Решил поделиться информацией из своих наработок. Я расскажу, как настроить резервный (подменный) веб сервер с актуальной копией сайта, на который можно временно переключиться в случае проблем с основным. Так же этот резервный сервер можно использовать как dev, проводить над ним эксперименты или восстанавливать старые копии сайтов.

Содержание

Введение

Данная статья не будет подробным пошаговым описанием, с помощью которого можно будет простым копипастом настроить то же самое у себя. Я покажу только идею и некоторые части bash скриптов. Это не потому, что мне жалко или трудно написать подробную статью. Просто у меня чаще всего настройки как бэкапа, так и восстановления, уникальны и зависят от общей инфраструктуры, которая используется. Сайты чаще всего за nginx proxy, а бэкапы складываются в разные места с использованием разных технологий и разной глубиной хранения. Все настраивается по месту в зависимости от потребностей.

Идея статьи в том, чтобы показать, как подготовить запасной веб сервер, где будет храниться и обновляться актуальная версия сайта. На этот сервер можно будет оперативно переключиться в случае аварии на основном. При этом не важно, как вы будете переключать трафик - через редактирование DNS записи или переключение бэкенда на балансере. Так же я настрою мониторинг резервных сайтов с помощью Zabbix, чтобы автоматически проверять

работу подменного сайта.

В статье я буду считать, что у вас установлен и настроен сервер centos примерно так же, как описано в моих статья. В качестве web сервера выступает подобная инсталляция.

Бэкап текущего сайта

Прежде чем подготовить подменный сервер, надо сделать бэкап сайта с текущего сервера. Тут я не буду ничего изобретать и показывать какие-то особые приемы. Все делается очень просто через **rsync**. Подробно его работу я рассказываю в статье - настройка бэкапа через rsync. С его помощью на сервер с бэкапами уезжают как исходники сайтов, так и дампы баз данных mysql. С базами тоже все просто - делаем обычный дамп через **mysqldump**. Если база большая, нужны инкрементные бэкапы - использую Percona XtraBackup.

Для долгосрочного хранения сжимаю исходники в архивы и кладу куда-то в s3 или другое хранилище. Ниже пример скрипта для бэкапа исходников. Я делаю дамп базы, кладу его в корень сайта, сжимаю все в архив и удаляю дамп. Сделано так для того, чтобы все было в одном файле - исходники и база.

```
#!/bin/bash

# Добавляю метку времени в имя файлов
date_time=`date +"%Y-%m-%d_%H-%M"`
# Директория, куда кладем архив
bk_dir='/mnt/backup/day'
# Директория с сайтами
inf_dir=/web/sites
# Директория, которая идет в архив
dir_to_bk='www'
# Пользователь БД
user='serveradmin'
# Пароль пользователя БД
password='passmysql*%'
# Исключения по расширениям файла
exclude_ext='--exclude='*.log' --exclude='*.tmp''
# Исключения по директориям
```

```
exclude_dir='--exclude='wp-content/cache/*''

# Дамп базы данных кладу в корень сайта
/usr/bin/mysqldump --opt -v --no-create-db serveradmin -u$user -p$password >
$inf_dir/serveradmin.ru/www/mysql_serveradmin.ru_$date_time.sql
# Бэкаплю сайт вместе с дампом базы
/usr/bin/tar ${exclude_ext} ${exclude_dir} -czvf  $bk_dir/serveradmin.ru_$date_time.tar.gz -C $inf_dir/serveradmin.ru
$dir_to_bk
# Удаляю из корня сайта дамп БД
/usr/bin/rm -rf $inf_dir/serveradmin.ru/www/mysql_serveradmin.ru_$date_time.sql
```

Дальше этот архив можно отправить в s3 с помощью rclone или чего-то подобного.

Если вам нет нужды класть дамп базы вместе с исходниками сайта, а так же если у вас сервер БД это отдельная машина, то бэкапы можно делать таким скриптом. Он положит дамп каждой базы в отдель

```
#!/bin/bash

for i in `mysql -uroot -p'password' -e'show databases;' | grep -v information_schema | grep -v Database`;
    do
        /usr/bin/mysqldump -uroot -p'password' $i | /usr/bin/gzip -c > /backup/mysql/`date +%Y-%m-%d`-$i.sql.gz;
    done
```

Архивы всех баз по отдельности будут сжаты и положены в директорию */backup/mysql*. Не забудьте их потом почистить, чтобы не копились. Это относится как к дампам, так и архивам. Я обычно чищу их после того, как они уезжают на бэкап сервер. Часть копий остаются на исходном сервере, если есть возможность. Так можно оперативно что-то восстановить при необходимости, не обращаясь к бэкап серверу.

