Улучшения кода ранних этапов загрузки ядра ОС Linux

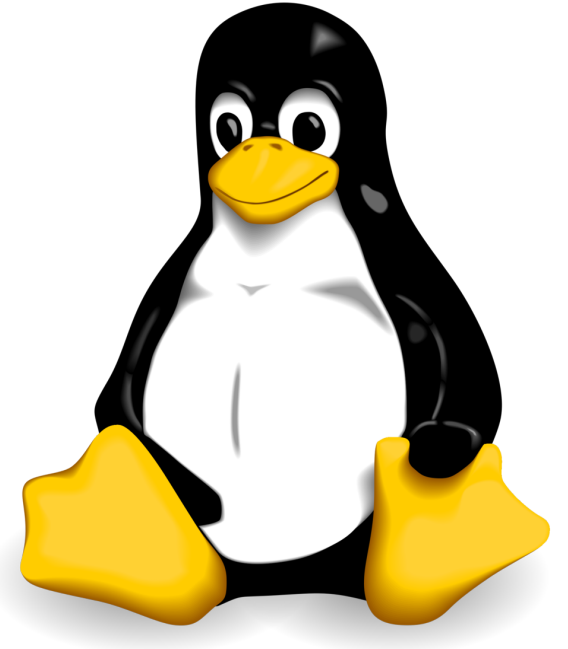
Басков Евгений

Москва
2023

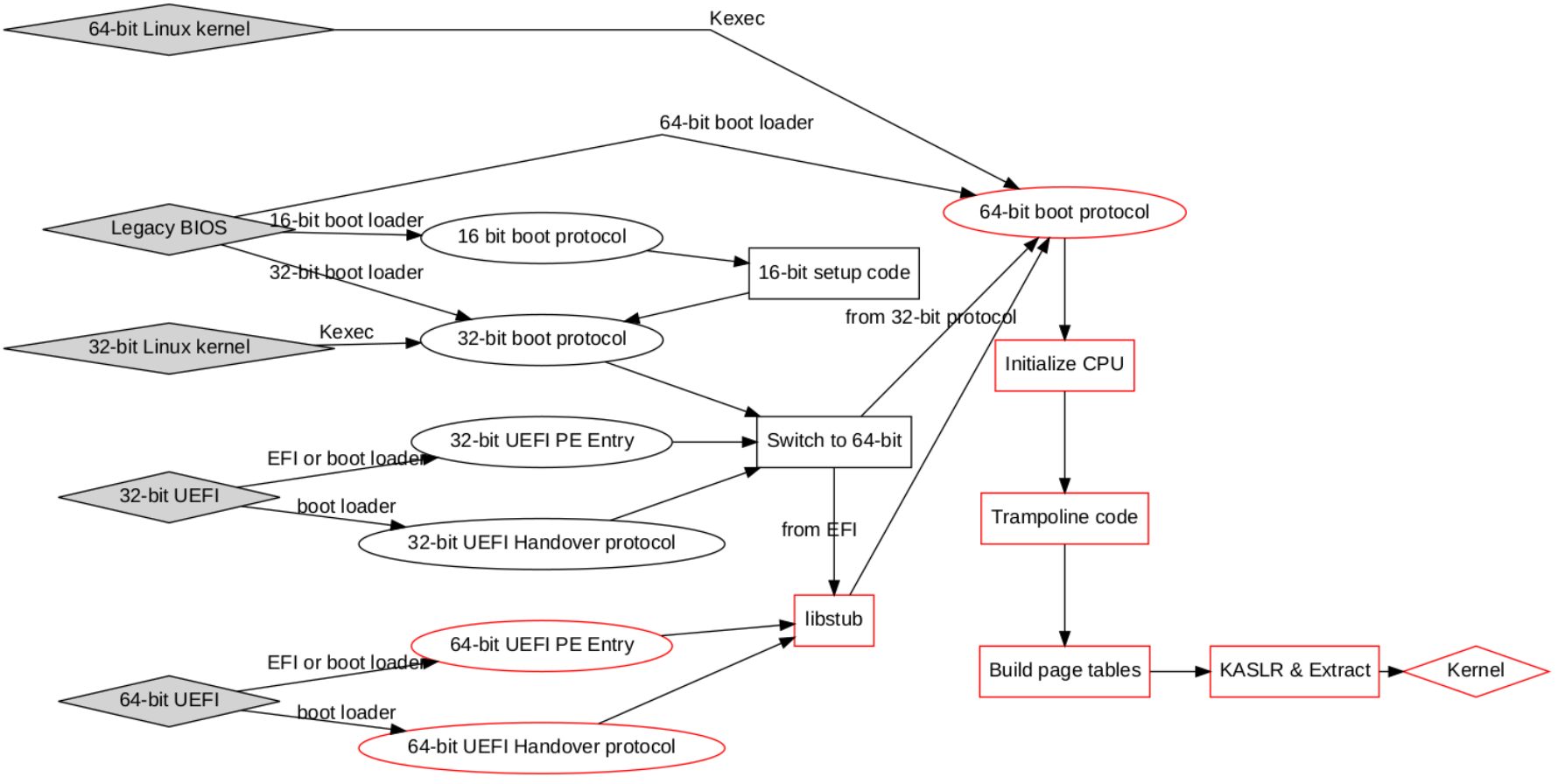
ИСП РАН

Введение

- Большая поверхность атаки, т.е. большое количество потенциально уязвимых мест
  - Большой объем кода
  - Взаимодействует с окружающим миром
- Критический для безопасности компонент
  - Высокие привилегии
  - Практически неограниченный доступ к ресурсами

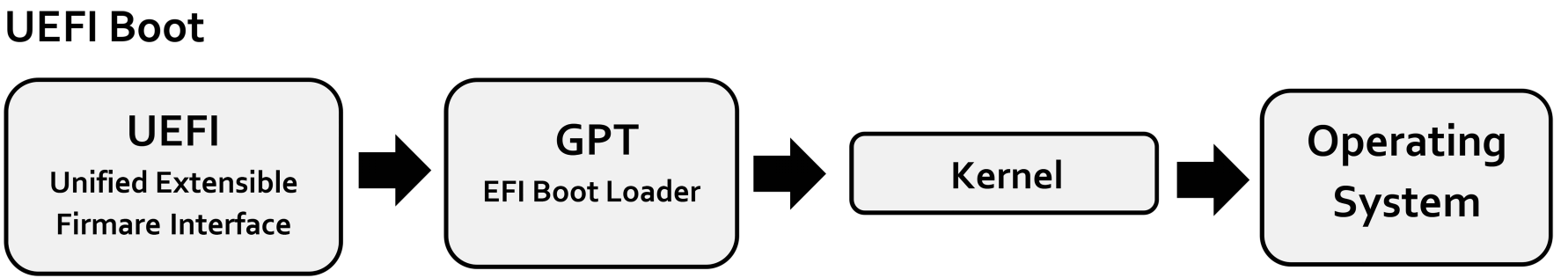


Ранняя загрузка ядра Linux



