PKI-форум 2024

Криптография внутри микроядерной ОС: разрешить ^ нельзя использовать

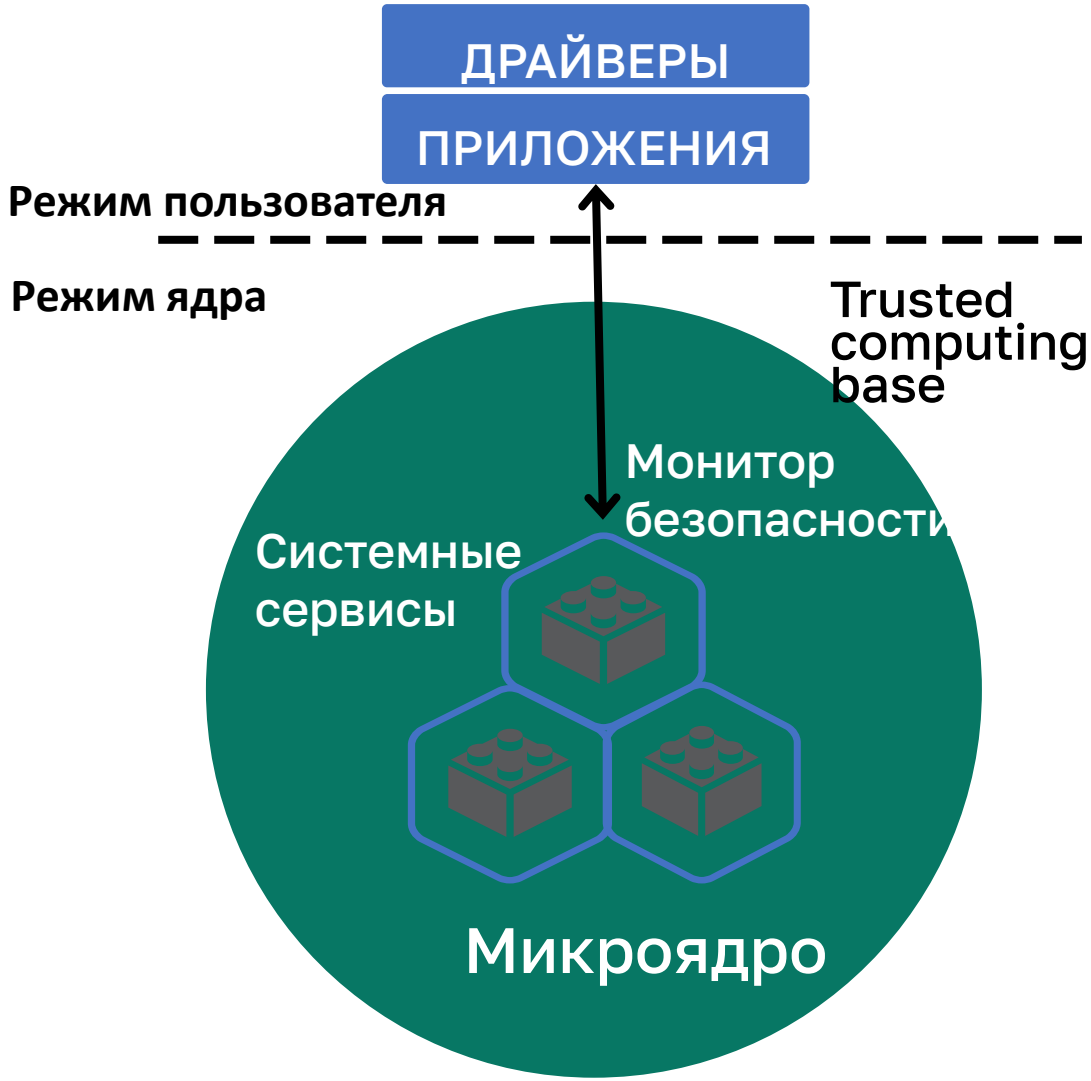
Поликарпов Александр
Старший системный аналитик
АО «Лаборатория Касперского»

kaspersky

ДА, мы разрабатываем операционную систему

Специфика:

- Микроядерная архитектура ОС
  Собственное ядро
- Разделение на домены безопасности (FLASK)
- Запрет любых действий не предопределённых политиками безопасности



