Конструирование ядра операционных систем (I+)

Безопасность платформы: инициализация

План

- Введение в цепочку доверия UEFI
- Угрозы для прошивки и операционной системы
- Сценарии реализации угроз с примерами
- Механизмы защиты в прошивке и операционной системе
- Практическая часть лабораторной работы

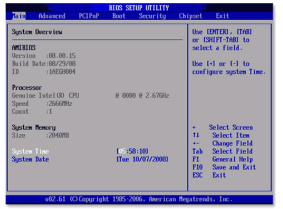
Свойства безопасности

1. Секретность (Secrecy): информация недоступна незаконному пользователю.
2. Целостность (Integrity): информация не изменяется, не уничтожается, т.е. в любой момент времени t+1 информация такая же, как и в момент времени t.
3. Доступность (Availablity): возможность беспрепятственного доступа к информации для проведения санкционированных операций по ознакомлению с ней, документированию, модификации и уничтожению.

Свойства безопасности

4. Подлинность (Authenticity): отправитель и, в свою очередь, получатель уверены, что сообщение дойдет до адресата (при этом 1-е и 2-е свойства безопасности могут не соблюдаться).
5. Неотказуемость (Non-repudiation): при выполнении действий впоследствии отсутствует возможность отказаться от каких-либо действий.
6. Анонимность (Anonymity): невозможность идентифицировать человека.
7. Неотслеживаемость (Untraceability): нельзя отследить действия анонима.

Эра BIOS (1975—2010)



- Механизмы 2-факторной аутентификации — редкость
- Полнодисковое шифрование в отдельных продуктах
- В операционной системе может не быть MAC/DAC
- Отсутствие стандартного механизма проверки ОС
- Полное доверие ко всему подключаемому оборудованию
- Отсутствие аппаратных КГСЧ, криптоускорителей, TPM
- Отсутствие встроенного root of trust (сугубо внешние МДЗ)

Эра UEFI (2006—н.вр.)

- Поддерживается операционными системами более 10 лет
- Внедряется во все уровни ОС от загрузчика до работы ядра
- Является кросс-платформенной (распространена на x86)
- В ОС есть сильная криптография, MAC/DAC, полнодисковое шифрование, W^X, SMAP, (K)ASLR, канарейки, etc.
- Содержит стандартизированный механизм проверки аутентичности ОС и драйверов оборудования
- Оборудование имеет встроенный root of trust, КГСЧ, ускорители хэширования и шифрования

Угрозы на стороне прошивки

- Утечка информации (пароль к носителю, настройкам)
- Исполнение кода
  - Песочница или иная ограниченная среда
  - Внешний драйвер / загрузчик ОС
  - S3 Boot Script (выход из сна) / PEI
  - System Management Mode
  - Внешняя среда для платформы (HV, ME, Boot Guard, microcode)
- Закрепление в прошивке (сохранение после сброса)
  - SEC/PEI буткит
  - DXE/SMM буткит (большая часть МДЗ)
  - Контроль внешней среды

Угрозы на стороне ОС

- Получение несанкционированного доступа к ОС
- Загрузка скомпрометированной операционной системы
- Загрузка другой операционной системы (не обязательно скомпрометированной, например, не предназначенной для данного компьютера)
- Загрузка уязвимой операционной системы
- Невозможность загрузки доверенной операционной системы после компрометации платформы

timeglider.com/timeline/5ca2daa6078caaf4

Firmware/SMU: AMD x86 SMU firmware analysis

NIC/Firmware/DMA: Project Maux Mk.II

Firmware/BIOS/BMC/VMX/VMM: A Myth or Reality – BIOS-based Hypervisor Threat

VMX/VMM: Poacher Turned Gatekeeper: Lessons Learned from Eight Years of Breaking Hypervisors

VMX/VMM: Breaking Out of VirtualBox through 3D Acceleration

BIOS/Firmware/SecureBoot: All Your Boot Are Belong To Us

AMT/ME/Firmware/BIOS: Intel ME Secrets

BIOS/Firmware/SecureBoot: A Tale of one Software Bypass of Windows 8 Secure Boot

Feb 17, 2014

FDE/TPM/BIOS/Firmware: Evil Maid Just Got Angrier: Why Full-Disk Encryption With TPM is Insecure on Many Systems

