Unikernels, microkernels и недалекое будущее виртуализации

Василий Сартаков

ksys labs, TU-Braunschweig

Agenda

- About
- Unikernels
- Intel SGX

About

- МИФИ
- 2004-2007 ЦОС, 2007-2008 RTSoft/Montavista, 2009-2011 EbookApplications, 2011+ ksys labs, 2013+ TU-Braunschweig
- Ksys labs:
  - Деятельность:
    - Экспериментальные разработки
    - Прикладные исследования
    - Трансфер технологий
  - Области:
    - Микроядра (L4RE, Fiasco.OC, NOVA, Genode, seL4)
    - IDS (Suricata)
  - Open source

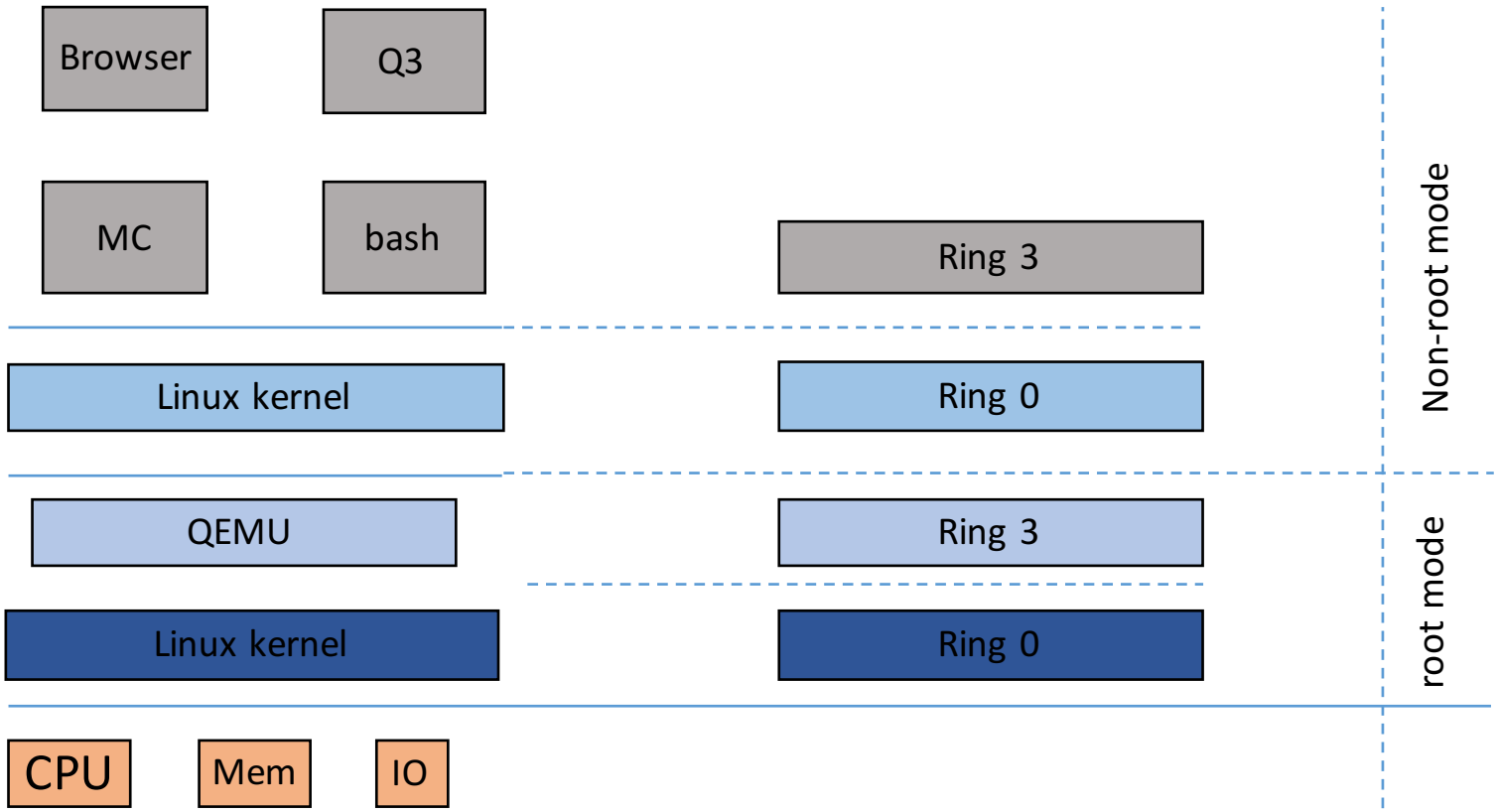
Немного о прошлом..

