Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра математической
кибернетики и компьютерных наук

РАЗРАБОТКА ВЫСОКОНАДЕЖНОЙ БАЗЫ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ КОНТЕЙНЕРА

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 451 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Шиндина Владислава Евгеньевича

Научный руководитель
доцент, к. ф.-м. н. _____ В. М. Соловьёв

Заведующий кафедрой
к. ф.-м. н. _____ С. В. Миронов

Саратов 2016

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
1 Современные высоконадежные системы 4
1.1 Отказоустойчивость и избыточность 4
1.2 Преимущества контейнерной технологии 5
2 Технологии виртуализации 6
3 Особенности LXC контейнеров и их применения 7
3.1 Настройка сети 7
3.1.1 Одноразовые настройка сети на хосте 7
3.1.2 Настройка сети для каждого контейнера 8
4 Работа с LXC контейнерами для повышения надежности вычислительных систем 9
4.1 Настройка репликации базы данных на контейнерах 9
4.2 Работа репликации в контейнерах 10
4.3 Восстановление данных при отказе чтения файла в рабочей базе . 10
ЗАКЛЮЧЕНИЕ 12
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 13

ВВЕДЕНИЕ

Бакалаврская работа посвящена разработке высоконадежной базы данных с использованием контейнера. Чтобы сделать базу данных более надёжной можно применить контейнерную технологию. Позволяющую, создавать резервные копии базы на одном диске. В случае выхода из строя части секторов жёсткого диска, часть данных можно спасти, на этом строится разработка высоконадежной базы данных с использованием контейнера.

Репликация — одна из техник масштабирования баз данных. Состоит эта техника в том, что данные с одного сервера базы данных постоянно копируются (реплицируются) на один или несколько других (называемые репликами).

Использование контейнеров позволяет сделать репликацию на одном хосте. В главном контейнере создаётся база данных, с которой работает пользователь. На резервном контейнере создаётся резервная копия базы данных с помощью разработанного скрипта. Применение резервного контейнера позволяет избежать логической нецелостности таблиц в безе данных, а так же не препятствует работе пользователя.

Целями бакалаврской работы являются:

- Разработка приложений, упрощающих настройку контейнера LXC;
- Настройка репликации и создание приложения, обеспечивающее резервное копирование в автоматическом режиме;
- Восстановление базы данных, используя контейнерную технологию.

1 Современные высоконадежные системы

1.1 Отказоустойчивость и избыточность

Отказоустойчивость — свойство технической системы сохранять свою работоспособность после отказа одного или нескольких составных компонентов. Отказоустойчивость определяется количеством любых последовательных единичных отказов компонентов, после которого сохраняется работоспособность системы в целом. Базовый уровень отказоустойчивости подразумевает защиту от отказа одного любого элемента — исключение единой точки отказа. Основной способ повышения отказоустойчивости — избыточность. Наиболее эффективный метод избыточности — аппаратная избыточность, которая достигается путём резервирования. В ряде приложений отказоустойчивость путём резервирования является обязательным требованием, предъявляемым государственными надзорными органами к техническим системам.[1]

Отличительными преимуществами отказоустойчивых систем являются: их высокая безотказность, бесперебойность работы системы при наличии отказов и более продолжительный жизненный цикл эксплуатации. Отказоустойчивые системы помимо преимуществ имеют и ряд специфических характеристик, а именно: сложность дизайна и высокая стоимость развертывания, повышение энергопотребления, усложнение системы.

Избыточностью называют функциональность, в которой нет необходимости при безотказной работе системы.

Примерами могут служить запасные части, автоматически включающиеся в работу, если основная ломается. В частности, большие грузовики могут потерять шину без серьёзных последствий. На них установлено много шин, и потеря одной не является критичной (за исключением передней пары, которая служит для поворотов). Впервые идея включения избыточных частей для увеличения надежности системы была высказана Джоном фон Нейманом в 1950-х годах.[1]

Существует два типа избыточности: пространственная и временная. Избыточность пространства реализуется путём введения дополнительных компонентов, функций или данных, которые не нужны при безотказном функционировании. Дополнительные (избыточные) компоненты могут быть аппаратными, программными и информационными. Временная избыточность реализуется путём повторных вычислений или отправки данных, после чего результат

сравнивается с сохранённой копией предыдущего.