BIOS/Firmware:Hardware Backdooring is Practical

Apple/UEFI/BIOS/OROM/Bootkit/Firmware: DE MYSTERIIS DOM JOBSIVS Mac EFI Rootkits

VMX/VMM: Virtunoid: A KVM Guest -> Host privilege escalation exploit

SMM/Firmware/KBC/EC: Sticky Fingers & KBC Custom Shop

VMX/VMM: Cloudburst: Hacking 3D (and Breaking Out of VMWare)

Sidechannel/SGX: CopyCat: Controlled Instruction-Level Attacks on Enclaves for Maximal Key Extraction

NIC/Firmware: Network Interface Firmware Backdoor with Tiagon 2

Firmware/AMD/PSP/fTPM: amdflaws.com

Firmware/OROM: Implementing and Detecting a PCI Rookit

BIOS/Firmware/OROM/UEFI/Apple: Thunderstrike 2: Sith Strike

Bootkit: VBootKit: Compromising Windows Vista Security

BIOS/UEFI/Firmware/SMM: How Many Million BIOSes Would you Like to Infect?

VMX/VMM: BluePilling the Xen Hypervisor

Firmware/BIOS/UEFI/SMM/ACPI/S3: Attacks on UEFI security, inspired by Darth Venamis's misery and Speed Racer

VMX/VMM: Subverting the Xen Hypervisor

ACPI/BIOS: Implementing and Detecting an ACPI BIOS Rootkit

BIOS/UEFI/Firmware: Analyzing UEFI BIOSes from Attacker and Defender Viewpoints

BIOS/UEFI/Firmware/SecureBoot: Extreme Privilege Escalation on UEFI/Win8 Systems

BIOS/UEFI/Firmware/SMX/TXT: SENTER Sandman: Using Intel TXT to Attack BIOSes

BIOS/Firmware/SecureBoot/Bootkit: Setup for Failure: Defeating UEFI SecureBoot

FDE: Evil Maid Attack

VMX/VMM: IsGameOver() Anyone?

BIOS/Firmware/SMM/SMX/TXT: Copernicus 2: SENTER the Dragon

BIOS/Firmware: Attacking Intel BIOS

Firmware/UEFI/S3/Apple: Wikileaks Vault 7 Dark Matter (CIA Apple EFI implant)

VMX: (Orig BluePill) Subverting the Vista Kernel for Fun and Profit

Firmware/BMC: The Unbearable Lightness of BMC

SMM: Using CPU System Management Mode to Circumvent Operating System Security Functions

Bootkit: eEye BootRoot

BIOS/SMM/Firmware: Defeating Signed BIOS Enforcement

DMA: Owned by an iPod: Hacking by FireWire

BIOS/SMM/Firmware/TPM: BIOS Chronomancy: Fixing the Core Root of Trust for Measurement

Программные сценарии атак

- Буткиты (HT rkloader, ThinkPWN, S3BootScript, Aptiocalypsis)
- Тулкиты моддинга прошивок (SmmBackdoor, PeiBackdoor)
- Множественные дыры в процессе обновления прошивок
- Возможность отката на уязвимую версию прошивки
- Уязвимости в загрузчике или ядре операционной системы
- Человеческий фактор и социальная инженерия

REF: www.blackhat.com/docs/asia-17/materials/asia-17-Matrosov-The-UEFI-Firmware-Rootkits-Myths-And-Reality.pdf

Программно-аппаратные сценарии

- Устройства с уязвимым или вредоносным Option ROM (ho.ax/De_Mysteriis_Dom_Jobsivs_Black_Hat_Paper.pdf)
- Устройства, нацеленные на баги в драйверах
- Скомпрометированные устройства вне x86 (e.g. ME: habr.com/p/430132)
- Устройства, атакующие в обход программного стека (e.g. OPAL: ieee-security.org/TC/SP2019/papers/310.pdf)

Как защититься?

- Атакуют всегда самое слабое звено
- Любую защиту можно сломать
- Стоимость атаки сопоставима с выгодой от её проведения
- Меньшую область для атаки легче защищать
- Сложные решения не работают, а комплекс простых мер на каждом уровне значительно повышает стоимость атаки
- Раскрытие механизмов и криптографических примитивов не должно влиять на защищённость
- *Качественного результата можно достичь только при грамотной организации процессов (см. SDL)*

Безопаснось платформы в разрезе

- Корень доверия платформы (root of trust)
- Разделение фаз загрузки и их верификация
- Верификация и изоляция загрузчика ОС
- Верификация кода начальной загрузки
- Верификация кода в пользовательском пространстве