- В 2005 мне довелось поработать пол года в [.m]
  - Датацентр на 8ого марта и на автозаводе
  - Collocation, dedicated servers, web hosting
  - Развертывание BSP
- Оборудование
  - Очень, очень дорогостоящее
  - Размещение дорогое
  - Все очень, очень дорого
- Сервера набивались под завязку..

Время шло вперед…

- Что в 70ые наука, в 00х – технологии
- Intel VT-x
  - VmWare
  - Xen
  - KVM
- Бизнес:
  - Аппаратная виртуализация
  - Контейнерная виртуализация
  - Обычный виртуальный хостинг
- Виртуальные машины:
  - Быстрый старт
  - Ограничение в вычислительных ресурсах

Стек


Browser
Q3
MC
bash
Ring 3
Linux kernel
Ring 0
QEMU
Ring 3
Linux kernel
Ring 0
CPU
Mem
IO
Non-root mode
root mode

Одна машина – один сервис?

- Виртуальные машины:
  - Изоляция
  - Скорость
  - Реконфигурация на лету (+/-)
- Раньше один сервер и очень дорого, теперь множество виртуальных серверов очень дешевых!
- Изменение методологий разработки и развертывания ПО
  - Требования к производительности и инструментарию

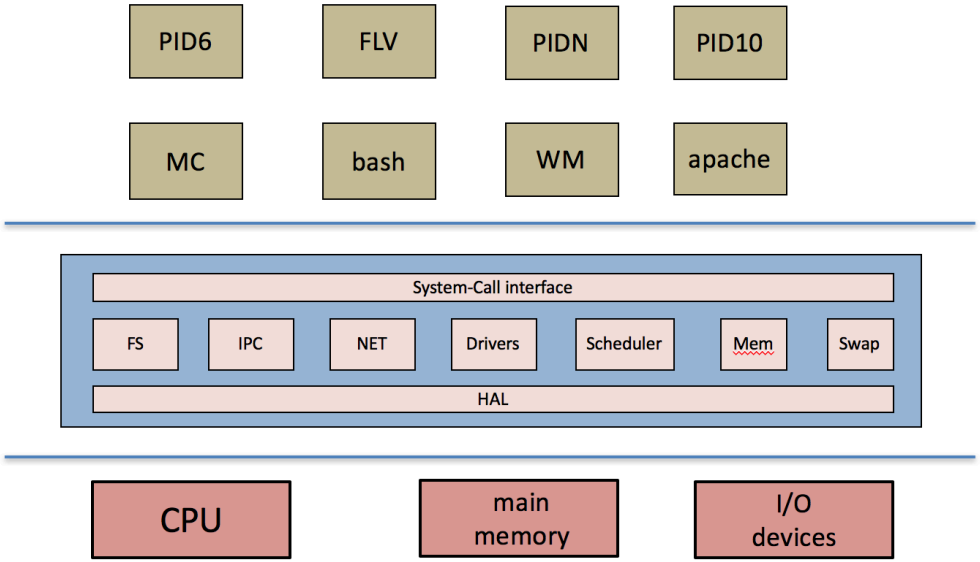
Одна машина – один сервис!

- Game changer.
- Было:
  - ОС – изоляция процессов, управление памятью, защита данных, доступ к устройствам, управление другими ресурсами
- Стало/будет:
  - Меньше технологий (компонент, функции ядра)
  - Зачем нам планировщик, процесс, блочный ввод-вывод, планировщик I/O, etc?
- Как должна выглядеть гостевая ОС?