Сценарии использования криптографическ их механизмов в ТСВ

TLS

Аутентификация

Полнодисковое шифрование

Проверка образа (ядра) ОС на этапе загрузки, не полагаясь на загрузчики сторонних разработчиков (защита от подмены);

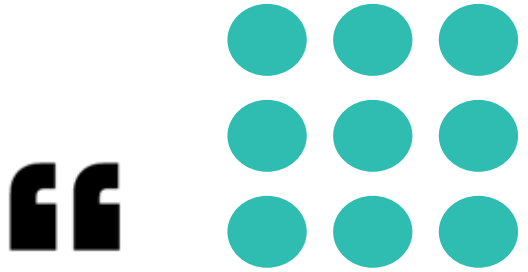
Проверка запускаемых сервисов и приложений (ALM - application lifecycle management) и последующий периодический контроль их функционирования;

Проверка файлов и контроль данных пользователей (предоставление API на уровень продуктовой логики).

Электронная цифровая подпись

ГОСТ Р 34.10-2012

- Готовность инфраструктуры к формированию подписи образов и компонент (с использованием СКЗИ).
  - Возможность хранить на устройстве только открытые ключи.
    - Возможность использовать списки отозванных сертификатов (Certificate revocation list).



Ivan Ristić

PKI is a field that requires deep expertise and dedication;
mistakes are easy to make.

