциональность не отличается, но фактически частота использования Cron все еще выше.

Для чего обычно используют Cron?

Обычно Cron заставляют повторять вполне очевидные задачи в духе регулярного создания резервных копий данных. Но это не все.

1. Некоторые пользователи с помощью планировщика корректируют системное время. На многих компьютерах оно настраивается через Network Time Protocol. А так как этот протокол настраивает только время ОС, время, установленное для «железа», может отличаться. Cron позволяют регулярно корректировать время, установленное для аппаратного обеспечения, в соответствии со временем ОС.
2. Еще один популярный сценарий – создание оповещений, появляющихся каждое утро и рассказывающих о состоянии компьютера. В эти сообщения может входить любая полезная для пользователя информация.
3. Cron иногда работает даже без ведома пользователя. Эту утилиту используют такие сервисы, как Logwatch, logrotate и Rootkit Hunter. Повторяющиеся задачи они настраивают, как и пользователи, через Cron.

С помощью Cron пользователи автоматизируют самые разные задачи, сокращая вмешательство системного администратора в работу сервера.

Базовые принципы работы с Cron и crontab (через панель управления)

Многие хостинг-провайдеры предлагают отдельное меню в панели управления для настройки расписания запланированного выполнения скриптов.

Разберем подобное меню на примере панели управления Timeweb. Чтобы создать новую задачу, необходимо открыть раздел Crontab в боковой панели веб-интерфейса, кликнуть по кнопке «Добавить новую задачу» и указать параметры повторяющейся команды. Поговорим подробнее о параметрах.

Сначала надо придумать для команды название (подойдет любой текст без спецсимволов).

Затем указываем исполнителя (нужно выбрать, будет ли планировщик работать с исполняемым файлом, PHP-скриптом или HTTP-запросом).

В графе «Путь до файла» вводим абсолютный путь до скрипта, запуск которого хотим запланировать. К примеру: /home/u/myusername/mytestscript.php. При желании можно воспользоваться встроенным файловым менеджером и выбрать заранее предзагруженный на сервер скрипт.

После этого указываем периодичность выполнения выбранного скрипта или исполняемого файла (в списке доступны предустановки в духе «Каждую минуту» или «Раз в день», но можно выбрать и пункт «Продвинутые настройки»).

Кликаем по кнопке «Создать задачу».

На этом все. Скрипт запланирован и будет регулярно повторяться.



Базовые принципы работы с Cron и crontab (через SSH-протокол)

Планировать задачи через панель управления удобно, но не всегда возможно. Не все хостинг-провайдеры предлагают такие функциональные веб-интерфейсы. В этом случае придется воспользоваться командной строкой, подключившись к серверу по протоколу Secure Shell.

Для работы с планировщиком в системе есть ряд команд, помогающих решать основные задачи:

`crontab -e` – открывает конфигурационный файл (поговорим о нем чуть подробнее в разделе с первичной настройкой).

`crontab -l` – показывает список задач из конфигурационного файла (все, что было запланировано).

`crontab -r` – удаляет конфигурационный файл вместе со всеми запланированными задачами.

`crontab -v` – показывает, когда в последний раз открывался конфигурационный файл.

Чтобы запланировать задачи, используя командную строку, необходимо выполнить базовую настройку Cron, проверить, не установлены ли ограничения, и заполнить расписание задач в соответствии с синтаксисом `crontab`.

Первичная настройка Cron

Как мы уже выяснили ранее, планировщик черпает параметры для выполнения своих задач из `crontab`-файлов (таблиц с расписанием). У каждого пользователя, включая `root`, должен быть свой `crontab`-файл. По умолчанию он не существует, поэтому придется создать его вручную.

Для этого существует команда **`crontab -e`**. Она автоматически генерирует таблицу в директории `/var/spool/cron`.

Вновь созданный файл будет пустым текстовым полем. Необходимо добавлять в него все параметры самостоятельно с нуля, опираясь на синтаксис `crontab` (более подробно поговорим о нем ниже). После ввода параметров нужно сохранить параметры редактора, нажав на клавишу `F2`, а затем покинуть конфигурационный файл, нажав на клавишу `F10`. При введении корректных параметров в терминале отобразится строка **`crontab: installing new crontab`**.