Чистим файлы стандартно через **find**. Удаляем все, что старше 7-ми дней.

```
/usr/bin/find /backup/mysql/ -type f -mtime +7 -exec rm {} \;
```

С бэкапами разобрались, переходим к восстановлению.

Восстановление сайта

Теперь восстановим сайт или несколько сайтов на запасном сервере. Данные для восстановления можно брать сразу же с исходного сервера, а можно с сервера бэкапов. В первом случае у вас будет самая свежая информация, а во втором вы автоматом проверяете работу бэкапов. Как будете делать вы - решайте сами. Это не принципиально. Просто меняется адрес источника при восстановлении.

Итак, настраиваем подменный web сервер точно так же, как и основной. Лучше продублировать всю структуру каталогов. Так просто проще, хотя и не обязательно. Можно иметь и разную структуру, особенно если у вас запасной сервер один для множества разных проектов.

Сразу поясню несколько нюансов. Если вы хотите иметь как можно более свежую и актуальную копию сайта, то вам надо настраивать репликацию баз данных. Делается не сложно, но не всегда подходит. Например, если у вас подменный сервер один для нескольких web серверов, то просто взять и настроить репликацию не получится, так как в качестве источников должны выступать разные mysql сервера. Плюс, репликация иногда отваливается, за ней надо следить. В целом, это более сложное техническое решение, в отличие от восстановления из дампа. В этом случае исходные сервера БД вообще трогать не надо.

Если у вас обычный информационный сайт и отставание базы данных в несколько часов или дней для вас не критичны, то вместо репликации достаточно восстанавливать базу данных из дампа с определенной периодичностью. Например, раз в день по ночам после бэкапа. В своем примере я покажу этот вариант.

Идем дальше. Веб сервер подготовили, теперь надо создать пустые базы mysql и учетные записи к ним, такие же как и на исходных сайтах. Так проще, когда все однотипное, не надо править настройки при восстановлении. Дальше копируем исходники сайтов и дампы баз на запасной сервер. Опять ничего нового, все тот же rsync.

```
/usr/bin/rsync --delete --progress -av -e "ssh -p 22777" root@18.128.36.49:/web/sites/serveradmin.ru/www
/web/sites/serveradmin.ru
```

```
/usr/bin/rsync --delete --progress -av -e "ssh -p 22777" root@18.128.36.49:"/backup/mysql" /backup
```

Теперь нам надо восстановить базу данных. Чаще всего в директории */backup/mysql* лежат несколько баз данных от сайтов на этом сервере, поэтому надо как-то аккуратно выбрать нужную и взять самую свежую версию. Делаю примерно так, для того, чтобы распаковать самую свежую версию архива с БД, где в названии присутствует слово serveradmin.

```
gunzip /backup/mysql/`ls -t '/backup/mysql' | grep serveradmin | head -1`
mysql -userveradmin -p'passmysql*%' serveradmin < /backup/mysql/`ls -t '/backup/mysql' | grep serveradmin | head -1`
```

С помощью конструкции *ls -t '/backup/mysql' | grep serveradmin | head -1* получаю самую свежую версию файла с именем serveradmin. Я ее сначала распаковываю, а потом заливаю в базу данных.

Исходники мы залили, дамп восстановили. Не забудьте проверить права доступа на файлы сайтов. Веб сервер должен иметь к ним доступ. Если структура пользователей на подменном сервере другая, права нужно будет расставить отдельно. Теперь нужно забрать актуальные tls сертификаты. Я их обычно с помощью let's encrypt настраиваю, так что забираем их с помощью rsync.

```
/usr/bin/rsync --delete --progress -av -e "ssh -p 22777" root@18.128.36.49:/etc/letsencrypt /etc
```

Сертификаты могут лежать не на исходном веб сервере, а на прокси. Так что думайте сами, как и откуда их забирать. Дальше нужно подготовить конфиги nginx. Если у вас prod окружение и резервный сервер полностью одинаковые, то конфиги можете тоже по rsync забрать и не заморачиваться. Если нет, придется вручную готовить для каждого сайта. Когда конфиги не меняются, не проблема сделать один раз вручную. Если меняются, то приводите окружение к идентичному состоянию и так же их копируйте по rsync.

```
/usr/bin/rsync --delete --progress -av -e "ssh -p 22777" root@18.128.36.49:/etc/nginx/conf.d /etc/nginx
```

Вот и все. Мы подготовили полностью идентичный подменный web сервер. Можете у себя на компе отредактировать файл hosts, заменив ip адрес домена сайта и проверить работу подменного сервера. В его логах вы должны увидеть свои запросы.