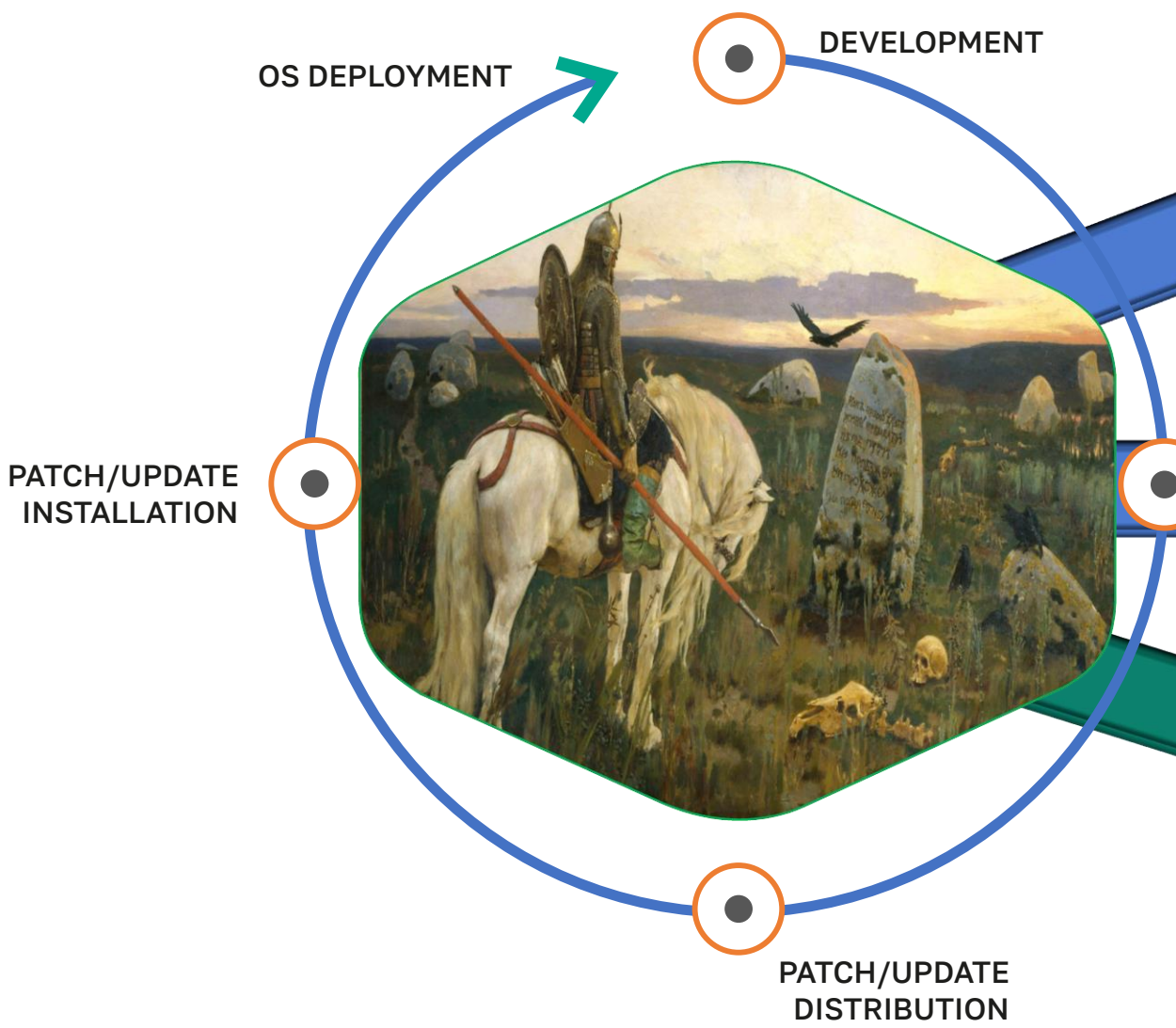
- Производительность

(Время запуска приложения <2 секунд).

  - Не решает проблему доступа злоумышленника к устройству.

	без проверки ЭП, мс	с проверкой ЭП, мс	Размер пакета, б
Browser	3178	3964	11186176
Calculator	2554	3127	8179712
Large	2113	8248	105598320

Криптография уязвимости ТСВ



Универсальный криптографический API (Microsoft crypto API)

Стандартизированный интерфейс (PKCS#11)

API *публичной библиотеки* (OpenSSL)

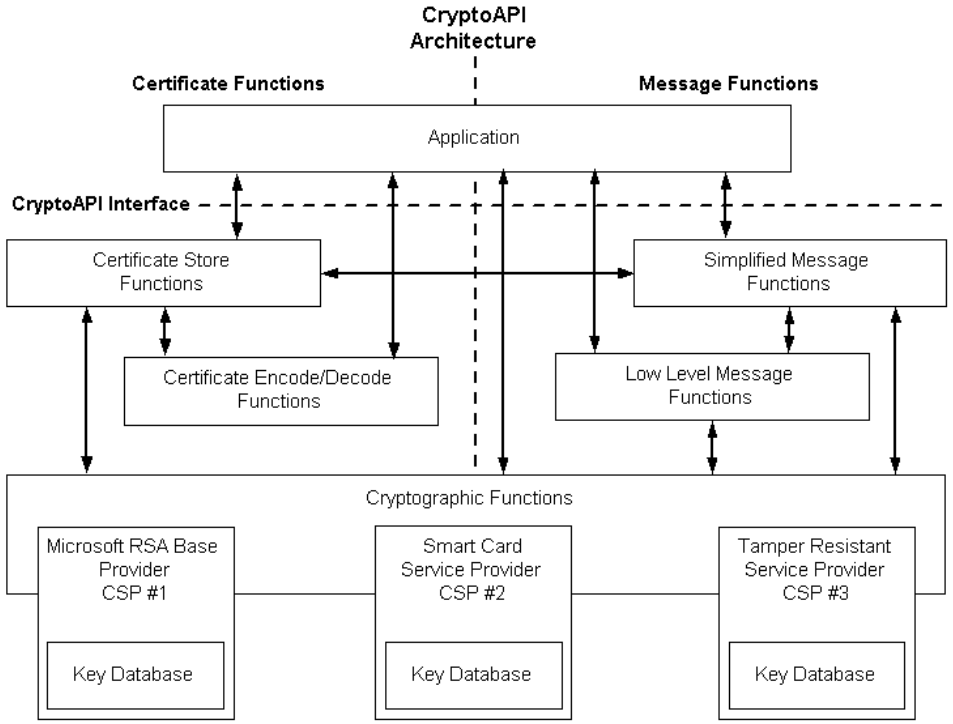
“ *Микросерв* ”

с упрощенным интерфейсом

Универсальный криптографический интерфейс

- Высокая стоимость разработки **«инженер—тысячелетие»**
- Портирование приложений и доверие клиентов
- Сложности с интеграцией с сертифицированными СКЗИ