Опытные разработчики и системные администраторы не рекомендуют использовать для редактирования расписания текстовые редакторы в духе Nano, Emacs или Vi. Команды `crontab` позволяют не только внести изменения в таблицу запланированных задач, но и перезапустить фоновый процесс `cron`d, отвечающий за работу утилиты после сохранения настроек.

Ограничения Cron

У Cron есть функция установки ограничений на использование, задающихся через два специальных файла: `cron.allow` и `cron.deny`.

Первый файл находится в директории `/usr/lib/cron/cron.allow` и содержит в себе список учетных записей (имен пользователей), которые имеют право на планирование задач с помощью встроенных системных утилит.

Второй файл находится в директории `/usr/lib/cron/cron.deny`. В нем указываются имена пользователей, которые не могут запускать встроенный в систему планировщик задач.

Если первого файла не существует, то любой пользователь может планировать задачи с помощью встроенного в систему планировщика, но только при условии, что его имени нет во втором файле. Если удалить оба файла, то каждый пользователь сможет планировать задачи без ограничений.

Синтаксис `crontab`

```
# crontab -e
SHELL=/bin/bash
MAILTO=mymail@someprovider.com
PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin

# Детали смотрите в следующих разделах

# Примеры оформления задач в планировщике (формат данных):
# .----- минуты (0 - 59)
# | .----- часы (0 - 23)
# | | .----- дни месяца (1 - 31)
# | | | .----- сами месяцы (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- дни недели (0 - 6) (0 или 7 это воскресенье в зависи
# | | | | |
# * * * * * имя пользователя команда, которую нужно запустить

# создание копии всей операционной системы с помощью кастомного скрипта
01 01 * * * /usr/local/bin/bckp -vbd1 ; /usr/local/bin/bckp -vbd2

# установка соответствия между временем операционной системы и "железа"
03 05 * * * /sbin/hwclock --systohc

# проведение обновления операционной системы в заданный период времени
25 04 1 * * /usr/bin/apt-get update
```

Первые три линии кода в таблице отвечают за первичную настройку. Сначала указывается оболочка, в которой будет работать Cron. У утилиты нет каких-либо предпочтений, поэтому

можно указать любую на собственное усмотрение (в нашем примере это `bash`). Затем указывается адрес электронный почты, на который будут отправляться отчеты о работе планировщика. И напоследок указывается путь к окружению.

Ниже находятся параметры, используемые для запуска процессов в определенный период времени. В комментариях описан базовый синтаксис, включающий в себя формат времени, имя пользователя и команду, которая должна быть запущена.

В нашем случае указаны команды:

```
02 04 5 * * /usr/local/bin/bckp -vbd1 ; /usr/local/bin/bckp -vbd2
04 06 * * * /sbin/hwclock --systohc
10 05 5 * * /usr/bin/apt-get update
05 * * * * rm /home/myusername/tmp/*
```

Примеры использования Cron в командной строке

Команда

```
02 04 5 * * /usr/local/bin/bckp -vbd1 ; /usr/local/bin/bckp -vbd2
```

создает в таблице расписания задачу на запуск скрипта под названием `bckp` (представим, что такой существует), который создает резервную копию всей системы на стороннем накопителе. Он выполняется 5 числа каждого месяца в 4 часа 2 минуты утра. Это видно по числовым значениям. Звездочки же указывают на отсутствие конкретного значения. Cron воспринимает их как «выполнять каждый раз», то есть каждый месяц, день или неделю.

Команда

```
04 06 * * * /sbin/hwclock --systohc
```

меняет время аппаратного обеспечения на то, что используется в системе. Делает это каждый день, каждую неделю и каждый месяц в 6 часов 4 минуты утра. Как видите, здесь пропущено третье значение. Поэтому команда и запускается ежедневно, так как нет более конкретных правил.

Команда

```
10 05 5 * * /usr/bin/apt-get update
```

запускает обновление пакетов с помощью пакетного менеджера apt каждый месяц 5 числа в 05:10.

