Tmpa'14, 14.11.2014
Кострома

Подходы к фрагментации системы паравиртуализации

Василий Сартаков,Николай Голиков
ksys labs
{sartakov,golikov}@ksyslabs.org

Agenda

- Введение
- Previous works
- Концептуальный подход
- Дизайн L4Linux
- Фрагментация
- Текущий статус

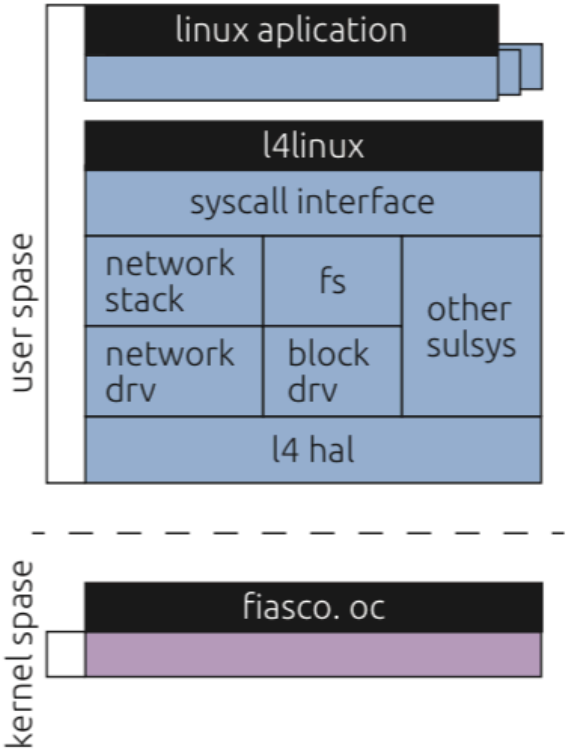
Code reuse

- Не смотря на активные исследовательские работы в области операционных систем новые разработки крайне редко доходят до реального использования
- Реальность рынка программных продуктов такова, что продукты не совместимые с уже существующими или требующие значительных затрат по адаптации существующего ПО так и остаются экспериментальными разработками
- Пример:
  - Доминирование идеологии everything is file
  - Доминирование POSIX
  - В то время как давно уже существует идеология Everything is an Object

Виртуализация

- Виртуализация является решением
- Виртуализируется все:
  - API, ABI, POSIX…
- Механизм паравиртуализации

L4Linux



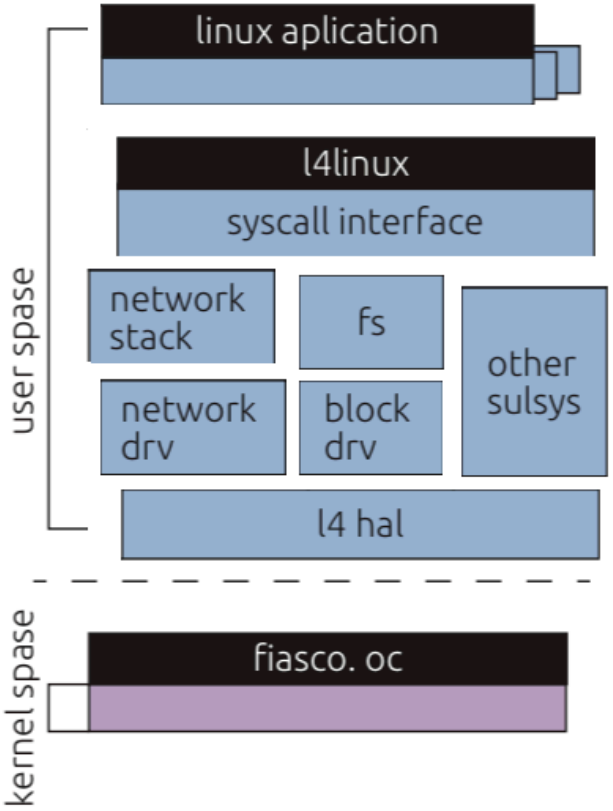
- Перенос ядра Linux в пользовательское пространство
- Запуск двоичных программ Linux в окружении микрядра

Защищенность L4Linux

Сартаков В. А., Тарасиков А. С. Защита микроядерного окружения L4Re от атак на стек // Безопасность информационных технологий. 2014. Т. 4.

