О VPS на CentOS для сайта

Оглавление

Установка ОС и разметка диска

При установке операционной системы на VPS есть ряд аспектов, на которые надо обратить внимание:

Во-первых, долговременная память (жёсткий диск, SSD) VPS весьма ограничена. В рассматриваемом примере этого всего лишь 20 Гб.

Во-вторых, в процессе эксплуатации VPS весьма вероятно появление необходимости изменить размеры разделов этого диска без переустановки ОС. Так, например, если запросить у хостинг-провайдера дополнительное дисковое пространство на виртуальном сервере, то, в отличие от физического сервера, совсем не обязательно и не рационально ни выполнять переустановку системы, ни создавать дополнительный раздел во вновь выделенном дисковом пространстве, монтируя его к существующему дереву каталогов. Гораздо правильнее и экономичнее изменить

размер одного из разделов имеющейся системы, расширив его на вновь выделенное пространство. Возможна и обратная ситуация, в которой размер одного из разделов потребуется уменьшить.

Поэтому перед разметкой диска при установке системы следует ответить на ряд вопросов:

1. Где хранится содержимое (контент) сайта?
2. Сколько места можно выделить под содержимое сайта?
3. Где хранится база данных системы управления содержимым (CMS)?
4. Каков максимальный допустимый размер этой базы?
5. Разрешено ли пользователям прямо или косвенно увеличивать размер базы данных и содержимого сайта? Имеют ли пользователи возможность загружать свои файлы, оставлять комментарии, и т. д.?
6. Нужен ли раздел подкачки?

В данном случае (CentOS 7, Apache 2, MySQL/MariaDB, 20 Гб, учебно-демонстрационная задача) приняты следующие ответы на эти вопросы:

Контент сайта будет храниться в каталоге **/var/www** .

Под содержимое сайта следует выделить максимум дискового пространства, т. е. все дисковое пространство, которое останется с учетом требований других компонентов системы.

База данных CMS согласно конфигурационному файлу **/etc/my.cnf** , хранится в каталоге **/var/lib/mysql** .

Максимальный приемлемый размер этой базы не превышает 4 Гб.

Пользователи не имеют возможности прямо влиять на контент, загружая файлы, но могут оставлять комментарии, тем самым увеличивая занимаемое на диске пространство.

Несмотря на то, что использование раздела подкачки на строевой системе есть уже признак проблемы, здесь в демонстрационных целях такой раздел размером 1 Гб будет создан.

Наиболее оптимальной в данной задаче представляется использование следующей схемы разметки диска с DOS-таблицей разделов:

Точка монтирования	Размер	ФС	Назначение
/	4 Гб	ext4	корневая ФС, система, системные журналы (логи), загрузчик и т. д.
	1 Гб	swap	раздел подкачки
/var/lib/mysql	4 Гб	ext4	отдельный том для базы данных CMS

/var/www	> 10 Гб	ext4	содержимое сайта

Необходим ряд пояснений:

Таблица разделов MSDOS выбрана вместо LVM, так как преимущества LVM не играют в этой ситуации никакой роли, а поэтому правильнее выбрать проверенную временем минималистичную классику.

Для корневой файловой системы выбран столь небольшой размер в силу того, что и сам виртуальный диск невелик. С другой стороны, для установки системы большой том и не нужен: X-сервер не будет установлен, дополнительного программного обеспечения будет установлен самый минимум, содержимое сайта и его база данных вынесены в отдельные тома. К тому же, при такой схеме разметки всегда возможно удалить раздел подкачки и расширить корневую файловую систему за его счет.

В качестве файловой системы для всех томов выбрана ext4 как надежная, проверенная временем, функциональная с большим количеством расширений, легко масштабируемая в обе стороны (в отличие от xfs).

Разбивка диска на несколько томов имеет свои достоинства и недостатки. Главным недостатком является потеря дискового пространства на служебную информацию нескольких файловых систем. Однако, эти потери совсем невелики. Главным достоинством такого подхода является разграничение дискового пространства жестко и без использования системы квот. Допустим, на сайт производится атака, направленная на переполнение файловой системы. В этом случае ущерб будет локализован. В случае атаки через MySQL будет переполнен лишь том, примонтированный в **/var/lib/mysql** . В случае атаки через веб-сервер – лишь том, примонтированный в **/var/www** . Разумеется, если эти службы выполняются от имени разных пользователей (а не от root обе, например), то того же самого можно было бы добиться использованием квот. Однако обработка квот наряду с обработкой границ разделов – это пусть и небольшая, но все же дополнительная нагрузка на процессор. При этом (что описано в соответствующем пункте) для корневой файловой системы квоты все же будут введены, так как скомпрометированный сервис может попытаться переполнить не только свой рабочий каталог, но и любой доступный ему для записи, например /tmp, расположенный при данной схеме разметки как раз в корневой ФС.

Также можно рассматривать вопрос о выделении отдельного тома для системных журналов и журналов служб. В таком случае смонтировать его целесообразно в каталог **/var/log** . Дело в том, что журналы могут быть использованы как средство атаки на переполнения файловой системы. В разделе «Журналирование» это описано подробнее. В рассматриваемой задаче такой подход представляется избыточным. С той же целью можно выделить отдельный том и для временных файлов (**/tmp**).

Также при установке CentOS создаются пользователи: суперпользователь (root) и непривилегированный пользователь. В целях повышения уровня безопасности удаленный вход в систему по протоколу ssh от имени суперпользователя будет в дальнейшем отключен. И как раз учетная запись непривилегированного

пользователя и будет использоваться для этих целей. В связи с этим логин и пароль такого пользователя следует выбирать запоминающимся, но трудным для угадывания и словарного подбора. В таком случае злоумышленнику придется подбирать не только пароль, но и логин, и, даже преуспев в этом, он обнаружит, что проник в систему с правами обычного пользователя, а затем ему еще придется подбирать и пароль суперпользователя. В сочетании с некоторыми описанными далее мерами (fail2ban, отключение sudo, настройка сервера ssh) это может стать практически непреодолимой преградой для попыток проникновения подобного рода.

В качестве примера здесь приняты следующие имена и пароль пользователей:

Логин (имя пользователя)	Пароль
root	CentOS7
zubastad	InVincible_13

После создания пользователей и разметки диска остается лишь дождаться окончания установки системы и перезагрузки сервера.

Первичная настройка после установки

Прежде всего после установки системы следует задать имя хоста (если это не было сделано на этапе установки).

Пусть в рассматриваемой задаче имя хоста mysite.ru, а ip-адрес 192.168.0.99.

Для этого следует в строго указанном далее порядке сделать следующее:

1. Определиться с FQDN сервера.

2. Изменить старое FQDN сервера на новое во всех его вхождениях в файле **/etc/hosts** , если такие вхождения там присутствуют. В рассматриваемой системе таких вхождений нет, в этом файле присутствуют лишь значения по умолчанию для локальных петлевых интерфейсов.

3. Изменить старое FQDN сервера на новое в файле **/etc/hostname** . Этот файл должен содержать единственную строку, заканчивающуюся символом новой строки и содержащую FQDN-сервера.

4. Выполнить команду изменения имени для текущей сессии (здесь и далее команды, начинающиеся с приглашения командной строки # следует выполнять с привилегиями суперпользователя):

```
# hostname mysite.ru
```

Затем можно приступать к настройке сети. Сначала необходимо узнать имя сетевого интерфейса. Для этого достаточно выполнить команду:

```
# ip link show
```

На рассматриваемой системе имя сетевого интерфейса enp0s3. Далее следует отредактировать файл **/etc/sysconfig/network-scripts/ifcfg-enp0s3** и привести его к следующему виду (взаимный порядок строк произволен, строки, автоматически

добавленные в этот файл и не указанные ниже удалять не обязательно и даже не желательно, нужно лишь чтобы в итоге файл не противоречил приведенным параметрам):

```
IPADDR=«192.168.0.99»
GATEWAY=«192.168.0.1»
NETMASK=«255.255.255.0»
BOOTPROTO=«static»
DEVICE=«enp0s3»
ONBOOT=«yes»
IPV6INIT=«yes»
DNS1=8.8.8.8
DNS2=8.8.4.4
ZONE=public
```

При необходимости или желании использовать другие DNS-серверы, следует заменить текущие адреса (8.8.8.8, 8.8.4.4) на желаемые.