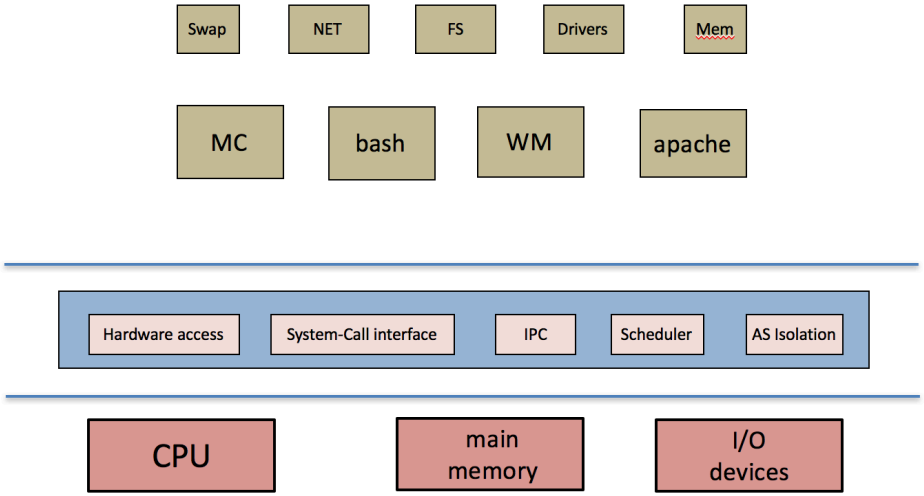
Опять экскурс в историю

- Все новое – хорошо забытое старое
- Unikernels:
  - Exokernel
  - Nemesis
- Экзоядро: концептуальное развитие микроядра

Опять экскурс в историю

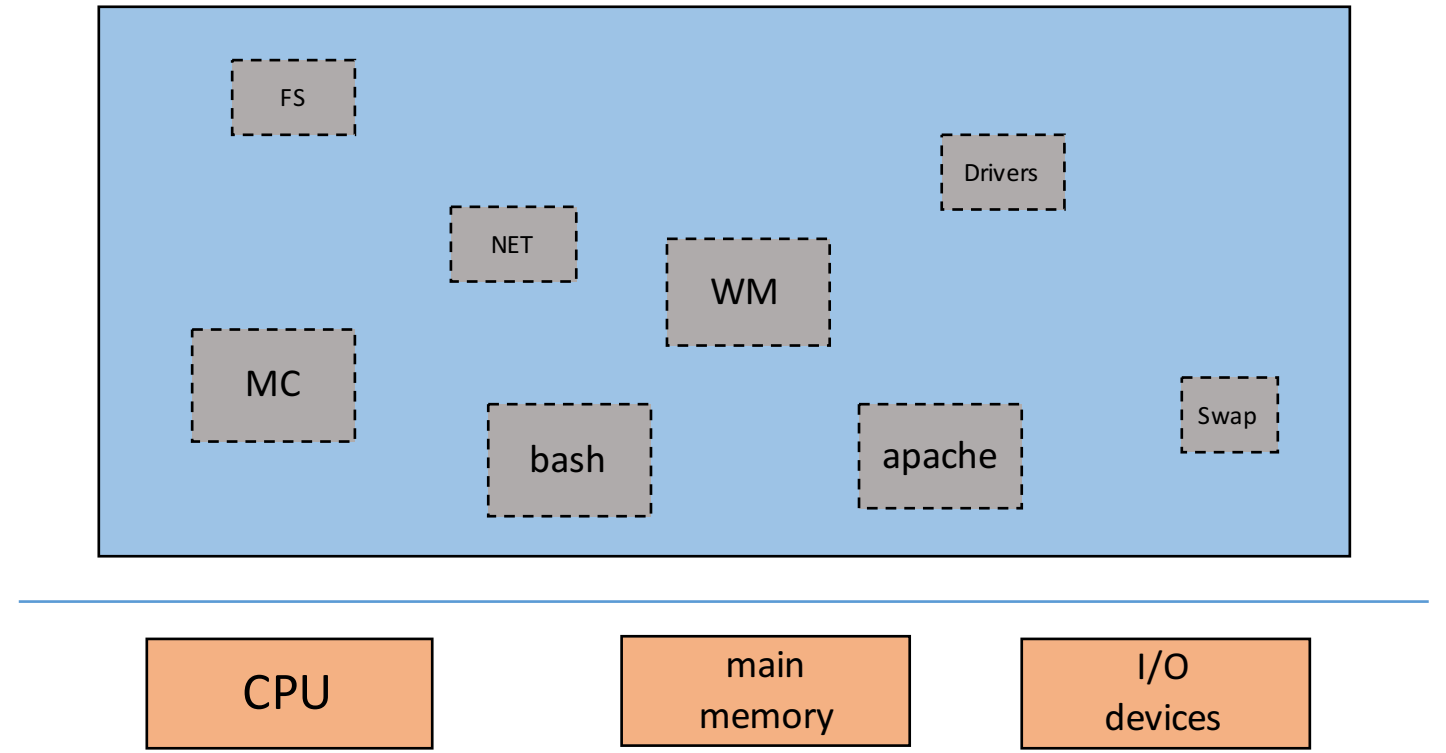


Монолит



Микроядро

Экзоядро/libOS/Unikernel



Pros:

- Производительность
- Предсказуемость производительности

Cons:

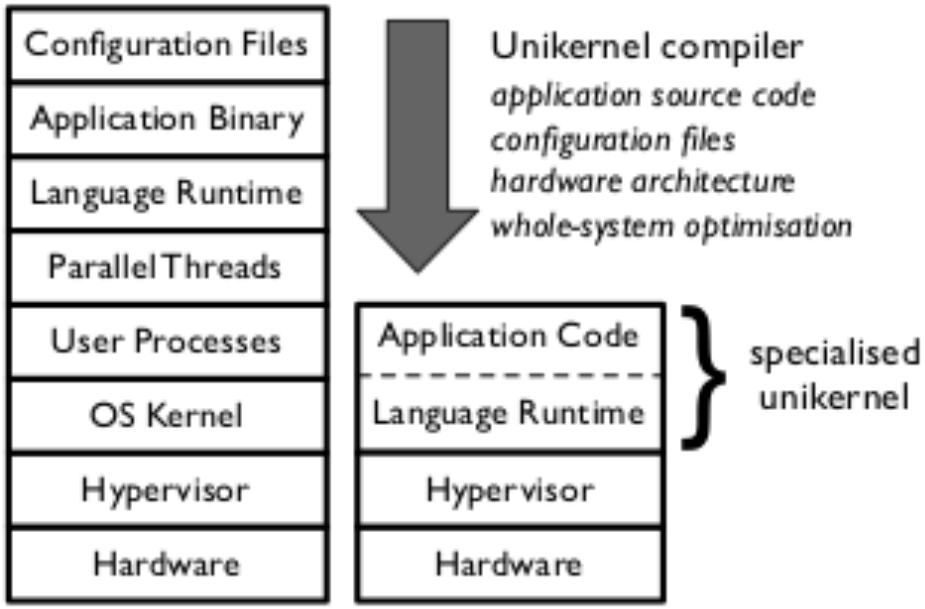
- Безопасность
- Аппаратная поддержка (драйвера)

Проекты

- VxWorks
- eCOS
- SingularityOS

….Но причем тут виртуализация?

Виртуализация и unikernels



Source: https://mirage.io/wiki/technical-background

Pros:

- Производительность
- Предсказуемость производительности
- Изоляция
- Безопасность
- Быстрый старт
- Поверхность атаки/footprint

Современные проекты

Unikernel	Язык	Платформы
Mirage	OCaml	Xen, kFreeBSD
Drawbridge	C	Windows “picoprocess”
HaLVM	Haskell	Xen
ErlangOnXen	Erlang	Xen
OSv	C++/Java	Xen, KVM
GUK	Java	Xen
NetBSD Rump kernel	C	Xen, Linux, POSIX
ClockOS	C++	Xen

Source: Unikernels: Rise of the Virtual Library Operating System

Современные проекты

Mirage:

- OCaml
- Статическая проверка типов
- Автоматическое управление памятью
- Модульная структура
- «Метапрограммирование»
- Xen

OSv:

C++-11

Основная идея – приложения для Linux только лучше

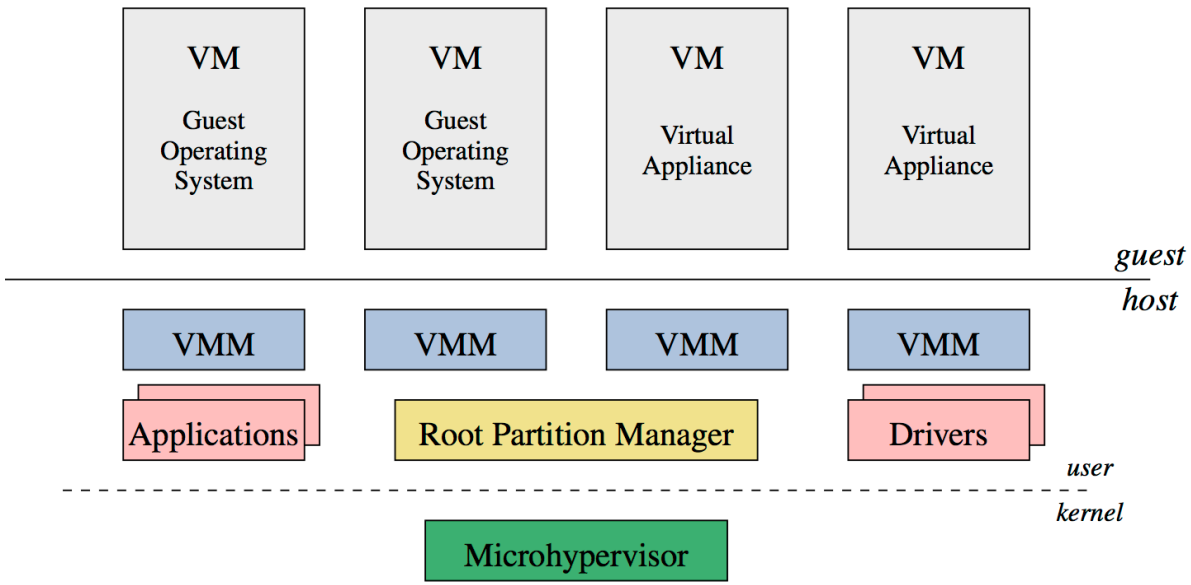
Capstan – управление, сборка, развертывание

REST интерфейс

Минимальные изменения

Кстати, причем тут микроядра?

NOVA и пыльные облака



- Микроядра, а конкретно микрогипервизоры – отличная платформа для массовой виртуализации
- Capability-based security
- изоляция компонент
- защита от вторжений (IO-MMU)
- высокая производительность
- минимальный TCB

Source: NOVA: A Microhypervisor-Based Secure Virtualization Architecture

NOVA + ??

- К сожалению NOVA – это только ядро
- Нужно окружение
  - Genode (VirtualBox as VMM)
  - NUL (Vancouver VMM)
  - NRE
- компоненты окружения, хотябы libvirt – нужно портировать и адаптировать. Есть базовые требования (миграция, виртуальные сети, удаленное управление) и это все нужно писать с нуля.
- Суровые платформы – рейды, драйвера, сетевые карты. Это очень, очень много экспериментальных разработок. Для Linux это делают разработчики оборудования и Enterprise дистрибутивов. На энтузиазме не вытянуть.

Конец первого акта

Мотивация

- При размещении в [.m] приходилось доверять датацентру, конкретно мне.
- У меня (или любого другого сотрудника) доступ к DC, мы всегда можем перегрузить сервер в single user mode, вытащить любые данные, заменить любые данные, загрузить зловред, залить свое ПО.
- TPM – trusted platform module, HSM (Hardware Secure Modules), доверенная загрузка, пломбирование сервера, установка камеры – это все затратно.
- А если вспомнить про «облака», микросервисы, виртуализацию -- гипервизору приходится доверять как приходится доверять оборудованию и персоналу.

Вторжения

Ноябрь 2013: Эскалация привилегий Hyper-V

Октябрь 2014: Доступ одних виртуальных машин Xen к другим

Май 2015: «Venom» - эскалация привилегий в Xen, KVM

Облака

- Мы хотим исполнять свои приложения/сервисы/виртуальные машины изолировано:
  - Гарантия целостности и конфиденциальности
  - Детерминированное исполнение
  - Коммуникация с другими сервисами
  - Отсутствие вторжения со стороны других приложений/сервисов/VM
- Модель угроз
  - Мы не доверяем провайдеру и его сотрудникам
  - Все ПО провайдера – зловредно
  - Мы не доверяем аппаратуре провайдера