Иногда обеспечение отказоустройчивости аппаратуры требует, чтобы вышедшие из строя части были извлечены и заменены новыми, в то время как система продолжает работать (в области компьютеров известно как горячая замена).[1]

1.2 Преимущества контейнерной технологии

Одной из технологий высоконадёжных систем является RAID (Redundant Array of Independent Disks — избыточный массив независимых дисков) — технология виртуализации данных, которая объединяет несколько дисков в логический элемент для избыточности и повышения производительности.[2]

При использовании аппаратного RAID необходимо иметь одинаковые модели жёстких дисков, одинаковый их объём. Подобная технология является дорогой. В случае если производитель прекращает выпуск используемых дисков, все используемые должны быть заменены на другие. На что так же потребуется время и финансовые средства.

Если используется программный RAID, то допускается использование разных жёстких дисков. Однако, жёсткие диски, установленные в одно время, будут выходить из строя примерно в одинаковые периоды времени. Подобная технология решает проблемы, она более дешёвая, чем аппаратного RAID, но требует дополнительного оборудования и финансовых средств.

В отличии от RAID, контейнерная технология позволяет делать резервную копию данных, не прибегая к покупке дополнительных дисков, за счёт этого контейнерная технология является более дешёвой. В тот момент, когда число bad-секторов будет увеличиваться и данные будут выходить из строя, диск возможно частично восстановить. Так как на диске содержатся резервные копии, их можно использовать, как восстановлении данных при ошибках пользователя, так и при ошибках чтения файла. Так же резервные копии данных могут быть скопированы на новый жёсткий диск в случае его замены. Такая технология повышает эффективность использования жёсткого диска. Использование контейнеров позволяет предотвратить логическую нецелостность таблиц в безе данных при создании резервной копии, при этом пользователь не потеряет возможность работы с данными. Таким образом, контейнерная технология не влияет на работу пользователя и не отличима от других технологий со стороны пользователя.

2 Технологии виртуализации

Виртуализация — предоставление наборов вычислительных ресурсов или их логического объединения, абстрагированных от аппаратной реализации, и обеспечивающих изоляцию вычислительных процессов.

Понятие виртуализации можно условно разделить на две категории:

- виртуализация платформ (продуктом этого вида виртуализации являются виртуальные машины);
- виртуализация ресурсов (преследует целью комбинирование или упрощение представления аппаратных ресурсов для пользователя и получение неких пользовательских абстракций оборудования, пространств имен, сетей).

Взаимодействие приложений и ОС (операционной системы) с аппаратным обеспечением осуществляется через абстрагированный слой виртуализации.

Существует несколько подходов организации виртуализации:

- эмуляция оборудования (QEMU, Bochs, Dynamips);
- полная виртуализация (KVM, Hyper-V, VirtualBox, VMware ESXi);
- паравиртуализация (Xen, L4, Trango);
- виртуализация уровня ОС (LXC, Virtuozzo, Jails, Solaris Zones).

Среди всех технологий можно выделить контейнерную виртуализацию. Контейнерная виртуализация или виртуализация на уровне операционной системы — это метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя, вместо одного. Это снижает накладные расходы и позволяет использовать виртуализацию наиболее эффективно. Виртуализация на уровне операционной системы даёт значительно лучшую производительность, масштабируемость, плотность размещения, динамическое управление ресурсами, а также лёгкость в администрировании, чем у альтернативных решений.

Можно обозначить следующие варианты использования продуктов виртуализации:

- Консолидация серверов;
- Разработка и тестирование приложений;
- Использование в бизнесе;
- Использование виртуальных рабочих станций.

3 Особенности LXC контейнеров и их применения

LXC не использует виртуальные машины, а создает виртуальное окружение с собственным пространством процессов и сетевым стеком. [3] Все экземпляры LXC используют один экземпляр ядра операционной системы.

Ядро Linux может изолировать ресурсы (процессор, память, ввод/вывод, сеть и так далее) при помощи cgroups, не прибегая для этого к использованию виртуальных машин. cgroups (control group) — механизм ядра Linux, который ограничивает и изолирует вычислительные ресурсы (процессорные, сетевые, ресурсы памяти, ресурсы ввода-вывода) для групп процессов. Механизм позволяет образовывать иерархические группы процессов с заданными ресурсными свойствами и обеспечивает программное управление ими. Посредством cgroups изолируются так же пользователи и файловые системы.[4]

Пространства имен (Namespaces) бывают разные — PID namespace, IPC namespace, UTS namespace, user namespace, mount namespace, network namespace. Пространства имен изолируют друг от другая процессы таким образом, что процессы в одной группе не могут видеть ресурсы другой группы. Например, PID namespace предоставляет уникальные идентификаторы процессов в рамках группы. Внутри одной группы может быть процесс с pid'ом 1 и внутри второй группы тоже может быть процесс с pid'ом 1, хотя это два совершенно разных процесса, которые друг о другие ничего не знают. Притом, все процессы все также имеют уникальные id в рамках ОС. Если смотреть на процессы из группы, то эти id отображаются в другие.[5]

3.1 Настройка сети

Создан контейнер `container1`. Требуется настроить сеть на этом контейнере и проверить работу сети.