Отдельно в разделе «Описание файервола» будет сказано об отключении протокола IPv6, что не является обязательным.

Используя эти параметры, сценарии настройки сети CentOS 7 автоматически сгенерируют файл настройки DNS-резолвера **/etc/resolv.conf** .

На этом базовая настройка сети завершена. Остается лишь перезапустить службу сети и проверить параметры:

```
# service network restart
# service network status
# ip addr show
# ip route show
# cat /etc/resolv.conf
# hostname -f
```

Здесь уместны следующие замечания:

Для некоторого повышения безопасности сервера можно использовать файлы **/etc/hosts** , **/etc/hosts.allow** и **/etc/hosts.deny** .

Файл **/etc/hosts** служит локальной базой DNS-записей и имеет высший приоритет: ни одна его запись не может быть переопределена данными, полученными от внешних DNS-серверов. Таким образом, внесение сопоставлений ip-адреса и dns-имени в этот файл позволяет одновременно сократить число обращений к внешним DNS-серверам и защититься от подмены DNS-ответов (атака DNS-poisoning). Такое решение особенно важно, если информационная система состоит из нескольких взаимодействующих между собой серверов. В рассматриваемой системе, состоящей из единственного сервера, такое решение не дает существенно ничего. Тем не менее имя самого сервера сопоставить его ip-адресу будет явно не лишним. Для этого (на рассматриваемой системе) следует внести в файл **/etc/hosts** следующую строку:

```
192.168.0.99    mysite.ru
```

Файлы **/etc/hosts.allow** и **/etc/hosts.deny** являются конфигурационными файлами posix-acl – одного из старейших и базовых рубежей защиты posix-подобных систем (acl – access control list, список контроля доступа). В них можно явно указать какие хосты (как по имени, так и по адресу) могут подключаться к каким сервисам на

рассматриваемом сервере. Так, например, имея статический ip-адрес компьютера администратора сайта можно создать еще один рубеж защиты: север не примет ssh-соединение ни от одного хоста, кроме явно указанного компьютера системного администратора, причем не зависимо от поведения межсетевого экрана. Другим примером (в случае распределенной, т. е. работающей на нескольких серверах, системы) может быть настройка, явно указывающая серверу баз данных список всех его клиентов (веб-серверов). В таком случае нарушитель не сможет получить доступ к базе никак кроме как через взлом одного из клиентов. В рассматриваемой системе это едва ли применимо, если системный администратор не имеет статического ip-адреса.

Первым делом после настройки сети следует обновить систему. Для этого необходимо обновить сведения о доступных пакетах:

```
# yum check-update
```

затем установить все обновления командой

```
# yum update
```

Для удобства работы не будет лишним установить файловый менеджер mc:

```
# yum install mc
```

Учитывая ограниченность дискового пространства и в целях соблюдения «чистоты» системы, полезно удалить все пакеты, ставшие ненужными после обновления (если такие есть) командой:

```
# yum autoremove
```

а также очистить кэш пакетов:

```
# yum clean packages
```

По умолчанию в составе CentOS 7 присутствует и активирован SELinux. Для рассматриваемой системы он представляется избыточным. SELinux способен защитить систему при компроментации даже корневого процесса веб-сервера, работающего от имени суперпользователя. Однако в этом случае работа информационной системы парализована или взята под контроль злоумышленником, а ценность операционной системы стремится к нулю, т. к. в случае такой компрометации рекомендуется полная переустановка системы и восстановление данных из резервных копий. В силу вышесказанного вполне допустимо отключить SELinux.

Для проверки текущего состояния SELinux следует выполнить команду:

```
# sestatus
```

Отключение SELinux выполняется в конфигурационном файле **/etc/selinux/config**:

```
SELINUX=disable
```

Замечание: на только что установленной системе этот файл имеет вторую жесткую ссылку /**etc/sysconfig/selinux**, т. е. изменить это значение, как может показаться, можно и в нем. Однако при обновлении системы был замечен следующий эффект:

Обновление SELinux привело к пересозданию файла /etc/selinux/config и, тем самым, разрыву связи со ссылкой. Т. е. в системе образовалось два разных файла, а SELinux

прочитал при запуске вновь созданный. В итоге SELinux включился и заблокировал работу ssh на измененном порту.

В завершение базовых настроек системы следует проверить состояние локализации командой

```
# localectl
```

Наиболее универсальным на рассматриваемых системах выглядит использование кодировки UTF-8 в англоязычной (американоязычной) кодировке. Чтобы ее установить, достаточно выполнить команду

```
# localectl set-locale LANG=en_US.UTF-8
```

Настройка сервера SSH

В CentOS по умолчанию уже установлен сервер openssh. В целях повышения уровня безопасности системы можно несколько изменить его настройки по умолчанию.

Во-первых, следует изменить порт, на котором openssh принимает соединения. Само по себе это не повышает уровень безопасности, однако потенциальному нарушителю придется выяснять номер этого порта, что, в зависимости от степени осторожности нарушителя, может немного или значительно увеличить время подготовки к атаке. К тому же, хостинг-провайдер вряд ли останется безучастным к сканированию портов одного из поддерживаемых им VPS.

Во-вторых, соответственно задаче, следует изменить ряд внутренних параметров сервера ssh. В частности, следует запретить поддержку протокола X11 (так как X-сервер все равно не установлен), запретить вход в систему по ssh от имени root, явно указать учетные записи, от имени которых разрешено входить в систему по ssh и запретить использование пустых паролей.

Для достижения этих настроек следует в первую очередь привести конфигурационный файл **/etc/ssh/sshd_config** к такому виду, чтобы он содержал (в произвольном порядке) следующие строки (в контексте рассматриваемой задачи):

```
Port 33631
X11Forwarding no
PermitRootLogin no
AllowUsers zubastad
PermitEmptyPasswords no
```

Сделать это можно при помощи удобного текстового редактора mcedit, входящего в состав пакета mc:

```
# mcedit /etc/ssh/sshd_config
```

Затем следует сообщить межсетевому экрану (файерволу, брандмауэру), что рабочий порт ssh изменился.

CentOS использует по умолчанию межсетевой экран iptables опосредованно через службу firewalld. С одной стороны такой посредник может показаться избыточным, особенно тем, кто привык работать с iptables напрямую. С другой стороны, firewalld появился как следствие опыта эксплуатации iptables в CentOS на наиболее

популярных задачах. И хотя вся гибкость firewalld едва ли сможет проявиться в рассматриваемом случае, из соображений следования стилю сообщества CentOS далее рассматривается настройка iptables именно посредством firewalld.

Для изменения номера порта службы ssh средствами firewalld необходимо отредактировать файл **/usr/lib/firewalld/services/ssh.xml** . В этом файле следует найти номер порта (по умолчанию 22) и заменить его на тот же номер, что был внесен в конфигурационный файл сервера ssh. Помнить о выполнении такой замены следует и при обновлении системы: не исключено, что при обновлении systemd эта подмена может быть «затерта». Поэтому после обновления системы следует проверять правильность состояния системы до разрыва соединения с ней.

После внесения этих изменений следует перезапустить службы межсетевого экрана и сервера ssh а также убедиться в их работоспособности.

Внимание! В случае ошибок при внесении указанных выше изменений или при перезапуске служб велика вероятность того, что система станет недоступна по протоколу ssh. Поэтому подобные правки желательно вносить имея подключение к серверу иным способом, например через систему виртуализации хостинг-провайдера по протоколу spice или vnc.

Перезапуск служб выполняется, соответственно, командами:

```
# service firewalld restart
# service sshd restart
```

Для тех же целей возможны альтернативные команды:

```
# systemctl firewalld restart
# systemctl sshd restart
```

Проверка состояния служб после перезапуска может быть выполнена соответственно командами:

```
# service firewalld status
# service sshd status
```

или

```
# systemctl status firewalld
# systemctl status sshd
```

Просмотреть правила, созданные firewalld для iptables можно, например, так:

```
# iptables-save | less
```

В выводе этой команды следует удостовериться в наличии правила, разрешающего установление с системой tcp-соединений на выбранном для ssh порту (для выхода из программы просмотра файлов достаточно нажать кнопку «q»).