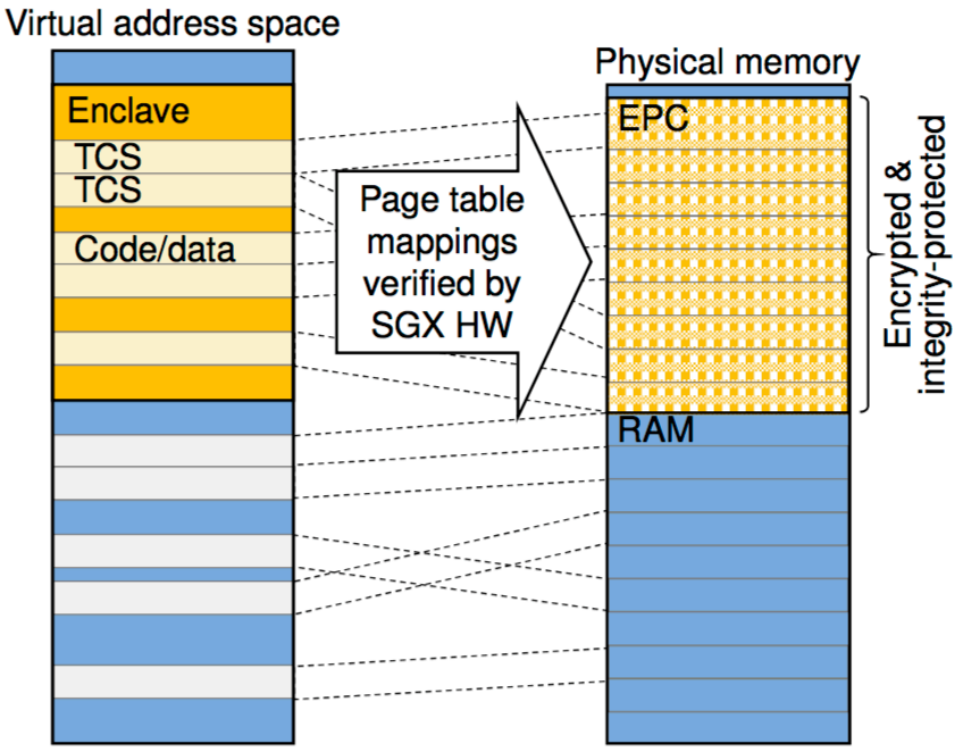
Intel SGX

- Intel Software Guard Extensions
- Развитие технологий виртуализации VT-x, VT-d
- Инверсионная изоляция (inverse sandbox)
  - Аппаратно-поддерживаемые анклавы (enclaves)
  - Новые инструкции процессора
  - Шифрование памяти (нет, это не гомоморфное шифрование)
  - Удаленная аттестация

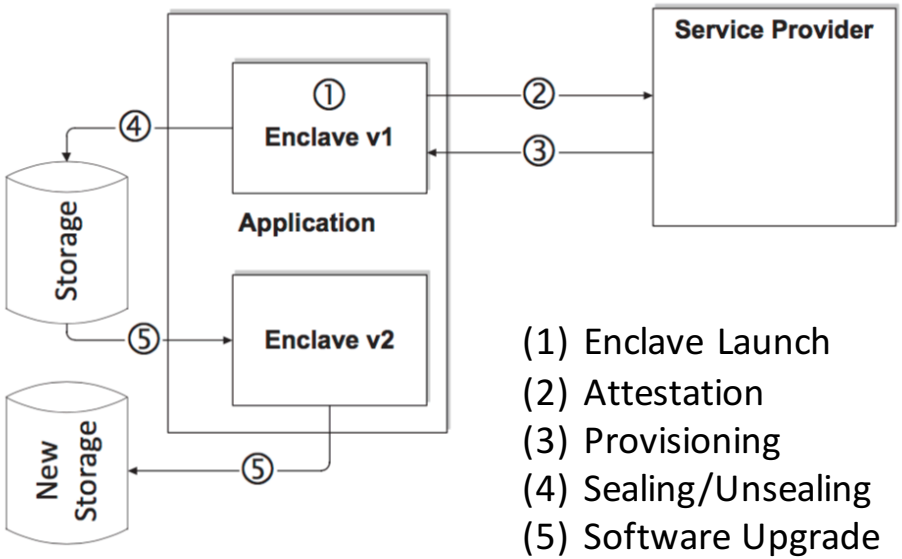
Концепция

- Мы создаем в ОП область изолированную от внешнего мира.
- Эта область зашифрована, благодаря сложной ключевой системе доступа к этому региону у нас нет, но в него может загрузить зашифрованные данные пользователь
- Регистровый файл шифрован, память шифрована, с внешнего мира (ядро, системное/прикладное ПО) доступа нет. Контекст анклава сам хранится в анклаве.
- Анклав можно обновить, он может сохранять шифрованные данные на носитель, поддерживать связь с клиентом, выполнять удаленную аттестацию