- Память под L4Linux выделяется одним регионом (dataspace)
- Невозможно применить ни ASLR, ни стек протекцию
- Синтетические тесты на проникновение выявили уязвимость к атакам на стек

Иными словами

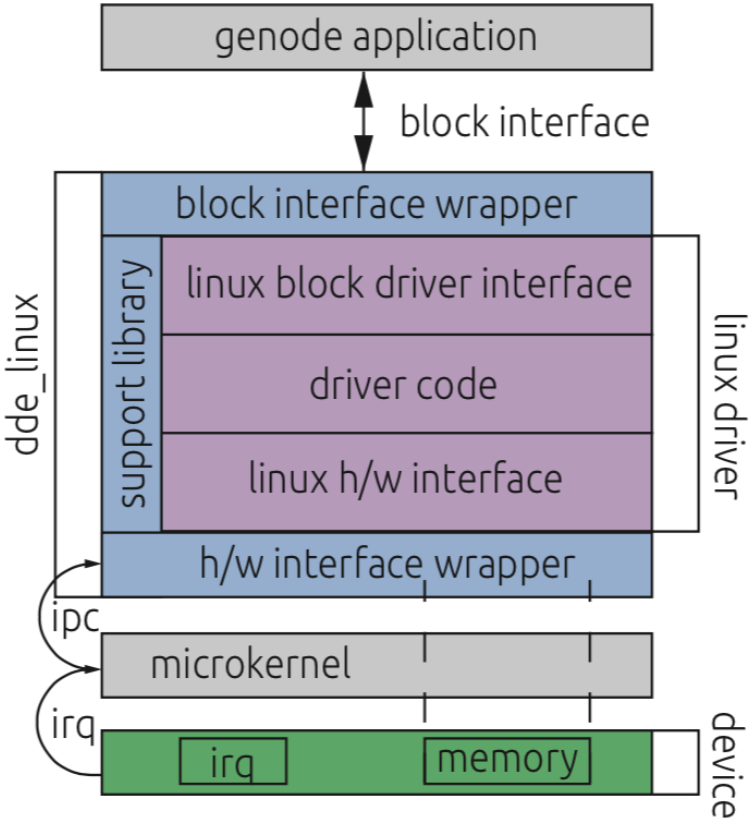


- Необходимо разделение на компоненты (фрагментация)
- С последующим выстраиванием связей

Agenda

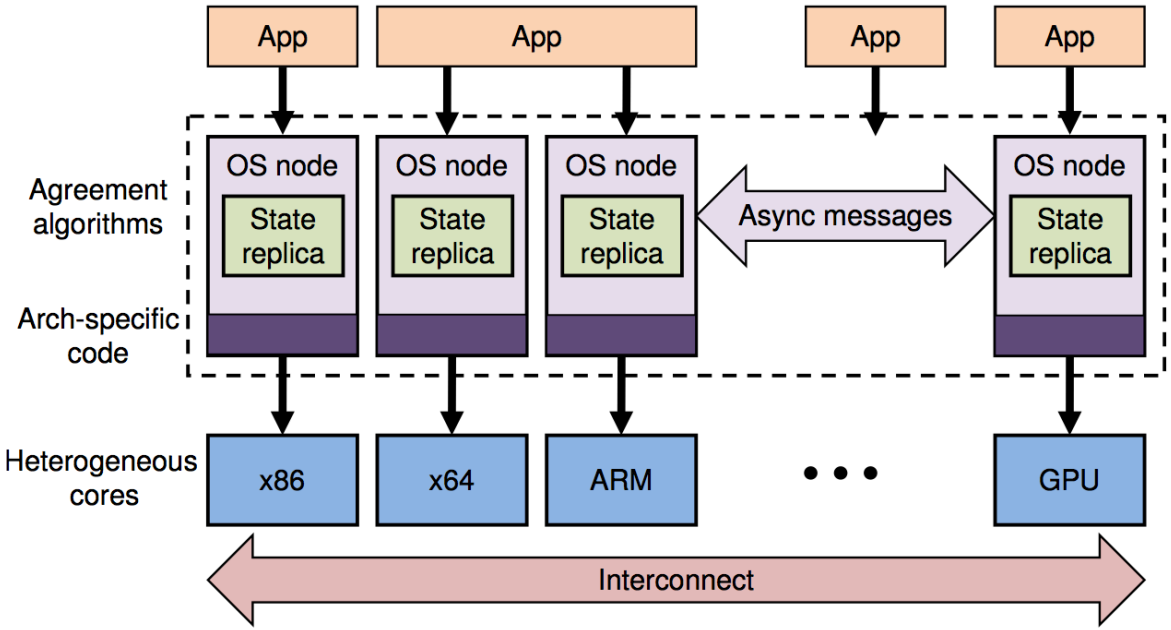
- Введение
- Previous works
- Концептуальный подход
- Дизайн L4Linux
- Фрагментация
- Текущий статус

Device-Driver-Environment Kit



- DDE kit – слой сопряжения на уровне исходного кода
- Weisbach Hannes. DDEKit Approach for Linux User Space Drivers. 2011.