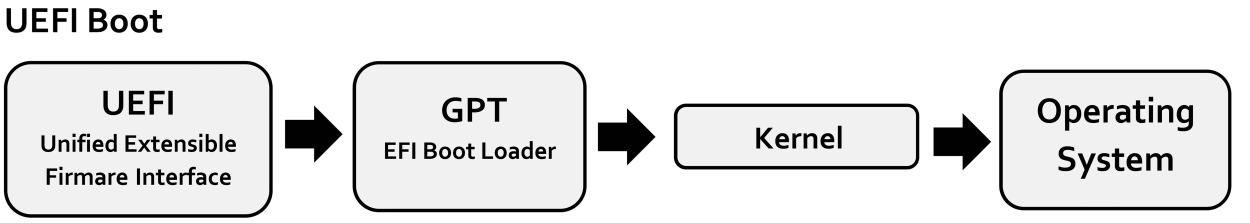
Ранняя загрузка

- Код ранней загрузки выполняет все более важную роль и становится более объемным
  - Выступает промежуточным звеном в цепочке доверия между прошивкой и ядром
  - Объем выполняемой им работы возрастает
  - Требования к данному кода повышаются



Ранняя загрузка

- Большая часть внимания разработчиков проходит мимо кода ранней загрузки
- Защита памяти на этапе ранней загрузки ядра **Linux** *практически отсутствует*!
- Также, код загрузки ядра недостаточно соответствует спецификациям