Также при помощи команды

```
# ss -l | grep LISTEN | grep tcp
```

можно увидеть, что система «слушает» порт 25 на себе (localhost) по ipv4 и ipv6 и порт 33631 по tcp отовсюду по ipv4 и ipv6. Согласно **/etc/services** порт 25 закреплен за почтовым протоколом smtp. В текущих настройках этот протокол используется исключительно внутри системы и на данном этапе не представляет угрозы

безопасности, однако при описании настройки пользователей о почтовой подсистеме будет высказано несколько замечаний.

Теперь для подключения по ssh с удаленного компьютера следует использовать команду вида:

```
# ssh -p 33631 zubastad@mysite.ru
```

При этом даже после ввода правильного пароля при попытке подключения по ssh от имени суперпользователя

```
# ssh -p 33631 root@mysite.ru
```

соединение отклоняется.

Смежным с настройкой ssh вопросом является контроль за другими способами удаленного подключения к системе. Чаще всего это telnet и rsh. Как было показано при выводе открытых портов системы, эти службы на рассматриваемом сервере неактивны по умолчанию (их порты закрыты). При помощи команд

```
# which rsh
# which telnet
```

можно убедиться, что они не только не активны, но и не установлены вообще.

Настройка пользователей

При установке системы были созданы два пользователя: root и zubastad. При этом пользователь zubastad был указан как администратор, в следствие чего он был добавлен в системную группу wheel, которой в конфигурационном файле **/etc/sudoers** даны особые привилегии (использование sudo). Дополнительной мерой безопасности можно считать подход, в котором у обычного пользователя нет таких привилегий и для выполнения административных команд он должен использовать не команду sudo с вводом своего пароля, а команду su с вводом пароля суперпользователя. Таким образом на пути злоумышленника встает еще одна преграда: ему не достаточно войти в систему под именем некоторого пользователя (здесь zubastad), а необходимо еще и знать пароль суперпользователя.

Для реализации этого подхода рационально еще на этапе установки не делать пользователя администратором системы. В противном случае пользователя можно исключить из группы wheel.

Для этого требуется сначала войти в систему от имени пользователя (zubastad), выяснить в каких группах он состоит, затем от имени root исключить пользователя из группы wheel, затем войти в систему повторно под именем пользователя и убедиться, что команда su ему доступна, а команда sudo – нет. Далее приведен фрагмент терминального сеанса, демонстрирующий выполнение этих операций и отклик системы на них (следует помнить что команды, начинающиеся с приглашения командной строки # следует выполняются от имени root):

```
$ groups
zubastad wheel
$ ls /root
```

```
ls: невозможно открыть каталог /root: Отказано в доступе
$ sudo ls /root
anaconda-ks.cfg
$ su
Пароль:
# usermod -G zubastad zubastad
# exit
$ groups
zubastad wheel
$ exit
```

После повторного входа в систему:

```
$ groups
zubastad
$ ls /root
ls: невозможно открыть каталог /root: Отказано в доступе
$ sudo ls /root
[sudo] пароль для zubastad:
zubastad is not in the sudoers file.  This incident will be reported.
$ su
Пароль:
# ls /root
anaconda-ks.cfg
# exit
```

Еще одним возможным направлением атаки является локальная почта. В имеющейся на данный момент конфигурации почтовый сервер не принимает сообщения из сети вообще, но выполняет доставку писем пользователям внутри системы. Потенциально это открывает следующее направление для атаки: добиться возможности отправки письма внутри системы (войдя в систему как zubastad, воспользовавшись уязвимостями в mysql и/или apache, MX-injection и т. д.), исчерпать доступное дисковое пространство в корневой файловой системе (при имеющейся разметке диска) путем локальной отправки большого количества спама.

Для предотвращения таких атак имеет смысл полностью отключить почтовую систему, если она не используется вообще или отказ от нее некритичен. Имеется в виду то, что некоторые компоненты системы (например cron, smartd, система управления пакетами) могут отправлять локальные сообщения для root. Если отключение почтовой подсистемы допустимо, то на рассматриваемой системе оно может быть выполнено следующим образом:

```
# systemctl stop postfix
# systemctl disable postfix
# ss -l | grep LISTEN | grep tcp
```

Последняя команда позволяет убедиться, что сетевой порт почтовой системы закрыт.

О службе точного времени (NTP)

Часто в различных руководствах упоминается о том, что необходимо настроить на сервере службу точного времени. Однако в данном случае (VPS) это излишне: виртуальный сервер пользуется теми же аппаратными часами, что и сервер виртуализации (хост), на котором он расположен. Т. е. в вопросах точного времени можно совершенно положиться на оборудование провайдера VPS.

О chroot-окружении

Часто в различных руководствах рекомендуется помещать потенциально подверженные взлому сервисы в chroot-окружение. Такой подход имеет как преимущества, так и недостатки, некоторые из этих недостатков критичны в рассматриваемой задаче. Главным преимуществом является то, что в случае перехвата управления сервисом (вплоть до внедрения в него исполняемого кода злоумышленника), весь нанесенный ущерб будет локализован единственным подкаталогом файловой системы сервера. Главным недостатком является дополнительный расход дискового пространства на организацию chroot-окружений. И это при том, что дисковое пространство в этой задаче весьма ограничено. Другим недостатком является относительная сложность развертывания и поддержания в актуальном состоянии (обновления) chroot-окружения для имеющего недостаточный опыт администратора. При этом не следует забывать, что в chroot-окружении окажутся не только исполняемые и конфигурационные файлы служб, но и обрабатываемые ими данные, и их журналы. Поэтому кроме перенастройки самой службы для работы в такой среде придется перенастраивать и взаимодействующие с ней службы.

Также с учетом мирового опыта использования chroot-окружений, современных тенденций развития подобных технологий и специфики рассматриваемой задачи, можно отметить следующее:

- chroot-окружения оправданы на системах, поддерживающих одновременно работу нескольких не взаимосвязанных или частично взаимосвязанных сервисов;
- прямым развитием такого подхода является выделение для каждой службы отдельного сервера (физического или виртуального);
- в рассматриваемой задаче сервисы тесно взаимосвязаны: падение одного из них автоматически означает бесполезность или неработоспособность другого.

В итоге следует вывод: использование подобных технологий в рассматриваемой задаче явно избыточно.

Настройка MariaDB

Настройку СУБД MariaDB можно разделить на следующие этапы:

- установка
- настройка собственных параметров безопасности mysqld
- настройка службы systemd
- активация службы systemd
- обеспечение взаимодействия с межсетевым экраном
- обеспечение взаимодействия с fail2ban

Прежде всего СУБД необходимо установить. Для этого достаточно выполнить команды:

```
# yum search mariadb
# yum install mariadb mariadb-server
```

В качестве зависимостей серверного пакета также будут установлены некоторые пакеты perl.

После установки служба по умолчанию не активирована. Вместо того, чтобы активировать ее с настройками по умолчанию разумно будет сначала изменить ее настройки на соответствующие задаче.

Для этого в составе дистрибутива mariadb есть сценарий автоматизированной настройки этих параметров, которым и следует воспользоваться при помощи команды:

```
# mysql_secure_installation
```

В процессе работы этого сценария будет предложено задать пароль суперпользователя СУБД (root). Надо понимать, что root операционной системы и root системы управления базами данных – это две разных учетных записи разных систем, сведения о которых (в том числе хэш-суммы паролей) хранятся в разных местах. Поэтому настоятельно рекомендуется и пароли для них задавать разные. Здесь, в качестве примера задан следующий root-пароль СУБД: Zl0DeiK0 .