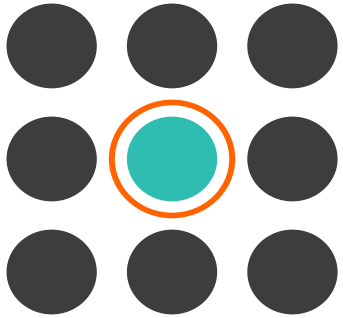
FAR FAR FUTURE



https://learn.microsoft.com/

Стандартизированный интерфейс

- OpenSSL (library & API)
- PKCS #11

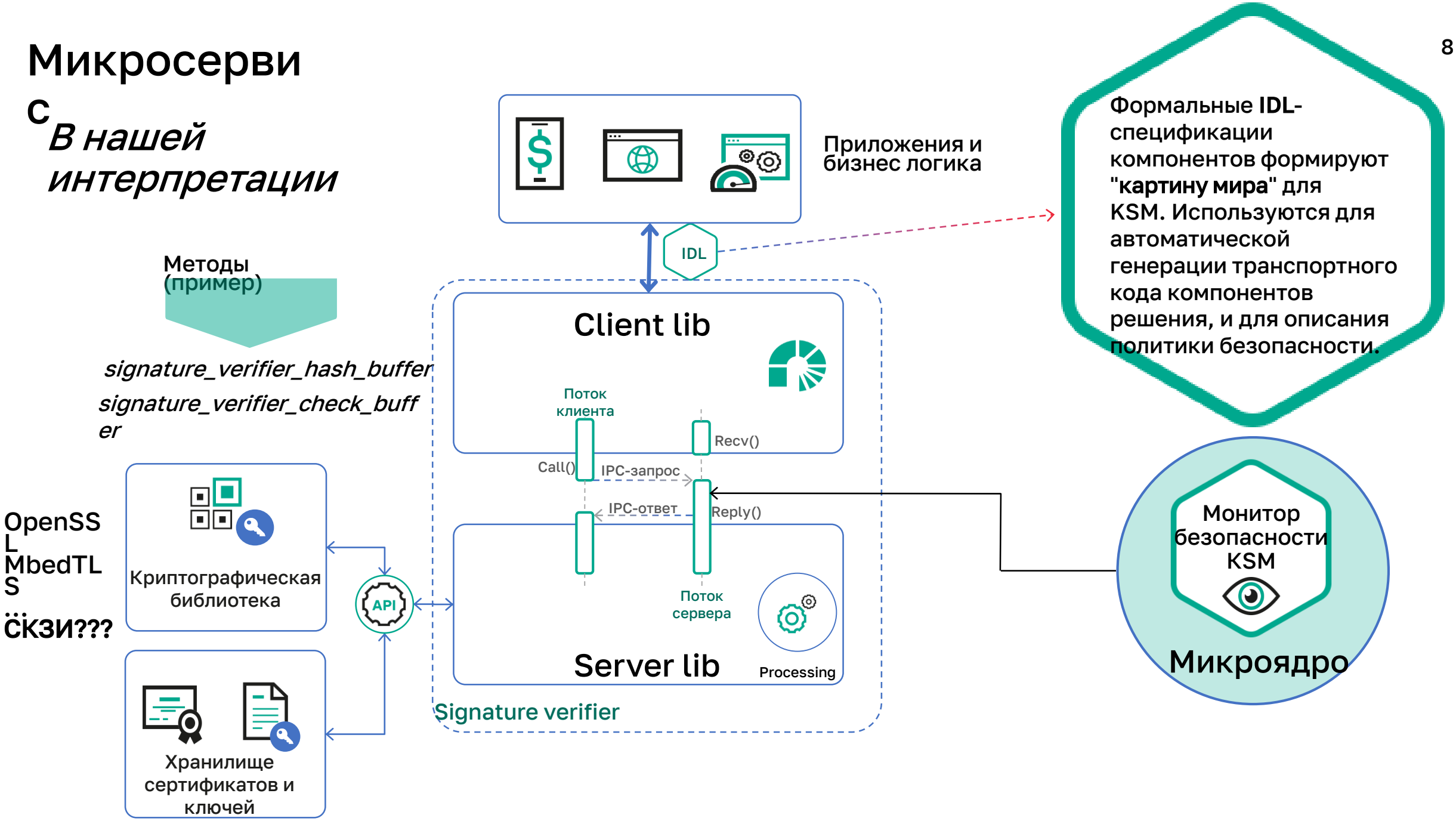


- Уже есть опыт отладки и использования
  - Множество приложений используют
    - Пройден путь исправления CVE