Обзор существующей защиты в Linux

В ядре Linux на ранних этапах уже применяются:

- Рандомизация раскладки памяти (aka **KASLR**)

- Шифрование памяти и регистров (**AMD SEV/SEV-ES** или **Intel TDX**)
  - Защита от атак со стороны гипервизова в *VM*
  - Только в новых серверных процессорах
  - Во время загрузки—только для **SEV**

- **UEFI Secure Boot**—протокол проверки цифровых подписей
  - Не проверяется подлинность **initramfs**—образа начальной файловой системы, содержащей драйвера и скрипты инициализации
  - Существующее сейчас решение—Unified Kernel Image

UEFI

- **UEFI** (Unified Extensible Firmware Interface) —современный стандарт интерфейса прошивки, пришедший на замену **BIOS** (Basic Input/Output System) Boot.

- Поддержка **UEFI** необходима для использования современных возможностей аппаратного обеспечения

- Несоответствие спецификации: предположение ядра о том что вся память исполняема:
  - Может делать невозможной или затруднять загрузку ядра на более безопасных реализациях прошивок

Secure Boot

- **UEFI Secure Boot**—протокол проверки цифровых подписей

- Для его применения существует необходимость подписывать ядро сертификатом Microsoft
  - *Технически не обязательно, но затрудняет установку*

- Для этого оно должно отвечать требованиям
  - *4k выравнивание, соответствие спецификации PE, W^X, и т.д.*
  - Недавно требования стали более строгими
  - `https://techcommunity.microsoft.com/t5/hardware-dev-center/updated-uefi-signing-requirements/ba-p/1062916`

Write-xor-Execute

- Политика `W^X`—хороший способ сократить поверхность атаки
  - `W^X`—отсутствие одновременно исполняемой и записываемой памяти.
  - Применяется в других ОС, таких как **OpenBSD** на всех этапах исполнения
  - В ядре **Linux**—только при полностью инициализированном ядре
  - Является одним из требования **MS**
  - Затрудняет эксплуатацию уязвимостей
  - Помогает выявлять ошибки доступа к памяти

Цель

- Улучшить защищенность и корректность кода ранней загрузки ядра **Linux** для платформы **x86_64** при использовании **UEFI**-совместимых прошивок

- Задачи:
  - реализовать поддержку политики `W^X` и общую защиту памяти на этапе ранней загрузки;
  - сохранить существующую функциональность;
  - не вносить существенных накладных расходов;
  - улучшить соответствие спецификациям **UEFI** и **PE**.

Реализованные улучшения

Для всех путей исполнения можно улучшить защиту памяти:

- Убрать неявное выделение памяти из обработчика `#PF`
- Разделить код/данные трамплина
- Выровнять секции кода и данных по границе страницы в ldscript для распакованного и сжатого ядра
- Указывать требуемые права доступа к памяти при отображении
- `W^X` не везде применима для них:
  - Код перемещается
  - Распаковка ядра производится на месте
  - В начале исполнения таблица страниц есть, но мы ей не владеем