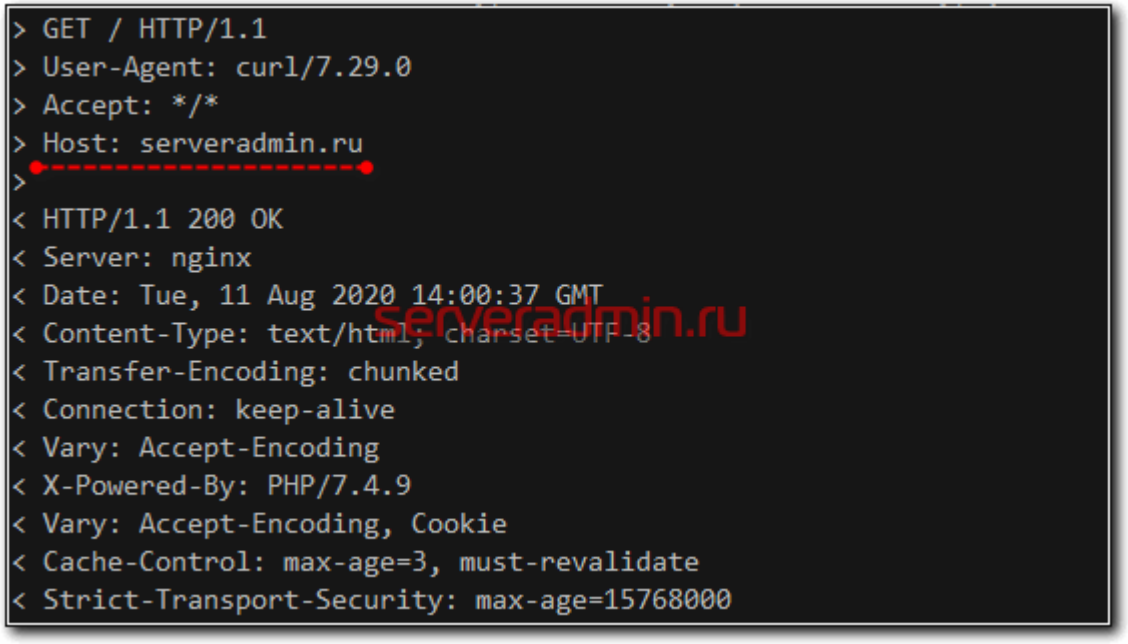
Если все в порядке, то добавляете остальные сайты по аналогии. Можете собрать все команды в один скрипт или разбить на несколько. Потом их надо добавить в cron и все готово.