3.1.1 Одноразовые настройка сети на хосте

Настройки на хосте делаются один раз. Нужно открыть файл:

```
/etc/sysctl.conf
```

Затем нужно раскомментировать строку [6], позволяющую включить форвардинг: `net.ipv4.ip_forward=1` Форвардинг — маршрутизация транзитных IP-пакетов (не предназначенных для этого компьютера), или IP-форвардинг. В

данном случае будет осуществляться пропуск через себя пакетов от контейнеров [7].

На хосте нужно запустить первый приложение. Данный скрипт упрощает настройку. Сперва запрашивается IP адрес подсети и маска подсети. Потом требуется ввести сетевой интерфейс, позволяющий подключаться к интернету. В последней строке возможно, например, `ra0`, а не `wlan0`. Это зависит от того, какой интерфейс по выводу `ifconfig`. Настройки дописываются в конец файла конфигураций.

Затем нужно перезапустить сеть. Это делается командой:

```
/etc/init.d/networking restart
```

3.1.2 Настройка сети для каждого контейнера

Чтобы настроить сеть нужно для каждого нового контейнера выполнить действия:

Нужно запустить контейнер.

Открыть в контейнере файл `/etc/network/interfaces` и привести его к виду:

```
auto lo
iface lo inet loopback
```

Это делается двумя командами:

```
echo auto lo> /etc/network/interfaces
echo iface lo inet loopback>> /etc/network/interfaces
```

loopback интерфейс служит для того, чтобы можно было обратиться к 127.0.0.1

После чего нужно остановить контейнер.

На хосте нужно запустить второе приложение. Скрипт упрощает настройку новых контейнеров. Сперва запрашивается имя, необходимое в настройках по умолчанию, чтобы прописать путь. Затем запрашивается MAC и IP. После чего изменятся файл конфигураций.

Для правильной работы у каждого контейнера должен быть уникальный MAC адрес и IP адрес [6].

После применения настроек запускается контейнер. Теперь контейнер получил статические адреса, настройка сети контейнера завершена.Теперь можно протестировать успешно сеть.

4 Работа с LXC контейнерами для повышения надежности вычислительных систем

Имеется два контейнера, управление которыми доступно только администратору. Пользователь работает с приложением - реплицируемой базой данных и может вывести ее из строя. База данных, с которой работает пользователь находится в основном контейнере. Второй контейнер служит для создания резервной копии данных, позволяющий избежать нецелостности таблиц при резервном копировании. Таблицы основной базы данных не блокируется, и пользователь может непрерывно работать с ней. В случае выхода из строя используемой базы данных, пользователь должен связаться с администратором. Администратор по средствам данной технологии восстановит резервную копию или примет меры по замене жёсткого диска. После чего пользователь будет работать с восстановленными данными.

4.1 Настройка репликации базы данных на контейнерах

База данных, с которой работает пользователь, будет находиться в контейнере `work` — основной сервер с IP-адресом 10.8.8.2. Контейнер `backup` — резервный сервер с IP-адресом 10.8.8.3. Необходимо настроить репликацию базы данных на этих контейнерах.

Выбрана система управления базами данных MySQL, которая прекрасно подходит для поставленной задачи и поддерживает репликацию.

Для создания резервной копии можно воспользоваться утилитой mysqldump. Однако, основным недостатком использования mysqldump при достаточно большой базе, может оказаться логическая нецелостность таблиц в безе данных. Как известно mysqldump может не блокировать таблицы при осуществлении дампа, но тогда во время выполнения дампа, которое занимает продолжительное время, многие таблицы могут быть изменены работающими пользователями, что приведёт к логической нецелостности данных. Либо mysqldump блокирует таблицы, но в этом случае пользователи не смогут работать с базой данных, что во многих случаях недопустимо.[8]

Эти недостатки можно решить с помощью способа резервного копирования при помощи репликации на дополнительный сервер. В данном случае дополнительным сервером является второй контейнер. На основном контейнере настраивается сервер-master. На втором контейнере — сервер-slave. Сервер-

slave постоянно копирует все изменения с сервера-master.

4.2 Работа репликации в контейнерах

Создана база данных в контейнере work. И заполнена тестовая таблица.

При работе репликации данная база появилась в контейнере backup.