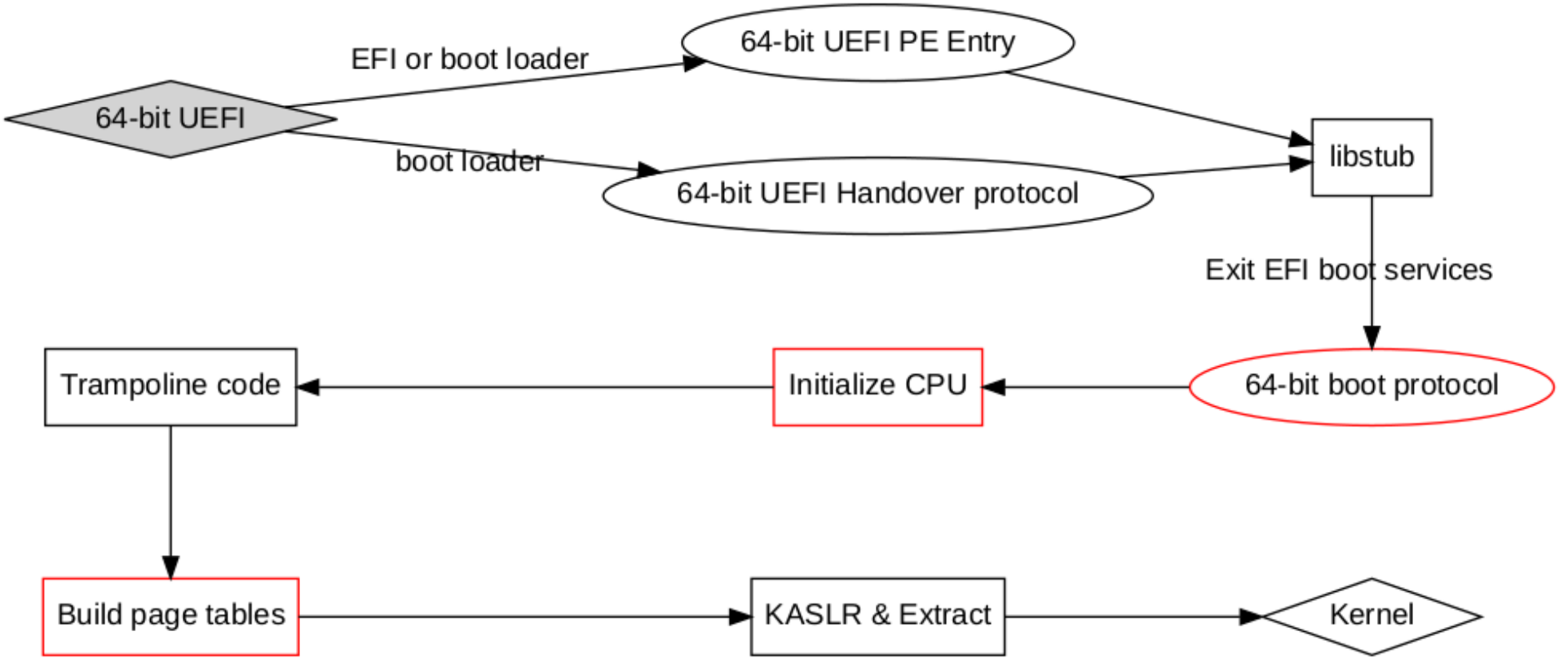
Реализованные улучшения

Для UEFI можно пойти дальше:

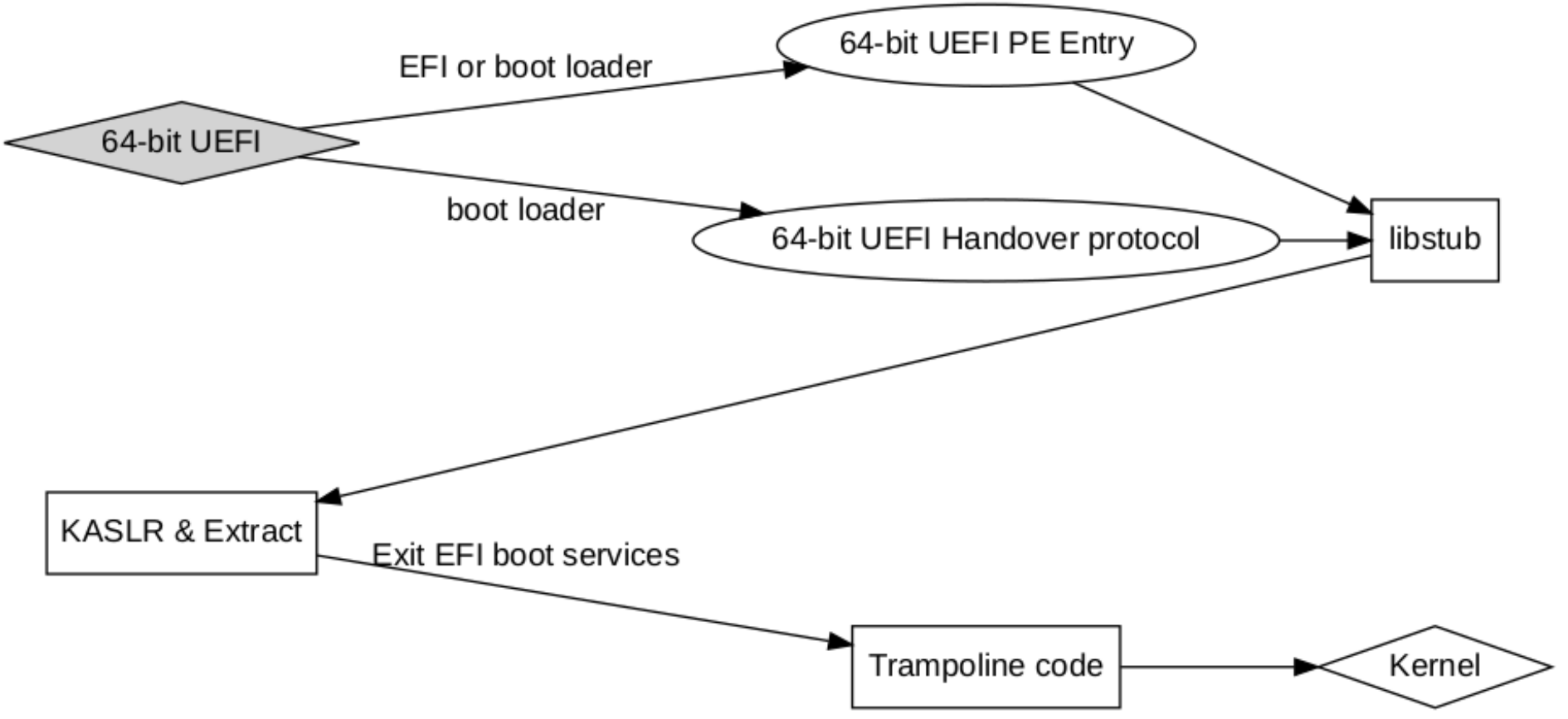
- Производить распаковку ядра из **libstub**
  - Позволяет полностью реализовать `W^X`
  - Позволяет избавиться от legacy кода