Мониторинг резервного сайта

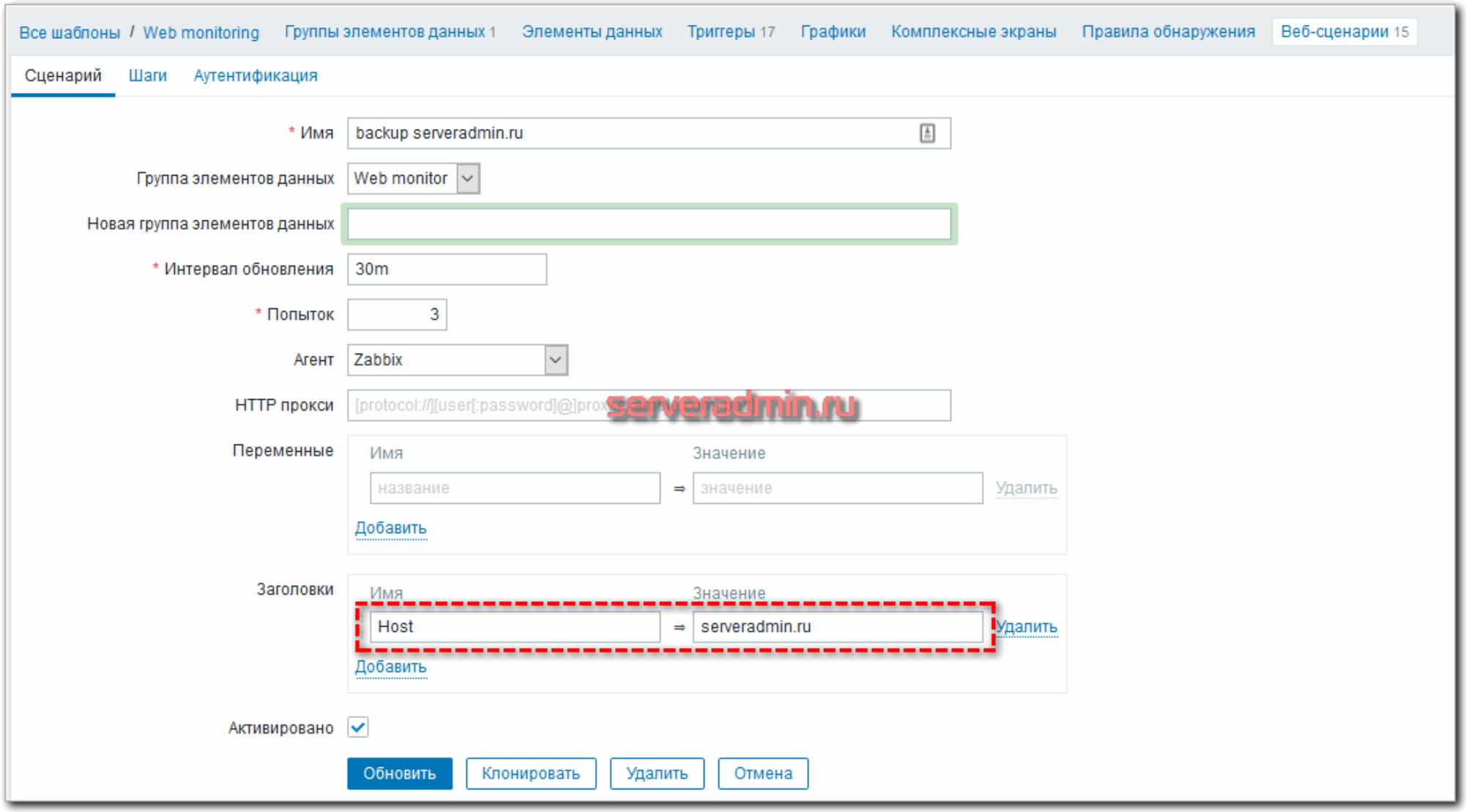
Расскажу, как можно настроить мониторинг резервного сервера и сайта, чтобы быть уверенным в том, что у нас есть рабочая копия, на которую можно переключиться в случае аварии. Как можно выполнить ручную проверку с помощью правки hosts я сказал выше. Но надо это дело автоматизировать. Как минимум, можно проверять работу сайта через консоль. Например, обращаться к резервному серверу по ip с помощью curl и передавать в заголовке имя нашего домена.

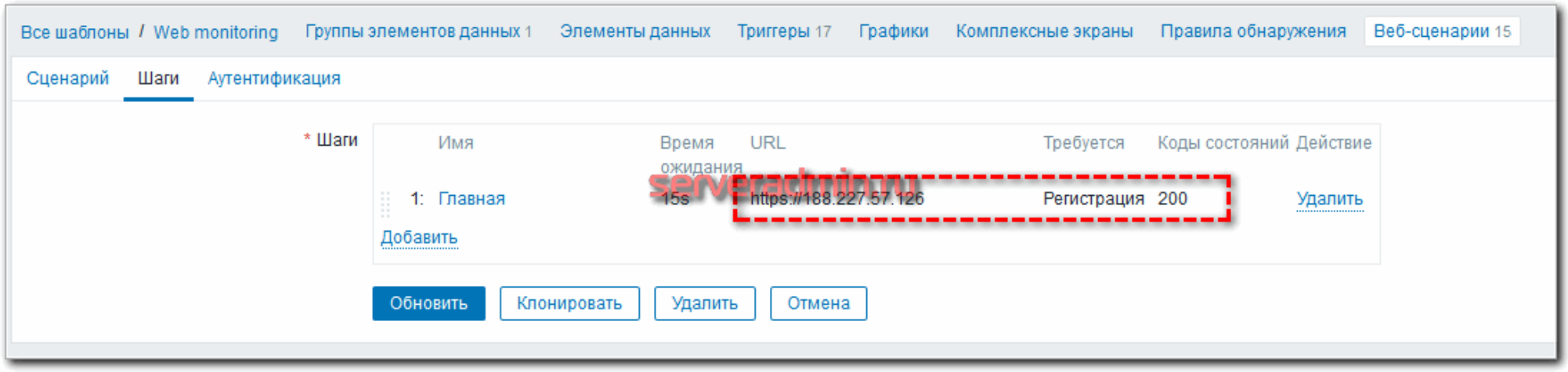
```
# curl --verbose -k -H "Host: serveradmin.ru" https://18.128.36.49
```

Веб сервер выдаст вам главную страницу вашего домена.