Также этот сценарий заставит демон mysqld работать только на внутреннем петлевом интерфейсе, т. е. заблокирует возможность подключения к серверу баз данных извне, а также отключит возможность загрузки локальных файлов, т. е. загрузку файлов из файловой системы для пользователей без прав на уровне файлов в базе данных, что отразится в конфигурационном файле **/etc/my.cnf** в разделе [mysqld] следующими строками:

```
bind-address=127.0.0.1
local-infile=0
```

Затем, согласно задаче, необходимо обеспечить автоматический перезапуск службы mariadb в случае ее отказов. Для этого следует обратиться к соответствующей службе systemd, расположенной в файле **/usr/lib/systemd/system/mariadb.service** .

В этом файле указано, что во избежание потери изменений при обновлении системы все правки лучше делать в файлах в каталоге **/etc/systemd/system/mariadb.service.d** .

Однако при установке СУБД такой каталог не был создан автоматически, поэтому необходимо его создать вручную, создать там конфигурационный файл (здесь для примера он будет называться **local.conf**) и внести туда необходимые записи. Все это можно выполнить следующим набором команд:

```
# mkdir /etc/systemd/system/mariadb.service.d
# touch /etc/systemd/system/mariadb.service.d/local.conf
```

В этот файл достаточно внести такие строки:

```
[Service]
Restart=always
RestartSec=180
```

Вносить дополнительные параметры StartLimitInterval и StartLimitBurst не обязательно: их назначение – прервать цикл неудачных повторных запусков службы в случаях ее отказов, чтобы не нагружать сервер бесполезными попытками ее запуска. Однако выставленный здесь достаточно большой интервал между обнаружением отказа службы и попытки ее перезапуска (180 секунд, т. е. 3 минуты) и без того обеспечит разумное расходование вычислительных ресурсов. У такой задержки есть и другое назначение: в случае целенаправленной атаки на информационную систему путем обрушения службы СУБД (например через исчерпание свободной оперативной памяти) у злоумышленника может сложиться иллюзия успеха, т. к. в течение трех минут информационная система (ИС) будет в неработоспособном состоянии.

Несколько забегая вперед следует отметить, что аналогичные меры будут приняты и к веб-серверу, что в итоге приведет к построению модели возобновления в защите от атак на отказ обслуживания. Т. е. идея защиты системы состоит не в том, чтобы жестко разграничить ресурсы сервера между службами и максимально снизить вероятность отказа (модель предотвращения), а в том, чтобы обеспечить автоматическое восстановление работоспособности.

Такой выбор сделан по следующим соображениям:

- ресурсы системы ограничены, реализация мер по их жесткому разграничению снизит максимальную производительность системы;
- в силу характера функционирования ИС работоспособность СУБД при неработоспособности веб-сервера и наоборот не имеют практической значимости;
- максимальная рабочая нагрузка (количество клиентов в единицу времени) при отсутствии атак трудно прогнозируема.

После выполнения описанных настроек остается лишь активировать службу и проверить ее состояние и способность к автоматическому перезапуску. Активация и запуск службы выполняются командами:

```
# systemctl enable mariadb
# service mariadb start
```

Проверка статуса службы выполняется командой

```
# service mariadb status
```

Также следует проверить, какие порты и по каким протоколам используются этой службой:

```
# ss -l | grep LISTEN | grep tcp
```

В отклике системы на последнюю команду очевидно, что прослушивается только tcp-порт mysql на адресе 127.0.0.1 .

Последним этапом является умышленное выведение службы СУБД из строя с проверкой автоматического восстановления работоспособности. Это можно сделать, послав сигнал аварийного завершения процессу демона СУБД (не рекомендуется это делать уже на введенной в строй СУБД, содержащей практически значимые базы

данных):

```
# ps -ef | grep mysql
# kill -9 4829
```

Здесь первая команда позволяет выяснить PID (здесь на этом конкретном запуске PID = 4829) искомого процесса, а вторая «убивает» его.

В течение следующих трех минут служба должна быть неактивна, команда

```
# ps -ef | grep mysql
```

не должна находить соответствующий процесс, а проверка

```
# service mariadb status
```

должна выдавать сообщение о нерабочем состоянии службы.

Через три минуты служба должна восстановиться, т. е. те же команды

```
# ps -ef | grep mysql
# service mariadb status
```

должны вернуть новый PID вновь запущенного процесса и подтвердить рабочий статус службы.

Что же касается взаимодействия с межсетевым экраном и fail2ban, то никаких дополнительных настроек вносить не требуется, так как служба работает локально. Подробнее см. далее при описании файервола и fail2ban.

Настройка Apache

Настройка службы httpd во многом схожа с настройкой СУБД. В ее настройке также можно выделить следующие этапы:

- установка
- настройка собственных параметров безопасности httpd
- настройка службы systemd
- активация службы systemd
- обеспечение взаимодействия с межсетевым экраном
- обеспечение взаимодействия с fail2ban

Установка apache2 (известного в репозиториях CentOS как пакет httpd) выполняется командой:

```
# yum install httpd
```

Как и в случае mariadb, по умолчанию служба не активирована. Ее активацию и запуск разумно выполнить после настройки.

Apache2 – мощный функциональный и гибкий программный продукт. Описание его настроек безопасности с одной стороны едва ли может быть выполнено в рамках одной статьи, с другой стороны, такие меры безопасности должны проектироваться в тесной связи с проектированием информационной системы, которая будет размещена на рассматриваемом сервере. Поэтому здесь невозможно привести детальную инструкцию. Здесь возможен лишь ряд базовых рекомендаций,

применимых или неприменимых в зависимости от подлежащей реализации информационной системы.

Кроме того, многие параметры безопасности взаимосвязаны с производительностью сервера и требует настройки и корректировки по результатом тестовой и строевой эксплуатации сервера.

Основные меры защиты apache2:

- Регулярно устанавливать обновления.
- Скрыть версию apache и другую чувствительную информацию.
- Установить и настроить модуль mod_security.
- Правильно настроить одну или несколько учетных записей для apache. По умолчанию все процессы apache запускаются от имени одноименных пользователя и группы (apache). Создание дополнительных пользователей едва ли оправдано на рассматриваемой задаче, т. к. это имеет смысл только если на одном веб-сервере работает несколько информационных систем (сайтов).
- Убедиться, что файлы вне web-root не обслуживаются сервером. В конфигурации apache это может быть прописано, например, так:

```
<Directory />
      Order Deny, Allow
      Deny from all
      Options None
      AllowOverride None
</Directory>
```

- Выключить просмотр каталогов, например, так:

```
<Directory />
      Options -Indexes
</Directory>
```

разумеется, возможны и исключения, например репозитории.

- Выключить, если не используются SSI, CGI, переходы по символьным ссылкам:

```
Options -Includes
Options -ExecCGI
Options -FollowSymLinks
```

- Если возможно, отключить поддержку .htaccess, если невозможно, запретить скачивание всех файлов вида .ht* :

```
<Files ~ «^\.ht»>
      Order allow, deny
      Deny from all
      Satisfy all
</Files>
```

- Отключить все ненужные модули.
- Убедиться в доступности исполняемых и конфигурационных файлов apache только для пользователя root и группы root.
- Уменьшить время таймаута для защиты от DoS-атак:

```
Timeout 45
```

• Ограничить большие запросы (также для защиты от Dos-атак):

```
LimitRequestBody 1048576
```

• Также обратить внимание на параметры LimitRequestFields, LimitRequestLine, LimitXMLRequestBody .

• Ограничить число одновременных запросов при помощи параметров MaxClients, MaxSpareServers, MaxRequestsPerChild, ServerLimit, ThreadsPerChild, MaxSpareThreads .

• По возможности ввести ограничения по IP или сетям (едва ли применимо к публичному сервису).

• Настроить параметры HTTP MaxKeepAliveRequests и KeepAliveTimeout .

Настройка службы systemd для httpd практически полностью совпадает с настройкой для mariadb. Здесь также необходимо вручную создать конфигурационный подкаталог и файл в нем (здесь для примера он также будет называться **local.conf**) и внести туда необходимые записи. Все это можно выполнить следующим набором команд:

```
# mkdir /etc/systemd/system/httpd.service.d
# touch /etc/systemd/system/httpd.service.d/local.conf
```

В этот файл достаточно внести такие строки:

```
[Service]
Restart=always
RestartSec=180
```

Здесь в качестве примера выставлен такой же интервал перезапуска, как и для СУБД, однако они не обязательно должны всегда совпадать.