- Установить минимально необходимые атрибуты памяти для каждого региона памяти для секций PE и при выделении памяти из UEFI

Загрузка через UEFI



Загрузка через UEFI



Улучшения соответствия спецификации PE

- Выровнять структуры PE на натуральное выравнивание
- Соблюдать минимальные выравнивания секций
  - 512 в файле и 4096 в памяти
- Корректно устанавливать флаги секций
- Корректно устанавливать значения полей.
  - SizeOfCode, SizeOfData, BaseRelocationTable, и т.д.
- Помечать файл как NX-совместимый

Результаты

- Реализованы 2 набора патчей ($\approx +2800, -1000$ строк):

- Набор патчей улучшения совместимости с **UEFI**
  - Убирает предположение ядра об исполняемости произвольных регионов памяти
  - Реализует поддержку более безопасных прошивок **UEFI**
  - Позволяет прошивкам устанавливать строгие атрибуты памяти
  - Уже включен в основную ветку кода

Результаты

- Основной набор патчей (x86_64, частично i386)
  - Добавляет поддержку `W^X` для таблиц страниц, которым владеет ядро
  - Полностью реализует `W^X` для загрузки с UEFI
  - Убирает неявное выделение памяти
  - Убирает перестроение таблиц страниц и упрощает код инициализации для UEFI
  - Улучшает соответствие PE-файла ядра спецификации и требованиям Microsoft
  - Убирает излишние копирования образа ядра
  - Получил достаточно положительные отзывы

- `https://lore.kernel.org/all/cover.1678785672.git.baskov@ispras.ru/`

Результаты

- В результате обсуждений мейнтейнерами была предложена альтернативная реализация части патчей, исключающая больше излишнего кода
  - Только для части, связанной с распаковкой ядра напрямую из EFISTUB

- Остальные части серии планируется разделить на более атомарные и предложить для последующих версии ядра
  - Особенно патчи, связанные с форматом PE-файла ядра

- `https://lore.kernel.org/all/20230607072342.4054036-1-ardb@kernel.org/`

Заключение

- Была улучшена защита на ранних этапах загрузки за счет реализации `W^X` и других улучшений, а также соответствие различным спецификациям и требованиям

- Дальнейшие шаги:
  - Реализовать лучшую поддержку Secure Boot—проверять **initramfs**
  - Предложить аналогичные патчи для других аппаратных платформ
  - Убедиться в поддержке `W^X` в уже распакованном ядре на дальнейших этапах загрузки

- **Вопросы?**