- Высокий риск ошибки при использовании API криптобиблиотек.
  - **Сложный API OpenSSL;**
    - **Неочевидные наименования методов:**
    - **некоторые пользователи иногда отключают проверки сертификатов.**
- **Геополитический фактор**

LibreSS
L

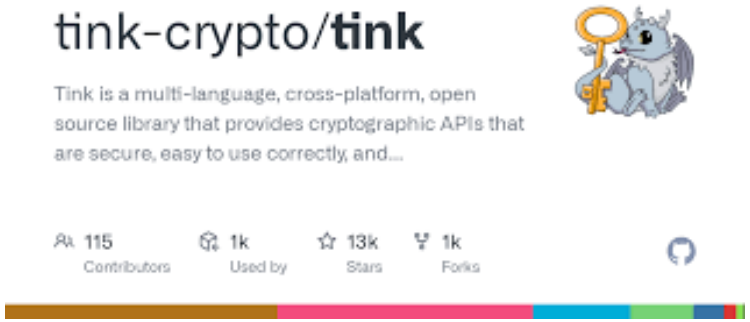
Микросерви
с *В нашей интерпретации*



Преимущества:

Возможность изменять внутреннюю логику сервиса и используемых криптографических «back-end» библиотек без необходимости менять логику приложений (при условии сохранения IDL);

(*Нам кажется*) Для испытательной лаборатории такая концепция упрощает понимание «границ» сервисов, реализующих функции безопасности, рассматривать все решение как совокупность таких элементов;

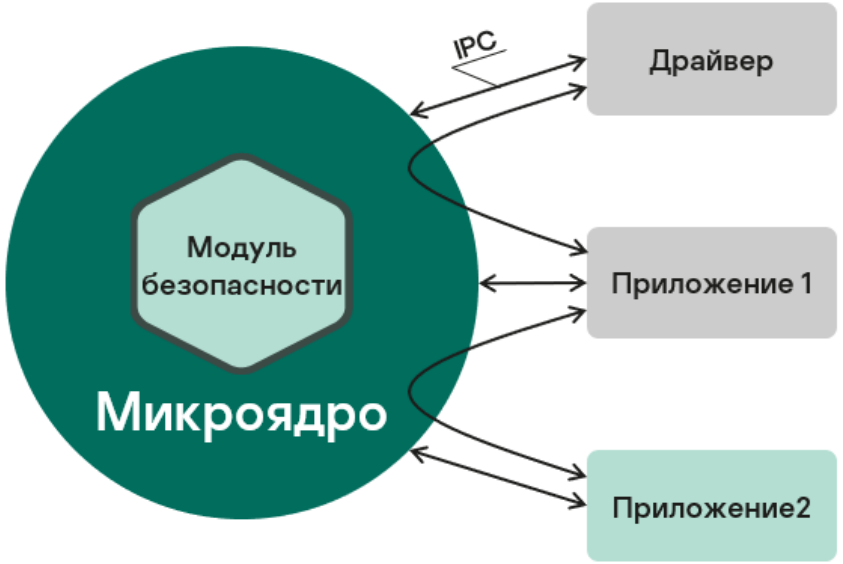


Недостатки:

Наращивание функциональности приведёт к разрастанию количества «микросервисов» или усложнению их интерфейса.

Формальная верификация и тестирование

TLA+ (temporal logic of actions) – язык спецификаций для описания свойств и поведения систем



Формулирует свойства **непротиворечивости** и **консистентности**, которые проверяются автоматически с помощью инструмента проверки моделей **TLC**

Фаззинг-тестирование

Стремимся - Code coverage – 80%

Динамический анализ

Интеграция санитайзеров

(AddressSanitizer; UndefinedBehaviorSanitizer)

в сборку с компонентами «доверенной кодовой базы»