There your BIOS
and hard drive is fried

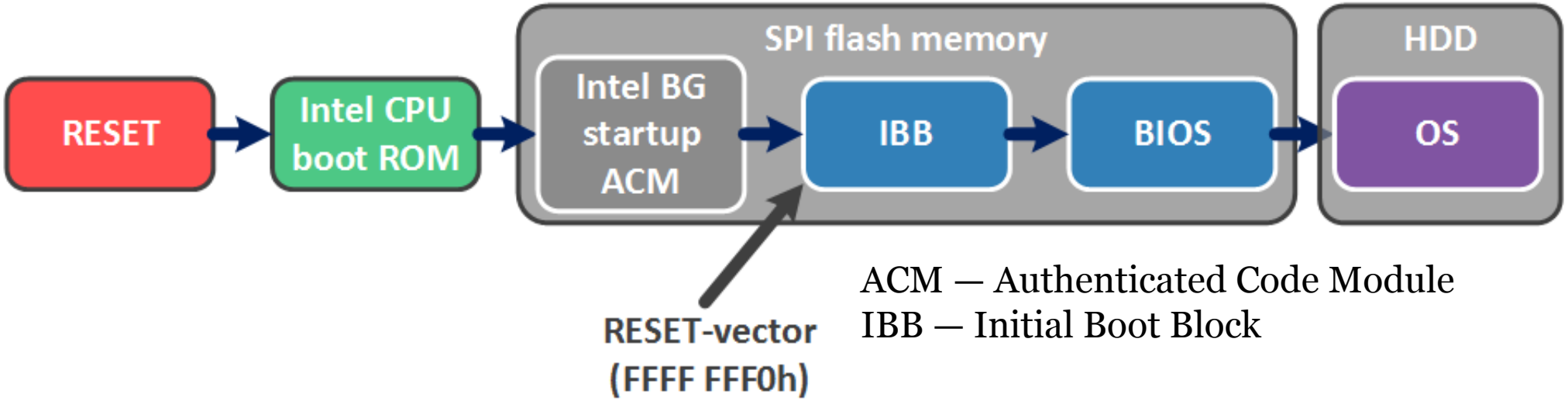
pls click my ad
I need to make money

Корень доверия (I)

Сущность, аутентичность которой принимается априори, выполняющая аутентификацию последующих элементов инициализации платформы.

- Apple T2 Security Chip (support.apple.com/HT208862)
  checkm8 для T2: github.com/h0m3us3r/ipwndfu
- Intel Boot Guard
- lowRISC OpenTitan (opentitan.org)
- Synopsys DesignWare Secure Boot (synopsys.com/designware-ip/security-ip.html)
- Read-only flash / ROM Mask / Гипервизор

Intel Boot Guard (I)