По завершении этих настроек службу можно активировать и проверять на способность к восстановлению:

```
# systemctl enable httpd
# service httpd start
# service httpd status
# ss -l | grep LISTEN | grep tcp
# ps -ef | grep httpd
```

Для проверки требуется «убить» процесс командой вида

```
# kill -9 5217
```

(где 5217 – PID одного из процессов httpd) все процессы httpd и после этого по истечении трех минут проверить состояние службы командами:

```
# service httpd status
# ps -ef | grep httpd
```

Что касается взаимодействия с межсетевым экраном, то здесь важно отметить следующее:

Во-первых, на рассматриваемой системе активация службы httpd не приводит к автоматическому открытию используемых ею портов, что можно увидеть по выводу команд:

```
# ss -l | grep LISTEN | grep tcp
```

```
# iptables-save | less
```

Первая команда отображает тот факт, что httpd принимает соединения на своих рабочих портах, а вторая – что эти порты не открыты на файерволле. Для того, чтобы средствами firewalld открыть эти порты (в предположении, что используются порты 80/tcp (http) и 443/tcp (https)), необходимо выполнить следующие команды:

```
# firewall-cmd --permanent --add-service=http
# firewall-cmd --permanent --add-service=https
```

Проверить состояние портов можно, например, так:

```
# iptables-save | less
```

Дисковые квоты

При установке системы было сказано об использовании квот. Именно для отказа от них при работе системы диск и был разбит на несколько томов. Таким образом, при работе основных компонентов информационной системы, размещенной на рассматриваемом сервере, системные ресурсы не будут тратиться на поддержание квотирования. Однако в корневой файловой системе квоты явно не будут лишними в рамках защиты сервера. При этом влияние квотирования в этой ФС на производительность системы пренебрежимо мало, поскольку количество операций по изменению размеров файлов в корневой ФС при штатной работе сервера незначительно.

Если поддержка квот в корневой файловой системе не была включена еще при установке (что было сделано умышленно в демонстрационных целях на рассматриваемой системе), то включить ее можно следующим образом:

Во-первых, требуется установить пакет quota:

```
# yum install quota
```

Затем следует в конфигурационном файле **/etc/fstab** изменить параметры монтирования корневой файловой системы с defaults на usrquota. После этого необходимо перемонтировать корневую ФС командой

```
# mount -o remount /
```

или перезагрузить сервер.

Далее следует инициализировать систему квот на корневой ФС следующей командой:

```
# quotacheck -cmv
```

В результате ее выполнения в корневой ФС появится файл **/aquota.user**, в котором и будут содержаться сведения о квотах.

Следующим шагом следует определиться с пользователями, для которых следует задать квоты, и с размером соответствующих им квот. На данной системе квоты следует задать для системных пользователей, от имени которых будут работать сервер баз данных и веб-сервер, а также для пользователя zubastad. При чем следует помнить, что для этого пользователя квота оказывает влияние (при текущей разметке диска) и на его домашний каталог.

Для того, чтобы определиться с размерами квот и текущим объемом данных на дисках, от которых они зависят, можно воспользоваться командой

```
# df -h
```

Теперь можно задать квоты пользователям zubastad, apache, mysql:

```
# setquota -u zubastad 512000 512000 10000 10000 /
# setquota -u apache   512000 512000 10000 10000 /
# setquota -u mysql    512000 512000 10000 10000 /
```

Здесь задан предел (одновременно «жесткий» и «мягкий») в 500 Мб (приближенно), что вполне достаточно для хранения администратором сайта некоторого количества заметок, сценариев, файлов и т. п. При этом размер квоты указывается в Кб, поэтому 500 Мб представлены как 512000 Кб. Кроме того, предел (так же одновременно «жесткий» и «мягкий») установлен и на количество создаваемых файлов и каталогов (10000), т. к. атака на файловую систему может быть направлена на переполнение не только дискового пространства, но самой файловой таблицы.

Следует сказать несколько слов о журналировании квот. Описанные выше квоты реализованы без журналирования. При этом в случае нештатного перезапуска сервера в процессе его загрузки система будет вынуждена полностью перепроверить состояние квот, вычислив в том числе суммарный объем файлов и их количество для каждой установленной квоты соответственно. В случае, если таких файлов окажется достаточно много, загрузка системы может замедлиться. Во-избежание этого можно использовать журналируемые квоты аналогично журналируемым файловым системам. Однако в рассматриваемом случае это не дает существенных преимуществ:

Во-первых, аварийная перезагрузка рассматриваемого сервера представляется крайне редким на практике явлением.

Во-вторых, число защищаемых квотами файлов и их суммарные объемы невелики, повторное вычисление квот не займет много времени.

В-третьих, журналируемые квоты потребляют немного больше системных ресурсов при штатной работе сервера. Ресурсы сервера в конкретном рассматриваемом случае весьма ограничены.

Отключение rpcbind

Rpcbind – это сетевая служба, уведомляющая клиентов о том, по какому адресу и на каком порту работает та или иная служба в составе сети. На рассматриваемой системе эта служба не используется. Более того, сервер, на котором эта служба работает, может стать инструментом DDoS-атаки на посторонний компьютер. Именно поэтому некоторые хостинг-провайдеры предупреждают своих клиентов о необходимости защитить эту службу. Исходя из этого очевидным решением является отключение или удаление этой службы.

Для отключения этой службы достаточно выполнить команды:

```
# systemctl stop rpcbind
# systemctl disable rpcbind
```

Проверить состояние службы можно командой:

```
# systemctl status rpcbind
```

А убедиться, что порт службы (111) не прослушивается можно так:

```
# ss -l | grep 111
```

Отсутствие вывода этой команды – есть подтверждение того, что порт закрыт.

Для удаления rpcbind можно воспользоваться командой:

```
# yum remove rpcbind
```

Однако удаление rpcbind несовместимо с использованием пакета quota, т. е. квот в файловой системе. Пакет quota имеет rpcbind своей пакетной зависимостью. Тем не менее, дисковые квоты продолжают работать, даже если служба rpcbind отключена. Убедиться в этом можно при помощи простого эксперимента (выполняя команды от имени пользователя zubastad):

```
> quota
> cat /usr/share/doc/readline-6.2/CHANGES > /tmp/file1
> quota
```

Тем самым выполнены проверка текущего состояния квот пользователя, создание от его имени файла /tmp/file1, копирование в него содержимого одного из сравнительно больших текстовых файлов (первого попавшегося) и повторная проверка квот. Сравнение выводов команды quota наглядно отображает как увеличение числа файлов (files), принадлежащих zubastad, так и увеличение задействованного дискового пространства (blocks).

Таким образом, при использовании квот rpcbind следует отключить, при отказе от использования квот – удалить.

Об установке PHP

По причине того, что на момент написания этого документа в репозиториях CentOS была размещена слишком старая версия php, при настройке системы может возникнуть необходимость установить более свежую версию из стороннего источника. Примером такого источника может служить https://webtatic.com. Инструкцию по установке можно найти по адресу https://www.stephenrlang.com/centos-7-apache-2-4-with-php-fpm/ .

С точки зрения безопасности сервера следует сделать два замечания:

Во-первых, предложенная выше версия php (php-fpm) способна взаимодействовать с веб-сервером не только через TCP-сокет, но и через сокет домена UNIX. Последний вариант гораздо предпочтительнее как в плане производительности, так и в плане безопасности, так как сетевой стек не задействуется. Следовательно, и взаимосвязи с межсетевым экраном такой механизм не несет.

Во-вторых, настройки параметров интерпретатора php позволяют ограничивать максимально задействованный объем памяти, предельные размеры загружаемых и выгружаемых файлов и т. д. Однако, аналогично веб-серверу, значения таких

пределов трудно прогнозируемы, сильно зависят от реализуемой на сервере информационной системы. Поэтому их оптимизацию наиболее рационально выполнять по опыту тестовой и/или строевой эксплуатации сервера.