Для автоматического создания резервной копии с периодичностью в 1 час с помощью третьего приложения можно воспользоваться приложением cron.

Изначально этого приложения нет в контейнере, поэтому его нужно установить.

Редактирование таблицы планировщика осуществляется с помощью команды:

```
crontab -e
```

В данном случае создание резервной копии будет осуществляться каждый час в 0 минут, поэтому выбран параметр параметр `*/1`. После первых пяти полей указывается путь к скрипту, после чего запись выглядит следующим образом:

```
0 */1 * * * /home/script_backup.sh
```

Теперь скрипт будет выполняться каждый час. После выполнения скрипта создаётся dump-файл, который может быть использован для восстановления базы данных. Если данный файл уже существует, то он заменяется новым.

4.3 Восстановление данных при отказе чтения файла в рабочей базе

В случае выхода из строя базы данных, пользователь связывается с администратором.

Чтобы восстановить базу данных, нужно с правами администратора запустить на хосте скрипт, приведённый четвёртом приложении. В результате работы скрипта, файл можно перенести в любой путь. Путь по умолчанию выбирается вводом `1`. Резервная копия будет скопирована с заменой в домашний каталог основного контейнера.

Чтобы восстановить базу данных нужно зайти с правами администратора в основной контейнер и в MySQL, затем необходимо создать пустую базу данных и из консоли контейнера выполнить команду:

```
mysql -uroot -ppassword test < backup.sql
```

В данном случае `test` — это новая пустая база данных, созданная ранее, `backup.sql` — backup-файл.

После выполнения команды база данных восстановится.

Однако, в случае, если произошла ошибка чтения файла, диск начал сыпаться. В таком случае необходима замена диска. Резервная копия может быть скопирована на новый диск, это можно сделать с помощью четвёртого приложения, выбрав второй пункт, а затем указав нужный путь.

После того, как администратор восстановит базу данных, пользователь может продолжить работу уже с восстановленными данными.

ЗАКЛЮЧЕНИЕ

В ходе бакалаврской работы были рассмотрены высоконадёжные системы, виртуализация. Были показаны основные особенности использования контейнера LXC:

Были созданы контейнеры LXC, разработаны приложения, упрощающие настройку сети.

В результате была настроена сеть, позволяющая контейнерам обмениваться между собой данными.

Была выбрана система управления базами данных MySQL и установлена в каждом контейнере. Была настроена репликация. В главном контейнере находится база данных, с которой работает пользователь. Разработан скрип, позволяющий делать резервные копии. Резервные копии создаются автоматически на резервном контейнере.

Была создана пробная база данных, по истечению времени создалась резервная копия. База данных на главном контейнере была удалена, а затем восстановлена с помощью резервной копии. Создано приложение упрощающее восстановление базы данных, и показано на примере, как происходит восстановление базы данных.

В результате получилась высоконадежная база данных с использованием контейнера.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Отказоустойчивость [Электронный ресурс]. — URL: https://ru.wikipedia.org/wiki/Отказоустойчивость (Дата обращения 28.05.2016). Загл. с экр. Яз. рус.

2 RAID [Электронный ресурс]. — URL: https://ru.wikipedia.org/wiki/RAID (Дата обращения 20.05.2016). Загл. с экр. Яз. рус.

3 LXC [Электронный ресурс]. — URL: https://ru.wikipedia.org/wiki/LXC (Дата обращения 25.05.2016). Загл. с экр. Яз. рус.

4 Установка и настройка LXC на Debian 8 [Электронный ресурс]. — URL: https://habrahabr.ru/post/271537/ (Дата обращения 17.05.2016). Загл. с экр. Яз. рус.

5 Туториал по контейнеризации при помощи LXC [Электронный ресурс]. — URL: http://eax.me/lxc/ (Дата обращения 26.05.2016). Загл. с экр. Яз. рус.

6 Настройка bridge-интерфейса LXC в Debian 8 для NAT [Электронный ресурс]. — URL: http://mnorin.com/nastrojka-bridge-interfejsa-lxc-v-debian-8-dlya-nat.html (Дата обращения 23.05.2016). Загл. с экр. Яз. рус.

7 Маршрутизация IP и форвардинг [Электронный ресурс]. — URL: http://gentoo.theserverside.ru/book/ar68s16.html (Дата обращения 23.05.2016). Загл. с экр. Яз. рус.

8 Резервное копирование нагруженого MySQL-сервера с помощью репликации [Электронный ресурс]. — URL: http://linuxshare.ru/docs/mysql/mysql-bck-repl.html (Дата обращения 05.05.2016). Загл. с экр. Яз. рус.