SGX



Source: Shielding Applications from an Untrusted Cloud with Haven



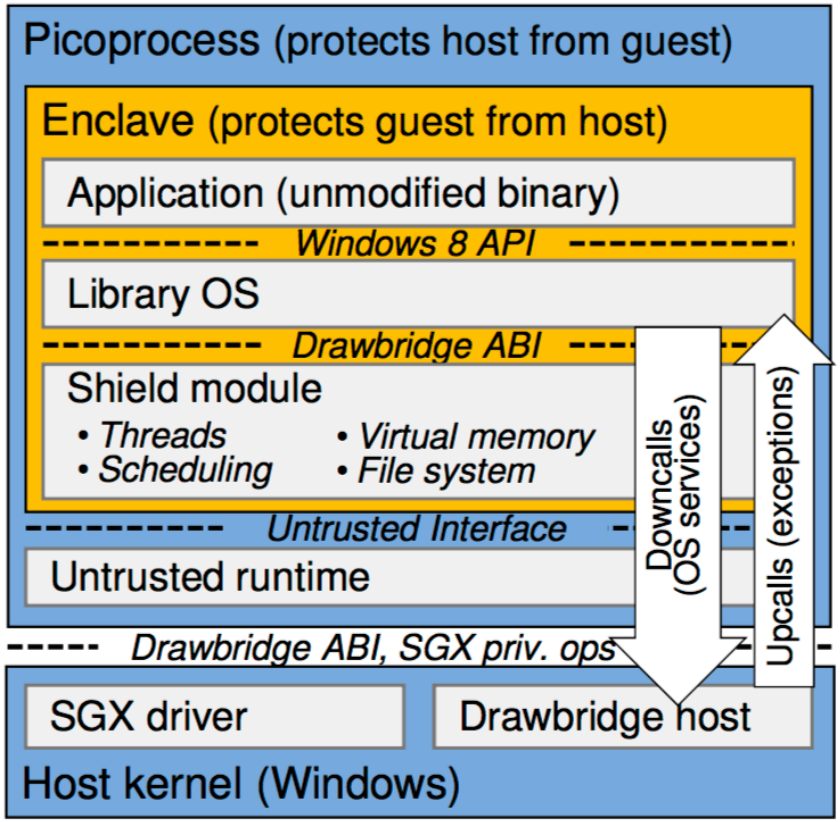
(1) Enclave Launch
(2) Attestation
(3) Provisioning
(4) Sealing/Unsealing
(5) Software Upgrade

Source: Innovative Technology for CPU Based Attestation and Sealing

Haven

- Shielding Applications from an Untrusted Cloud with Haven
- Andrew Baumann, Marcus Peinado, and Galen Hunt, Microsoft Research
- Drawbridge *LibOS*

SGX 2.0 на подходе!



На правах заключения

- Microservices – i+1 шанс для микроядер. Теперь в качестве платформы для вирутализации
- Microservices - прикрасный шанс для развития unikernels
- SGX от Intel – шаг в сторону безопасности и защищенности, новые возможности для защищенных microservices

References

- Engler, Dawson R., and M. Frans Kaashoek. *Exokernel: An operating system architecture for application-level resource management*. Vol. 29. No. 5. ACM, 1995.
- Madhavapeddy, Anil, et al. "Unikernels: Library operating systems for the cloud." *ACM SIGPLAN Notices*. Vol. 48. No. 4. ACM, 2013.
- Kivity, Avi, et al. "OSv—optimizing the operating system for virtual machines." *2014 usenix annual technical conference (usenix atc 14)*. 2014.
- Anati, Ittai, et al. "Innovative technology for CPU based attestation and sealing." *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*. Vol. 13. 2013.
- Baumann, Andrew, Marcus Peinado, and Galen Hunt. "Shielding applications from an untrusted cloud with haven." *ACM Transactions on Computer Systems (TOCS)* 33.3 (2015): 8.
- Madhavapeddy, Anil, and David J. Scott. "Unikernels: Rise of the virtual library operating system." *Queue* 11.11 (2013): 30.

Спасибо.

Sartakov@ksyslabs.org