Barelfish



- Решает проблему масштабирования. (проблемы блокировок, memory management)
- Нет коммуникации между ядрами на низком уровне
- Каждое ядро имеет свою копию ОС и общается асинхронно, как элементы в распределенной системе.

Source: http://www.barrelfish.org/barrelfish_sosp09.pdf

seL4, NewtOS, L4::Reap

- Keep net working - on a dependable and fast networking stack / Tomas Hruby, Dirk Vogt, Herbert Bos [и др.] // In Dependable Systems and Networks. IEEE, 2012. С. 1–12.
- Klein Gerwin, Derrin Philip, Elphinstone Kevin. Experience report: seL4: formally verifying a high-performance microkernel // ACM Sigplan Notices / ACM. Т. 44. 2009. С. 91–96.
- The multikernel: a new OS architecture for scalable multicore systems / Andrew Baumann, Paul Barham, Pierre-Evariste Dagand [и др.] // Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles / ACM. 2009. С. 29–44
- Сартаков Василий, Тарасиков Александр. Анализ производительно- сти сетевой подсистемы микроядерного окружения Genode // Tools & Methods of Program Analysis. 2013.

Agenda

- Введение
- Previous works
- Концептуальный подход
- Дизайн L4Linux
- Фрагментация
- Текущий статус

Концептуальный подход

- Избыточная фрагментация
  Разделение на компоненты плюс дублирование
- Рекомпозиция системы
  Группировка компонент на вычислительных ядрах

- Одно ядро ОС
- Минимизация межядерных коммуникаций
- Сборка на ядре минимального набора компонент L4Linux для поддержки работы конкретной программы

Agenda

- Введение
- Previous works
- Концептуальный подход
- Дизайн L4Linux
- Фрагментация
- Текущий статус

Дизайн L4Linux

- L4Linux как часть окружения
- Доступ драйверов к устройствам
- Запуск программ

Доступ к устройствам

- Отображение физической памяти устройств в виртуальное адресное пространство L4Linux
- L4Linux может запросить у окружения и процесса Init прерывание. Для этого в контексте исполнения создается специализированный поток регистрируемый в системе как обработчик прерывания. Доставка прерывания осуществляется через доставку сообщения с разблокировкой потока прерывания

Запуск программ

- Программы L4Linux являются программами окружения L4
- Все системные вызовы программы обрабатываются как page fault. Каждой программе назначается ядро L4Linux в качестве обработчика page faults.

Agenda

- Введение
- Previous works
- Концептуальный подход
- Дизайн L4Linux
- Фрагментация
- Текущий статус

Previous

- Извлечение подсистем :
  - Dde_linux
  - Dde_ipxe
  - Dde_oss
  - Rump kernel

…Используются через слои сопряжения

Обобщенный подход

- Выбирается изолированная подсистема ядра Linux
- Анализируется API подсистемы и зависимости в заголовочных файлах и других подсистемах
- Зависимые заголовочные файлы вместе с исходным кодом извлекаются исходного кода ядра Linux
- API подсистемы накладывается на механизм сообщений микроядерного окружения.

В нашем случае – необходима реализация интерфейса взаимодействия другими компонентами L4Linux.

Подход::минимальный сборка

- Вместо извлечения подсистем и создания слоя сопряжения для работы подсистемы в окружении микроядра, мы извлекаем минимальный набор компонент L4Linux необходимых для работы конкретной программы, разбиваем их по адресным пространствам и группируем вместе с программой на одном ядре
- Минимальный набор подсистем фрагментированного L4Linux:
  - загрузчик процессов
  - диспетчер системных вызовов
  - обработчик page fault

Agenda

- Введение
- Previous works
- Концептуальный подход
- Дизайн L4Linux
- Фрагментация
- Текущий статус

Текущий статус

- Ручное разделение L4Linux на компоненты
- Извлечение криптографической подсистемы, блочного ввода-вывода, сетевого стека и драйверов для proof of concept