Команда

```
05 * * * * rm /home/myusername/tmp/*
```

удаляет содержимое папки с временными файлами для конкретного пользователя (меня) на пятой минуте (первый пункт) каждого часа. Так как определенные значения отсутствуют для всех остальных пунктов, получается, что скрипт готов выполняться каждый день, каждый месяц и каждый час. Но первое значение указано, поэтому он будет дожидаться пятой минуты и запускаться в этот момент. То есть в 12:05, 13:05, 14:05 и т.п.

Как видите, разобраться с базовыми командами несложно.

Другие примеры настройки Cron

На примере команды для удаления временных файлов разберем пару-тройку нестандартных настроек расписания.

К примеру, некоторые показатели можно вводить не в виде цифр, а в виде сокращенных слов. Чтобы запускать удаление временных файлов каждую пятницу в 4 часа вечера, необходимо ввести в crontab:

```
00 16 * * Fri rm /home/myusername/tmp/*
```

Если не указывать в планировщике точных вводных, то выбранный скрипт будет выполняться каждую минуту:

```
* * * * * rm /home/myusername/tmp/*
```

Можно заставить планировщик выполнять задачу только в определенные месяцы. Чтобы запускать удаление временных файлов 1 февраля, 1 мая и 1 сентября в половину первого ночи, необходимо ввести в crontab:

```
30 00 1 2,5,9 * * rm /home/myusername/tmp/*
```

Некоторые скрипты необходимо выполнять только по будням, поэтому в Cron есть возможность исключить некоторые дни недели из расписания:

```
00 16 * * 1-5 rm /home/myusername/tmp/*
```

Часы тоже можно делить. Например, запускать ту или иную задачу каждые 3 часа. Для этого нужно поставить слэш в графе с установкой времени по часам:

```
*/5 */2 * * * rm /home/myusername/tmp/*
```

Вместо заключения

О работе с Cron стоит знать еще пару важных вещей.

Во-первых, всегда указывайте корректный почтовый адрес в параметрах. Это позволит собирать информацию о запусках задач по расписанию. В этих письмах содержится информация о возникающих ошибках. Во-вторых, при указании «исполнителя» в панели управления Timeweb важно делать корректный выбор, чтобы он соответствовал запуске задачи. В-третьих, информацию о работе Cron можно собирать в отдельный файл с помощью команд в духе:

```
30 18 * * * rm /home/myusername/tmp/* > /home/myusername/cronlogs/clean_t
```


На этом все. Следуйте инструкциям, не путайте порядок параметров и внимательно изучайте журнал ошибок, если что-то пойдет не так. После недолгой практики вы поймете, что работать с Cron не так уж и сложно!