Можете наблюдать за кодом ответа или искать какой-то текст в исходнике страницы, для того, чтобы быть уверенным в том, что она актуальна. Но лично я все делаю через Zabbix. У меня есть статья про мониторинг сайта. Предлагаю взять ее за основу, но мониторить не только основной сайт, но и резервный. Делается это следующим образом. Мы будем мониторить сервер по ip, а имя домена будем передавать в заголовке. Создаем еще одну веб проверку.


Все шаблоны / Web monitoring
Группы элементов данных 1
Элементы данных
Триггеры 17
Графики
Комплексные экраны
Правила обнаружения
Веб-сценарии 15
Сценарий
Шаги
Аутентификация
* Имя
backup serveradmin.ru
Группа элементов данных
Web monitor
Новая группа элементов данных
* Интервал обновления
30m
* Попыток
3
Агент
Zabbix
HTTP прокси
[protocol://][user[:password]@]prox
serveradmin.ru
Переменные
Имя
Значение
название
значение
Удалить
Добавить
Заголовки
Имя
Значение
Host
serveradmin.ru
Удалить
Добавить
Активировано
Обновить
Клонировать
Удалить
Отмена



Ожидаю код ответа 200 и ищу слово *Регистрация* в исходнике. Все как в основном сайте. Дальше к этой проверке, по аналогии с основным сайтом, делается триггер. Резервный сайт можно опрашивать гораздо реже, чем основной. Хотя можно и так же часто. Тогда сможете сравнить производительность и отклик сайта с двух разных серверов. Если на резервном показатели будут лучше, то имеет смысл сделать его основным.

Обращаю внимание, что при такой проверке не мониторится tls сертификат на подменном сервере. Но я решил, что можно обойтись и без этого, хотя настроить не сложно. У меня есть статья на эту тему - мониторинг срока действия ssl сертификата.

На этом у меня все. Я показал, как можно подготовить подменный web сервер и автоматизировать разворачивание свежей версии сайта на нем. В одно действие вы сможете переключить трафик на него и будете уверены, что на нам все работает и сайт отвечает.

Заключение

> Не понравилась статья и хочешь научить меня администрировать? Пожалуйста, я люблю учиться. Комментарии в твоем распоряжении. Расскажи, как сделать правильно!

Подведем итог того, что мы сделали:

1. Настроили бэкап исходников сайтов и баз данных.
2. Подготовили подменный web сервер.
3. Перенесли на него исходники и базы данных.
4. Развернули базы данных из дампов.
5. Перенесли сертификаты сайтов.
6. Подготовили конфиги nginx.
7. Проверили работу резервного сайта и сервера.
8. Автоматизировали заливку свежих версий сайтов.
9. Настроили мониторинг резервных сайтов.

Осталось еще настроить мониторинг бэкапов и можно спать спокойно. В идеале еще и автопереключение сделать на резервный сервер, но механизм будет зависеть от конкретного случая. Если есть общий балансер, то можно переключать сервера на нем. Это удобно, но получается еще одна точка отказа (балансер), которую тоже надо резервировать. Как мне видится, лучше всего найти dns хостинг с api. Настроить ttl записи в 5-10 минут и менять автоматически ip в dns записи, если основной сервак с сайтом недоступен.

Скажу сразу, я автопереключение никогда не делал. Обычно падение прода это лютый форс-мажор, который случается раз в несколько лет. Держать постоянно наготове механизм автопереключения и тестировать его - отдельный труд. Для мелких и средних проектов это излишне. За года либо забудешь, как он работает, либо что-то сломается и в трудную минуту не отработает. Мне достаточно того, что есть живые актуальные бэкапы, и я смогу вручную переключиться на подменный сервер.

- писать сложные хранимые процедуры, функции и триггеры;
- читать план запроса.

Проверьте себя на вступительном тесте и смотрите подробнее программу по .

Помогла статья? Подписывайся на telegram канал автора

Анонсы всех статей, плюс много другой полезной и интересной информации, которая не попадает на сайт.