Настройка fail2ban

Прежде всего fail2ban нужно установить. Для этого придется также подключить репозиторий, в котором он хранится. Все это выполняется командами:

```
# yum install epel-release
# yum install fail2ban
```

Затем требуется создать конфигурационный файл fail2ban. В стандартной установке присутствует образец такого файла (**/etc/fail2ban/jail.conf**), в котором описаны параметры по умолчанию. Изменять этот образцовый файл запрещено, т. к. все изменения в нем затираются при обновлениях. Вместо этого документация fail2ban рекомендует создать его копию под именем **/etc/fail2ban/jail.local** и в ней внести все изменения по сравнению с настройкой по умолчанию.

Утилита fail2ban способна отслеживать различные способы входа в систему. Поэтому перед ее настройкой следует ответить на ряд вопросов:

1. Какие сервисы следует защищать?
2. Какое количество инцидентов безопасности должно приводить к блокировке (для каждого защищаемого сервиса)?
3. Каким должен быть интервал блокировки?
4. Следует ли уведомлять системного администратора об инцидентах безопасности и применении блокировок?

Отвечая на эти вопросы следует иметь ввиду следующие соображения:

- включение защиты сервиса, не работающего в системе, ведет к сбою запуска fail2ban;
- чем больше служб подлежат защите, тем больше системных ресурсов задействуется;

В рассматриваемой системе приняты следующие соображения (ответы на поставленные вопросы):

1. Следует защищать только службы ssh (работает на нестандартном порту 33631) и httpd (apache), т. к. для службы баз данных прямое подключение извне не предусматривается.
2. Блокировка производится после 5 инцидентов безопасности в течение 10 минут, для обеих служб.
3. Интервал блокировки составляет 10 минут для httpd и 1 час для ssh.
4. Уведомление системного администратора не требуется.

С учетом изложенных соображений конфигурационный файл **/etc/fail2ban/jail.local** следует привести к виду:

```
[DEFAULT]

ignoreip = 127.0.0.1/8
enabled = false

# JAILS

[sshd]
enabled  = true
port     = 33631
bantime  = 3600
findtime = 600
maxretry = 5
logpath  = %(sshd_log)s
backend  = %(sshd_backend)s

[sshd-ddos]
enabled  = true
port     = 33631
bantime  = 3600
findtime = 600
maxretry = 5
logpath  = %(sshd_log)s
backend  = %(sshd_backend)s

[apache-auth]
#enabled  = true
port     = http,https
bantime  = 600
findtime = 600
maxretry = 5
logpath  = %(apache_error_log)s
```

Как видно из конфигурационного файла, по умолчанию не контролируется локальный трафик и не активирована ни одна из темниц (jail). Далее для каждой темницы задается набор параметров, регулирующих ее работу. При этом параметр enabled управляет тем, используется ли темница или нет. На данный момент используются только темницы sshd и sshd-ddos. Еще одна темница приведена как пример. В образцовом конфигурационном файле **/etc/fail2ban/jails.conf** приведен обширный набор темниц для разных аспектов функционирования веб-сервера apache и не только для него. Однако вопрос об их использовании следует соотносить с актуальностью угрозы и затратами вычислительных ресурсов на работу этих темниц. Наиболее рациональный способ выяснения ответа на этот вопрос состоит в опытной эксплуатации системы.

После завершения редактирования конфигурационного файла следует перезапустить службу fail2ban и убедиться в успешности перезапуска:

```
# systemctl enable fail2ban
# service fail2ban restart
# service fail2ban status
```

В ходе работы fail2ban иногда требуется просмотреть список заблокированных в данный момент сетевых узлов. Это можно сделать при помощи команды (например для темницы sshd):

```
# ipset -L fail2ban-sshd
```

т. к. для реализации блокировок на рассматриваемой системе fail2ban использует модуль ipset межсетевого экрана iptables.

Журналирование

В определенных условиях злоумышленник может направить свои усилия на систему журналирования сервера. Заставляя атакуемую службу генерировать большое количество записей в ее журнале, злоумышленник может преследовать две цели:

- «затереть» ротацией журнальных записей следы своей деятельности;
- заполнить дисковое пространство сервера «мусором», что потенциально может привести к отказу обслуживания.

Для противодействия таким атакам можно предпринять следующие меры:

- вести журналы служб на отдельном сервере журналирования;
- вести журналы служб на отдельном томе файловой системы;
- отказаться от ведения журналирования;
- ограничить системные службы дисковыми квотами;
- настроить ротацию журнальных файлов;
- использовать системы резервного копирования;
- тщательно настроить режимы журналирования самих служб;
- использовать дополнительные средства безопасности (SELinux и т. п.).

Каждая из этих мер имеет как достоинства, так и недостатки:

Перенаправление журналирования на отдельный сервер журналирования позволяет не задействовать дисковое пространство виртуального сервера вообще. Кроме того, попытки злоумышленника добраться до журнальных файлов натолкнутся на необходимость проникновения на ещё один сервер. Однако эта мера требует не только наличия отдельного сервера журналирования, но и его настройки, администрирования и защиты и/или изоляции связывающего серверы сегмента сети. Поэтому в рассматриваемой ситуации этот способ защиты едва ли применим.

Отдельный том для журналирования ведет к некоторым потерям дискового пространства. Обычно эти потери незначительны, но в рассматриваемой задаче размер диска и без того очень невелик.

Самым радикальным методом защиты от атаки на переполнение журналов является полный отказ от журналирования (например, запись журнала в **/dev/null**). Однако отсутствие журнала – это крайняя мера, т. к. в этом случае никаких сведений о злоумышленнике оставаться в системе не будет.

Ограничение системных служб дисковыми квотами позволяет защитить файловую систему сервера от переполнения, но, в общем случае, не предотвращает атаку на отказ обслуживания на конкретный сервис. Так, ограничив квотами сервер баз данных, можно прийти к ситуации, когда в следствие атаки СУБД будет парализована (если сама СУБД не имеет возможности работать в условия, когда запись информации

в системный журнал невозможна). Тем не менее в этой ситуации останутся работоспособными другие службы системы, в первую очередь ssh. Однако есть некоторые сложности с реализацией этой меры: если служба работает с журнальным файлом от имени суперпользователя (root), то установка квот для него затронет не только саму службу но и большую часть системы, поэтому она практически лишена смысла и/или вредна. В рассматриваемой системе и fail2ban, и httpd ведут свои журналы именно так, а mysqld, напротив, ведёт журналы от имени mysql, поэтому хорошо поддается этому методу защиты.

Ротация журнальных файлов позволяет время от времени затирать самые старые записи и удерживать общий объем журнальных файлов в некоторых пределах. Это очень удобное и полезное средство, активированное по умолчанию в большинстве систем, в том числе в рассматриваемой. Механизм работы утилиты logrotate, реализующей его, состоит в том, что служба временно приостанавливается (на короткий срок, на практике обычно исчисляемый долями секунд), её журнальные файлы обрабатываются (усекаются, сжимаются, архивируются, уничтожаются и т. п.), затем работа службы возобновляется. Сам запуск утилиты выполняется обычно по расписанию, обычно при помощи утилиты cron. Исходя из характера работы службы следует отметить, что она не способна противостоять атаке в реальном времени: если злоумышленнику не помешают другие средства защиты, он сможет переполнить файловую систему и заблокировать работу до следующего по расписанию запуска logrotate.

Использование систем резервного копирования может быть применено для перемещения архивов журналов на другую систему. Впрочем, это можно выполнять и вручную.

Использование дополнительных средств безопасности (SELinux) для рассматриваемого сервера явно избыточно.

Гораздо правильнее прибегнуть к механизмам настройки самих служб и добиться, чтобы они вели свои журналы разумно, без лишних подробностей. Также следует обязательно настроить систему ротации журналов, так как при ее отсутствии даже без атак рост с течением времени размеров журналов может стать проблемой.

Резюмируя вышесказанное в этом разделе, можно составить следующий план защиты систем журналирования:

1. настроить подробность журналирования;
2. обеспечить ротацию журналов при помощи logrotate.