#Серверы #Настройка #Разбор #OpenSource



12



Администрирование

echo -e "Все про серверы, сети, хостинг и еще раз серверы" >/dev/pts/0



Комментарии

Написать комментарий

Рекомендуем

Как облачные сервисы помогают компаниям развиваться даже в кризис

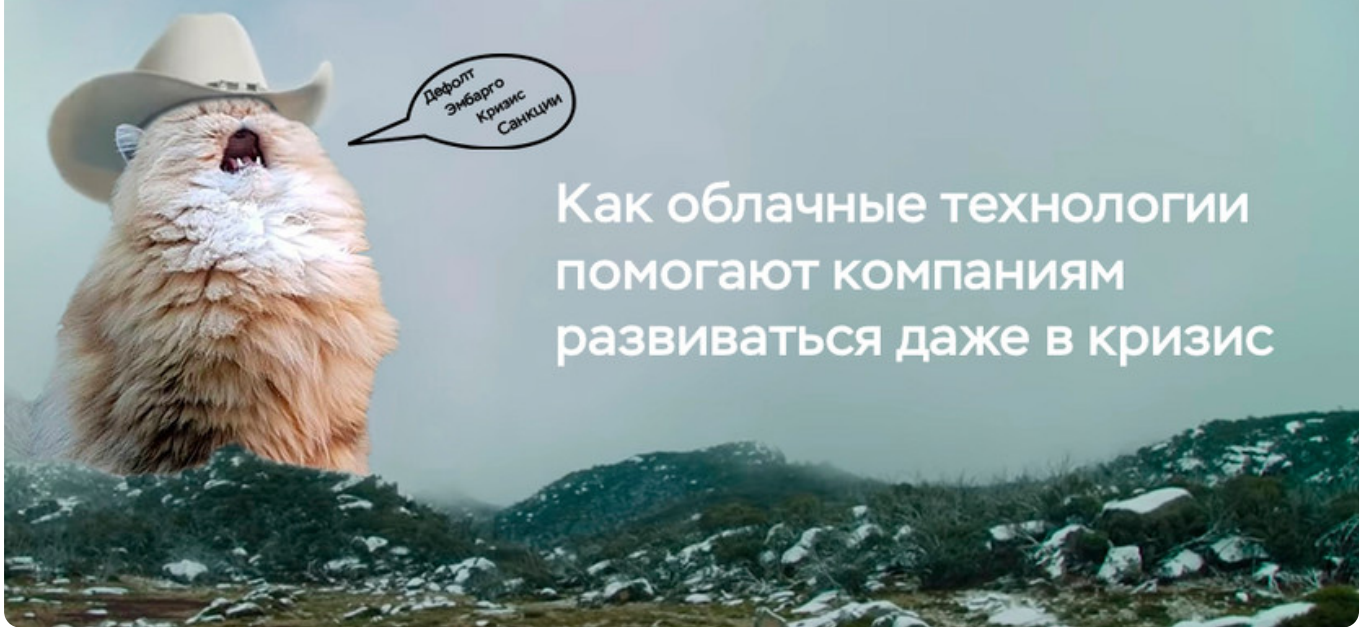
20 мая в 15:33



1207

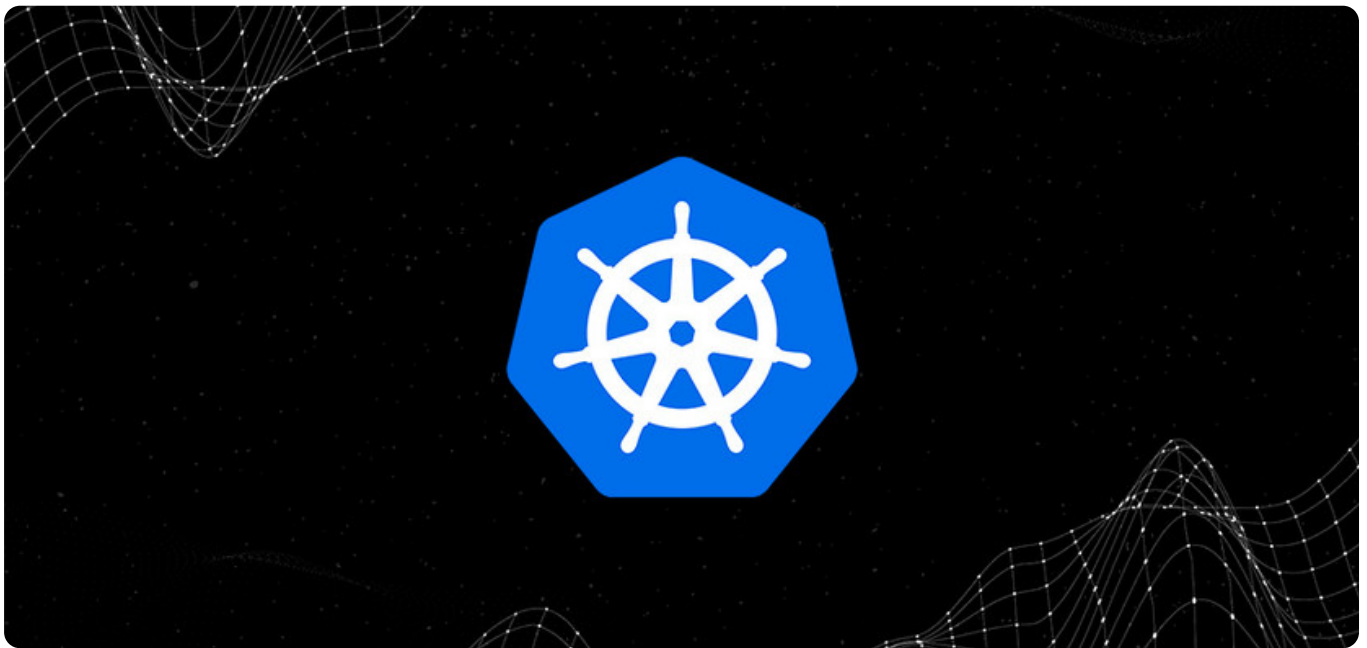


