Санкт-Петербургский государственный университет
Кафедра системного программирования
Программная инженерия

Малец Лев Константинович

Драйвер блочного устройства в ОС Embox на гипервизоре Xen

Отчёт о прохождении учебной практики

Научный руководитель:
Козлов Антон Павлович,
ассистент

Содержание

Введение

Виртуализация сегодня — это важное направление. Она активно применяется, например, в области облачных вычислений.

Некоторый гипервизоры используют паравиртуализацию — технику виртуализации, подразумевающей подготовку гостевой системы для запуска на конкретном гипервизоре, для достижения максимальной эффективности.

Одним из подобных гипервизоров является Xen[1] — гипервизор 1-го типа. Так, например, для устройств ввода/вывода рекомендуется использовать разделённый драйвер, концепция которого предполагает у гостевой системы реализацию специальных драйверов для различных видов устройств ввода/вывода (сетевая карта, блочное устройство). Подобный подход несёт меньше накладных расходов, чем эмулирование.

Unikernel — специализированный образ, содержащий в себе приложение и необходимый для его работы набор библиотек и компонент ядра. Идея unikernel’ов лучше раскрывает себя при использовании гипервизора, так как это предоставляет возможность избавления от большого количества драйверов. Unikenrel’ы могут использоваться для построения микросервисных архитектур.

ОС Ембокс[2] — это операционная система реального времени. За счёт своей модульной структуры она обладает высокой степенью настраиваемости, которая позволяет создавать минимальное по объёму решение для конкретной задачи. Это качество позволяет считать, что из ОС Ембокс можно сделать инструмент для создания unikernel’ов. Так как подобные инструменты, в большинстве своём, поддерживают Xen, было начато портирование ОС Ембокс на Xen.

1. Постановка задачи

Целью данной работы является реализация драйвера блочного устройства для ОС Ембокс на гипервизоре Xen.

Были выделены следующие задачи:

- Обзор устройства драйверов в ОС Ембокс и модели драйверов гипервизора Xen
- Реализация драйвера

2. Обзор

2.1. Модель драйвера блочного устройства в ОС Ембокс

В ОС Ембокс взаимодействие с блочными устройствами абстрагировано от типа устройства. Для работы с ними есть набор обобщённых функций, которые, в свою очередь, полагаются на драйвера конкретных устройств.

На драйвер блочного устройства в Ембокс возложено 3 функции:

- обнаружение устройств, которые могут быть обслужены драйвером
- чтение данных заданного размера в заданный буфер, начиная с заданного блока
- запись данных заданного размера из заданного буфера, начиная с заданного блока

2.2. Модель драйвера блочного устройства гипервизора Xen

Рисунок 1 — разделённый драйвер

Модель разделённого драйвера Xen следующая:

- Гостевая система имеет на борту frontend-часть, который по заданному интерфейсу общается с backend-частью.
- Backend-часть находится где-то в другом домене, который имеет реальный доступ к устройству.

Frontend-часть взаимодействует с backend-частью при помощи трёх механизмов гипервизора Xen:

- *XenStore* — это иерархическое пространство имен, предназначенное для конфигурации и хранения информации о состоянии. Frontend-часть драйвера использует его для обнаружения подключенных устройств и для взаимодействия с backend-частью при инициализации устройств. Через него frontend-часть сообщает такую информацию, как идентификатор *event channel* и идентификатор записи в *grant table*, которая ссылается на страницу, выделенную под буфер для передачи запросов/ответов. Также, с помощью *XenStore*, frontend и backend части сообщают, какие возможности поддерживаются ими.
- *Event channel* — аналог прерываний. Двусторонний канал сообщений, передающий ровно 1 бит информации. С помощью него frontend-часть сообщает о новых запросах, а backend-часть — об обработанных.
- *Grant table* — механизм разделяемой памяти. Позволяет домену указывать, какие страницы и с какими разрешениями доступны другому домену.

Fronend и backend части взаимодействуют посредством запросов. Запрос на чтение/запись содержат информацию о том какие блоки и как должны быть расположены на страницах, выделенных для передачи данных, а также набор идентификаторов записей в *grant table*, которые ссылаются на эти страницы.

3. Реализация

Данный драйвер был выполнен на языке C в виде отдельного модуля ОС Ембокс.

3.1. Обнаружение и инициализация

Обнаружение устройств представляет из себя поиск по *XenStore*. Для каждого типа устройств, в *XenStore* задан путь, по которому располагаются директории всех устройств этого типа. До инициализации в этих директориях присутствует лишь базовая информация: идентификатор домена, в котором находится backend-часть и путь до директории с информацией о backend-части.

Во время инициализации драйвер выделяет нужные ресурсы, такие как *event channel* и страницы. Затем заносит информацию о них в *XenStore*. После ждёт инициализации backend-части и считывает из *XenStore* параметры устройства.

3.2. Страницы

Основным моментом в реализации драйвера является организация использования страниц в запросах.

В решении был выделен пул, который содержит группы страниц. Группы содержат разное количество страниц, и при совершении операции из пула берётся группа нужного размера.

Подобная организация позволяет при инициализации установить флаг *feature-persistent*[3], который сообщает backend-части, что если идентификатор записи *grant table* участвует в двух разных запросах, то эта запись ссылается на одну и ту же страницу. Это позволяет backend-части экономить на операциях map/unmap.

Пул обеспечивает использование групп одного размера в LIFO порядке, что отчасти удовлетворяет рекомендации разработчиков Xen при использовании флага *feature-persistent*, которая говорит об использовании страниц в LIFO порядке.

3.3. Чтение и запись

Операции практически идентичны, за исключением момента копирования из/в заданного буфера.

1. Вычисляется необходимое количество страниц, и из пула берётся группа нужного размера.
2. В случае записи, данные из заданного буфера копируются на страницы из группы.
3. Заполняется запрос, который кладётся в буфер для передачи запросов, и уведомляется backend-часть.
4. Ожидание оповещения о завершении операции от backend-части.
5. В случае чтения происходит копирование прочитанного в заданный буфер.
6. Группа страниц возвращается в пул.

4. Заключение

В ходе работы были выполнены следующие задачи:

- Сделан обзор устройства драйверов в ОС Embox и модели драйверов гипервизора Xen
- Реализован драйвер блочного устройства для ОС Embox на гипервизоре Xen

В существующей схеме работы с блочными устройствами в ОС Embox есть недочёты. Дело в том, что, несмотря на интерфейс драйвера, который допускает чтение/запись произвольного размера, фактически функции чтения/записи драйвера вызываются исключительно по размеру блока. Это связано с устройством кэша. Но размер блока всегда меньше, чем максимальный размер запроса, а это значит, что при чтении/записи больших объёмов данных будет совершено больше запросов, что влечёт дополнительные издержки.

В качестве продолжения данной работы можно было бы улучшить реализацию кэша и схему работы с блочными устройствами в ОС Embox.

Список источников

1 Xen project
URL: https://wiki.xenproject.org/wiki/Main_Page
(дата обращения: 17.01.2020)
2 Ембокс
URL: https://www.embox.rocks
(дата обращения: 17.01.2020)
3 Описание параметров в заголовочном файле
URL: https://xenbits.xen.org/gitweb/?p=xen.git;a=blob;f=xen/include/public/io/blkif.h;h=4cdba79abaeb4244ffefe9f457359af9f8c9c7b0;hb=HEAD
(дата обращения: 05.06.2020)