По умолчанию сервер ssh настроен на достаточно подробное журналирование (LogLevel INFO). Эти настройки менять не следует: ssh не является публикуемой службой и сам факт обращения к ней – уже событие безопасности (хотя еще и не инцидент). В процессе штатной работы сервера и даже многократных попыток входа в систему демон ssh не спровоцирует лавинообразный рост размера своего журнала (в этом ему поспособствует fail2ban).

СУБД и веб-сервер напротив, предполагаются к доступу извне.

MariaDB предусматривает возможность подробного журналирования своей работы, по умолчанию ведение большинства журналов отключено. В рассматриваемой задаче следует принять эти настройки по умолчанию оптимальными (MariaDB ведет только журнал ошибок **/var/log/mariadb/mariadb.log**) .

С настройками по умолчанию apache2 ведет на рассматриваемой системе два журнальных файла: **/var/log/httpd/error_log** и **/var/log/httpd/access_log** . Отдельные журналы могут вести и модули веб-сервера. Так же как и для MariaDB, настройки по умолчанию следует считать оптимальными для рассматриваемой задачи.

Далее следует привести конфигурационные файлы logrotate к следующему виду (некоторые комментарии сокращены):

/etc/logrotate.d/httpd:

```
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
    delaycompress
    postrotate
        /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
    endscript
    rotate 5
    size 10M
    compress
}
```

/etc/logrotate.d/fail2ban:

```
/var/log/fail2ban.log {
    missingok
    notifempty
    postrotate
      /usr/bin/fail2ban-client flushlogs >/dev/null || true
    endscript
    rotate 3
    size 10M
    compress
}
```

/etc/logrotate.d/mariadb:

```
# This logname can be set in /etc/my.cnf
# by setting the variable "log-error"
# in the [mysqld_safe] section as follows:
#
# [mysqld_safe]
# log-error=/var/log/mariadb/mariadb.log
#
# If the root user has a password you have to create a
# /root/.my.cnf configuration file with the following
# content:
#
# [mysqladmin]
# password = <secret>
# user= root
#
```

```
# where "<secret>" is the password.
#
# ATTENTION: This /root/.my.cnf should be readable ONLY
# for root !

# Then, un-comment the following lines to enable rotation of mysql's log file:

/var/log/mariadb/mariadb.log {
        create 640 mysql mysql
        notifempty
        size 10M
        rotate 5
        missingok
        compress
    postrotate
      # just if mysqld is really running
      if test -x /usr/bin/mysqladmin && \
         /usr/bin/mysqladmin ping &>/dev/null
      then
         /usr/bin/mysqladmin flush-logs
      fi
    endscript
}
```

Тем самым logrotate для всех служб указано следующее:

- хранить не более 5 файлов журнала;
- ротировать файл журнала как только он превысит 10 МБ;
- сжимать все файлы журнала кроме последнего ротированного и текущего.

Для проверки правильности заполнения этих файлов следует запустить службу способом аналогичным тому, которым эту службу будет в рабочем режиме запускать cron:

```
logrotate -d /etc/logrotate.d/sshd
logrotate -d /etc/logrotate.d/httpd
logrotate -d /etc/logrotate.d/mariadb
```

Описание файервола

При рассмотрении файервола требуется прежде всего понимать причину того, почему ранее в этом документе, а также во множестве других документов и руководств при описании одной и той же системы упоминаются «различные» межсетевые экраны: netfilter, firewalld, nftables, iptables. Дело в том, что файервол на самом деле один – компонент ядра netfilter. Все остальное – это лишь его интерфейсы.

Не имеет смысла повторять описание всех настроек сетевого экранирования здесь, т. к. они уже были приведены ранее при описании защищаемых служб (ssh, httpd, mysql) и служб, взаимодействующих с сетевым экраном (fail2ban).

Уместно будет привести командой

```
# firewall-cmd --list-all
```

полный листинг состояния файервола рассматриваемой системы и прокомментировать его:

```
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: enp0s3
  sources:
  services: ssh dhcpv6-client http https
  ports:
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
```

Как уже было сказано выше, firewalld – это интерфейс управления файерволом, призванный максимально упростить управление сетевым экраном на типовых задачах. Если выполнить листинг состояния файервола командой

```
# iptables-save
```

можно получить гораздо более полное описание его состояния и внутреннего устройства, однако рассмотрение деталей реализации файервола не является целью этого документа. Необходимо лишь сделать ряд замечаний.

Среди доступных извне сервисов можно наблюдать ssh (с измененным номером порта, о чем брандмауэр осведомлен), http и https, что ожидаемо. Однако среди них присутствует еще и dhcpv6-client, который не был явно открыт при настройке системы.

Попытка отключения dhcpv6-client командой

```
# firewall-cmd --zone=public --permanent --delete-service=dhcpv6-client
```

приводит к сообщению

```
Error: BUILTIN_SERVICE: 'dhcpv6-client' is built-in service
```

Наличие этой службы потенциально (тем более, что адрес задается статически согласно настройкам сетевого интерфейса) позволяет злоумышленнику назначить рассматриваемой системе некоторый IPv6-адрес, однако не дает ему иных преимуществ.

Если же из соображений безопасности требуется все же заблокировать работу этой службы, то можно воспользоваться настройками сетевого интерфейса и параметрами ядра системы.

Во-первых, отключить поддержку IPv6 на уровне ядра системы, для этого следует добавить в конфигурационный файл **/etc/sysctl.d/99-sysctl.conf** следующие строки:

```
net.ipv6.conf.all.disable_ipv6=1
net.ipv6.conf.default.disable_ipv6=1
```

и убедиться, что в нем и подключаемых к нему файлах нет прямых противоречий этим строкам. На рассматриваемой системе по умолчанию таких противоречий нет.

Затем следует привести конфигурационный файл (на рассматриваемой системе, на других системах имя интерфейса и, соответственно, файла могут отличаться, см. подробнее раздел «Первичная настройка после установки») **/etc/sysconfig/network-**

scripts/ifcfg-enp0s3 к виду, не противоречащему следующим параметрам:

```
IPV6INIT="no"
IPV6_AUTOCONF="no"
IPV6_DEFROUTE="no"
IPV6_FAILURE_FATAL="no"
```

Затем достаточно перезагрузить систему и по выводу команды

```
# ip addr show
```

убедиться, что ни одному из интерфейсов не назначен никакой адрес IPv6.

При желании самостоятельно контролировать все процессы в файерволле можно рекомендовать отказаться от использования firewalld в пользу iptables, однако не стоит этого делать без достаточных навыков работы с iptables. Также можно рассмотреть вопрос об использовании нового интерфейса netfilter – nftables, обладающего очень удобным и легко читаемым синтаксисом. Однако и его использование на строевой системе требует опыта. Для неискушенного в сетевом экранировании пользователя VPS рациональнее всего использовать именно firewalld, смирившись с некоторыми его особенностями.

Далее приведен отчет о результатах TCP SYN сканирования портов рассматриваемого виртуального сервера средствами утилиты nmap в виртуальной среде. Следует отметить, что такое сканирование следует выполнять **исключительно** в виртуальной изолированной среде, т. к. сканирование портов в публичных сетях в подавляющем большинстве случаев является признаком подготовки атаки, и, соответственно, Интернет-провайдеры на любом участке сети, хостинг-провайдеры и соответствующие органы правопорядка реагируют на такое сканирование исходя из предположения о подготовке атаки.

Отчет о TCP SYN сканировании локальной виртуальной копии рассматриваемого сервера:

```
Nmap scan report for 192.168.0.99
Fetchfile found /usr/bin/../share/nmap/nmap-mac-prefixes
Host is up, received arp-response (0.00038s latency).
Scanned at 2019-03-05 11:07:18 MSK for 298s
Not shown: 65533 filtered ports
Reason: 65230 no-responses and 303 host-prohibiteds
PORT      STATE  SERVICE REASON
80/tcp    open   http    syn-ack ttl 64
443/tcp   closed https   reset ttl 64
33631/tcp open   unknown syn-ack ttl 64
MAC Address: 08:00:27:36:9C:6A (Oracle VirtualBox virtual NIC)
Final times for host: srtt: 375 rttvar: 189  to: 100000

Read from /usr/bin/../share/nmap: nmap-mac-prefixes nmap-payloads nmap-services.
Nmap done: 1 IP address (1 host up) scanned in 297.99 seconds
           Raw packets sent: 196451 (8.644MB) | Rcvd: 309 (22.056KB)
```

В отчете показано, что фильтруются (т. е. закрыты сетевым экраном) все порты кроме 80, 443 и 33631. При этом порт 443 закрыт веб-сервером, т. к. на момент тестирования этот порт не использовался самим веб-сервером (не было сконфигурировано ни одного такого сайта). Порт 33631 открыт, но nmap (при таком сканировании) не смогла определить, что это служба ssh.