2



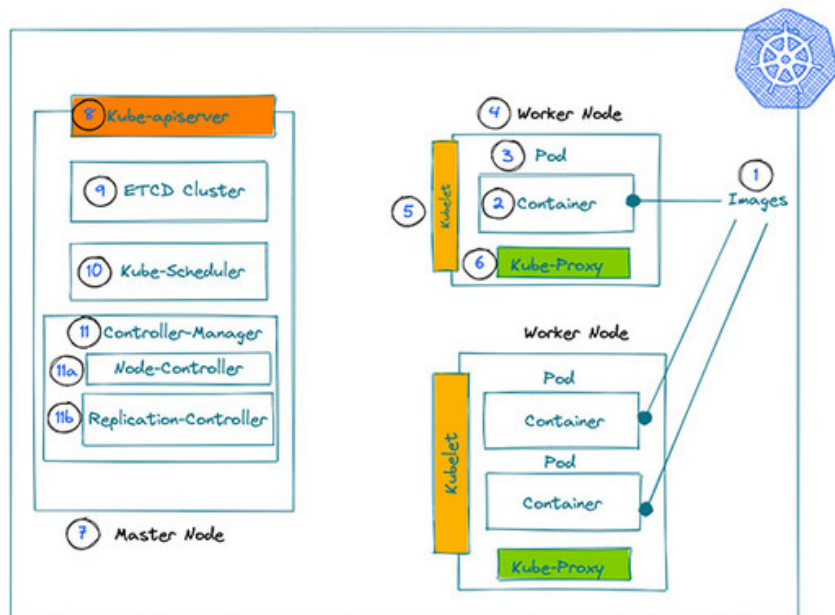
Все про облачные сервисы Kubernetes

04 мая в 13:44 👁 2213 💬 1





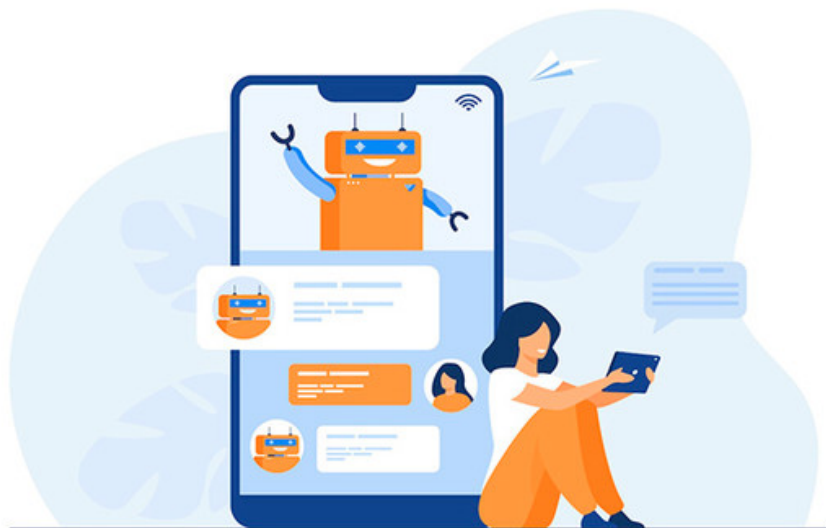
Основные понятия в Kubernetes

28 апр в 12:00 👁 2386 💬 0



Как выбрать хостинг для чат-бота

26 апр в 15:26  4294  0



Как создать интранет WordPress для совместной работы

21 фев в 13:27  2772  0