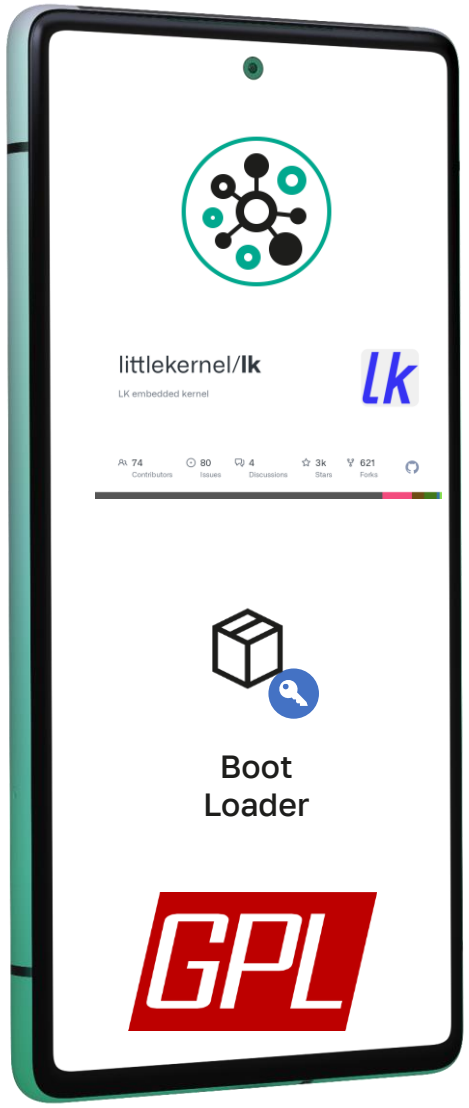
Любые вызовы в политике безопасности должны быть ассоциированы с субъектом, определены атрибуты (чтение\запись)

ГОСТ. Р 56939-2016

Статический анализ

Статический анализатор «Svace» – интеграция в конвейер, устранение критических «срабатываний» (major, critical)

kos_preloader



BARE METAL

Signature verifier

kos preloader

Монитор безопасности KSM

Микроядро

- Где хранить?
- Как обновлять?
- Как отзывать?

Здесь мог быть слайд о СКЗИ как части ОС

ОС как *среда функционирования СКЗИ.*

Интеграция решений вендоров СКЗИ.

разрешить ^ нельзя использовать = нужно использовать

А разрешать или нет ответит регулятор

отвечает нейросеть

Для решения задачи криптографии внутри микроядерной операционной системы необходимо понимать требования безопасности и конфиденциальности данных. Вот несколько шагов для разрешения проблемы:

1. Определение целей и задач: Установите четкие цели, какие данные должны быть защищены и от каких угроз. Это поможет выбрать подходящие инструменты и методы.
2. Анализ уязвимостей: Проведите анализ существующих уязвимостей и потенциальных угроз. Определите, какие атаки могут произойти и как их предотвратить.
3. Выбор криптографических методов: Выберите соответствующие алгоритмы шифрования и хэширования для защиты данных. Рассмотрите такие методы, как AES, RSA, SHA-256 и другие.
4. Реализация методов: Внедрите выбранные криптографические методы в микроядро операционной системы. Это может включать интеграцию библиотек и модулей, а также настройку параметров безопасности.
5. Тестирование и аудит: После внедрения проведите тестирование безопасности и аудит системы. Проверьте, что все необходимые меры приняты и работают корректно.
6. Мониторинг и обновление: Обеспечьте регулярный мониторинг системы и обновляйте ее при необходимости. Это позволит своевременно обнаруживать и устранять возможные угрозы.
7. Управление ключами: Разработайте стратегию управления ключами и сертификатами. Это включает генерацию, хранение и управление ключами для шифрования данных.

Эти шаги помогут создать безопасную систему, которая будет эффективно защищать данные внутри микроядра операционной системы.

Вопросы?

Александр Поликарпов | Старший системный аналитик | KasperskyOS Team **|** Kaspersky
Office: +7 495 797 8700 | Mobile:+7 977 845 11 25 | Alexander.Polikarpov@kaspersky.com
Business Centre “Olimpia Park”, 39A/3 Leningradskoe Shosse, Moscow, 125212, Russia | os.kaspersky.com | www.kaspersky.com
KasperskyOS. Be immune.

kaspersky 25 years DREAM TEAM. REAL PEOPLE.