Следует также отметить, что обычно при сканировании портов жертвы для минимизации привлекаемого внимания чаще всего ограничиваются младшей тысячью портов, поэтому на практике злоумышленник мог счесть, что служба ssh в системе не используется (например потому, что она работает на другом IP-адресе, или потому, что управление сайтом выполняется через панель управления хостинг-провайдера).

Передача файлов на VPS и обратно

Принятые жесткие меры безопасности радикально ограничили взаимосвязи сервера с внешним миром. В силу этого возникает вопрос о передаче файлов с компьютера администратора сайта на VPS и обратно. При этом следует понимать, что передаваемые файлы по способу передачи могут быть разделены на две категории: содержимое информационной системы, передачей которых управляет CMS, и системные файлы, в том числе и сама CMS на этапе своей установки.

Здесь описана лишь вторая категория. Но и в ней можно выделить два случая.

Первый случай – это необходимость с VPS загрузить данные с удаленного веб-сервера (например, скачать CMS). Здесь прекрасно подходит утилита wget. На рассматриваемой системе она по умолчанию не установлена, для установки следует выполнить команду

```
# yum install wget
```

Второй случай – это обмен данными без использования веб-сервера с какой-либо стороны. Здесь можно использовать утилиту scp, входящую в состав паакета ssh. Эта утилита использует те же протоколы, конфигурационные файлы и права доступа, что и ssh, по этой причине она не отличима от ssh с точки зрения сетевого экрана, не требует дополнительных настроек и не несет дополнительных рисков.

Далее приведены примеры команд со стороны клиента (компьютера администратора сайта) для передачи файлов в обоих направлениях:

Загрузка файла на сервер:

```
> scp -P 33631 ~/file1.txt zubastad@mysite.ru:/tmp/file1.txt
```

Выгрузка файла с сервера:

```
> scp -P 33631 zubastad@mysite.ru:/tmp/file2.txt ~/file2.txt
```

Следует обратить внимание, что в отличие от ssh ключ для указания порта не -p, а -P.

В случае использования администратором сайта не UNIX-подобной операционной системы на своем компьютере следует обратить внимание на программное обеспечение, реализующее функции для используемой операционной системы. Для MS Windows примером такого ПО является Winscp.

Общие правила эксплуатации

В дополнение ко всему вышеописанному следует сделать несколько замечаний:

Повторное, целевое по портам сканирование (еще более подозрительное с точки зрения провайдеров и служб безопасности), направленное на определение версий ПО, привело к следующим результатам:

```
Starting Nmap 7.40 ( https://nmap.org ) at 2019-03-06 11:46 MSK
Nmap scan report for 192.168.0.99
Host is up (-0.078s latency).
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.6 ((CentOS))
MAC Address: 08:00:27:36:9C:6A (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open
and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.10 - 4.2, Linux 3.2 - 4.6
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.60 seconds

Starting Nmap 7.40 ( https://nmap.org ) at 2019-03-06 11:46 MSK
Nmap scan report for 192.168.0.99
Host is up (-0.10s latency).
PORT    STATE  SERVICE VERSION
443/tcp closed https
MAC Address: 08:00:27:36:9C:6A (Oracle VirtualBox virtual NIC)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 4.42 seconds

Starting Nmap 7.40 ( https://nmap.org ) at 2019-03-06 11:46 MSK
Nmap scan report for 192.168.0.99
Host is up (0.00047s latency).
PORT      STATE SERVICE VERSION
33631/tcp open  ssh     OpenSSH 7.4 (protocol 2.0)
MAC Address: 08:00:27:36:9C:6A (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open
and 1 closed port
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.10 - 4.2, Linux 3.2 - 4.6
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.27 seconds
```

Очевидно, что удалось выяснить версии всех опубликованных служб (OpenSSH 7.4 (protocol 2.0), Apache httpd 2.4.6), дистрибутив (CentOS) и версию ядра в некотором диапазоне (Linux 3.10 - 4.2, Linux 3.2 — 4.6).

Едва ли злоумышленник сможет провести такое сканирование на практике, однако пренебрегать обновлением программного обеспечения не стоит: вероятность присутствия уязвимостей в новых (стабильных) версиях ПО всегда меньше,

обнаруженные уязвимости закрываются, новые уязвимости (если таковые есть) нарушителю еще надо обнаружить и исследовать, прежде чем использовать.

С другой стороны, изредка случается так, что обновление ПО нарушает привычный режим работы системы. Поэтому можно рекомендовать следующий способ поддержки ПО в актуальном состоянии: иметь клон строевого виртуального сервера в изолированной локальной подсистеме (компьютер администратора сайта), на этом клоне проверять все предполагаемые изменения, в том числе обновление ПО, в случае успеха проводить обновление и на строевой системе.

В дополнение к мерам безопасности, описанным выше, следует помнить о том, что для проникновения в систему или нанесения ей ущерба могут быть применены и пути, не описанные в этом документе: атаки на систему управления контентом (CMS), попытки подбора паролей к личному кабинету арендатора VPS на сайте хостинг-провайдера, проникновение в компьютер администратора сайта, перенаправление на поддельный сервер с целью выявления паролей и т. п.

Например, техника последней атаки строится примерно так:

1. подготовить сервер атакующего, принимающий запросы на любых портах с подробным журналированием трафика;
2. подменить DNS-ответ (DNS spoofing, он же DNS poisoning) для компьютера администратора сайта таким образом, чтобы он был перенаправлен не на свой VPS, а на сервер атакующего;
3. дождаться попытки администратора сайта войти в систему и выяснить, что ssh работает на порту 33631 и логин для входа zubastad@mysite.ru;
4. путь к подобру пароля (brute force) открыт;
5. brute force прерван, следовательно в системе работает fail2ban;
6. bute force возобновлен с другого адреса или по истечении времени блокировки, но теперь он растянут во времени.

Разумеется, нападение такого рода – это удел хорошо подготовленного и мотивированного нарушителя, тем не менее при обнаружении признаков такой атаки можно принять следующие меры:

Перед запуском собственного сервера в строевую эксплуатацию провести учения для себя и составить карту действий (в том числе запасные пароли, имена пользователей, команды для их смены), например:

При неудачных попытках входа на собственный сервер (этап атаки №3):

- срочно сменить Интернет-провайдера и/или маршрутизатор (особенно wifi), например зайти на сайт из другой wifi-сети или через мобильный Интернет;
- сменить пароли и/или имя пользователя;

- проанализировать системные журналы, текущие списки fail2ban, возможно увеличить интервал блокировки.

Возможно, вход на собственный сервер выполнить и не по протоколу ssh, а через личный кабинет на сайте хостинг-провайдера по протоколу spice или vnc. При входе в личный кабинет обязательно убедиться в адресной строке браузера, что адрес панели управления корректен и подтвержден сертификатом безопасности.

Другим примером атаки может быть следующий алгоритм:

1. подготовить сервер-клон, имитирующий поведение панели управления хостинг-провайдера, журналирующий все введенные логины и пароли;
2. блокировать администратору любыми доступными средствами доступ по ssh к серверу, тем самым вынудив его войти в систему через панель управления хостинг-провайдера;
3. ждать, рассчитывая на то, что администратор сайта попытается войти в панель управления своего VPS не убедившись, что это сайт именно хостинг-провайдера.

Разумеется, описаны далеко не все способы атаки, разумеется, такие атаки трудно осуществимы, разумеется, вероятность атаки на обычный VPS со стороны столь подготовленного и мотивированного нарушителя стремится к нулю, но принять меры защиты можно и сравнительно несложно.