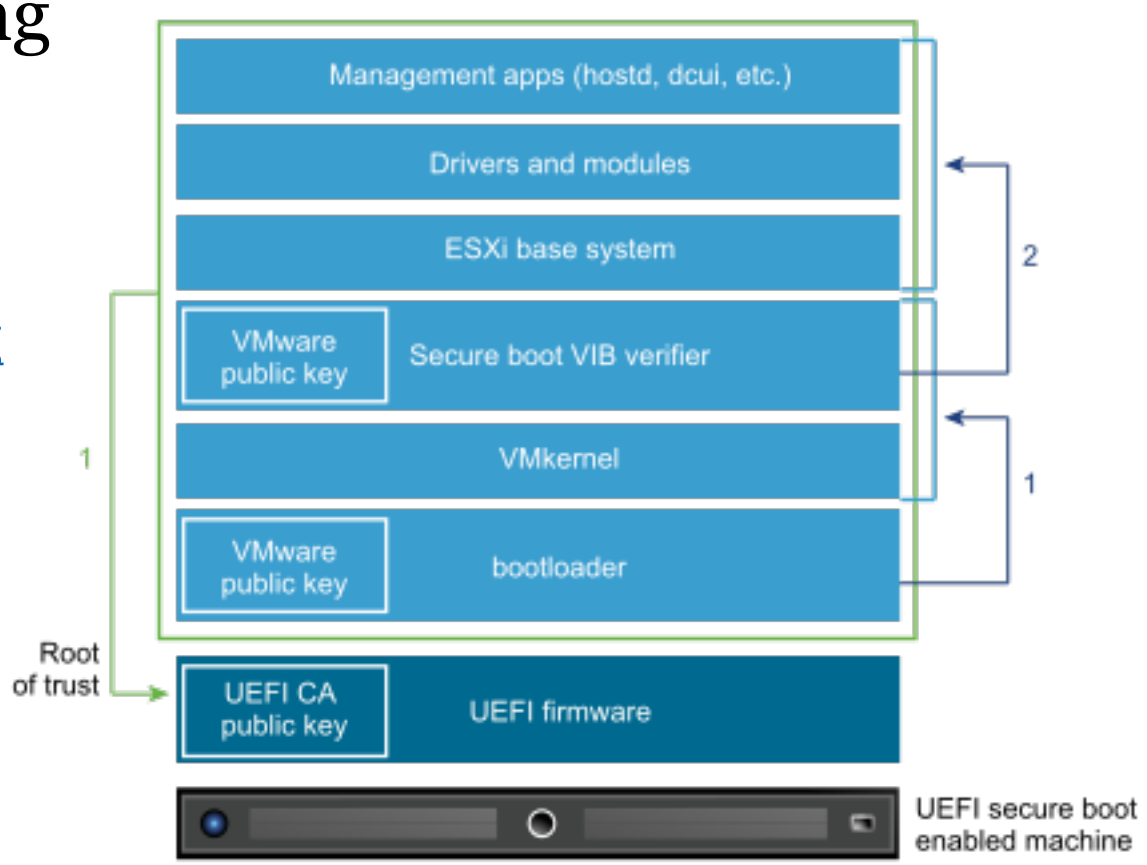


Доступен, начиная с Intel Haswell (на ноутбуках):

- habr.com/p/326556
- edk2-docs.gitbook.io/understanding-the-uefi-secure-boot-chain/secure_boot_chain_in_uefi/intel_boot_guard
- github.com/tianocore/tianocore.github.io/wiki/Boot-Guard-TOCTOU-Vulnerability-Mitigation

Безопасная загрузка (II)

- UEFI Secure Boot (uefi.org/specifications)
  32. Secure Boot and Driver Signing
  github.com/tianocore/edk2
- Apple Secure Boot
  (support.apple.com/HT208330)
  github.com/acidanthera/OpenCorePkg



Private Keys

Allows modification of PK and KEK

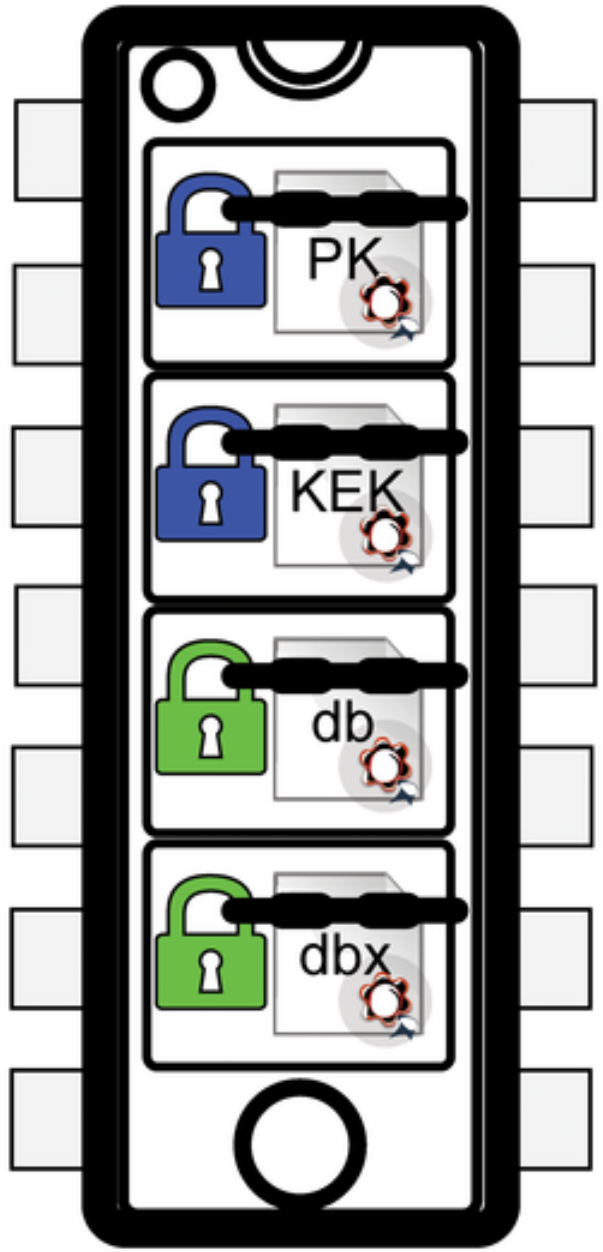
Allows modification of db and dbx

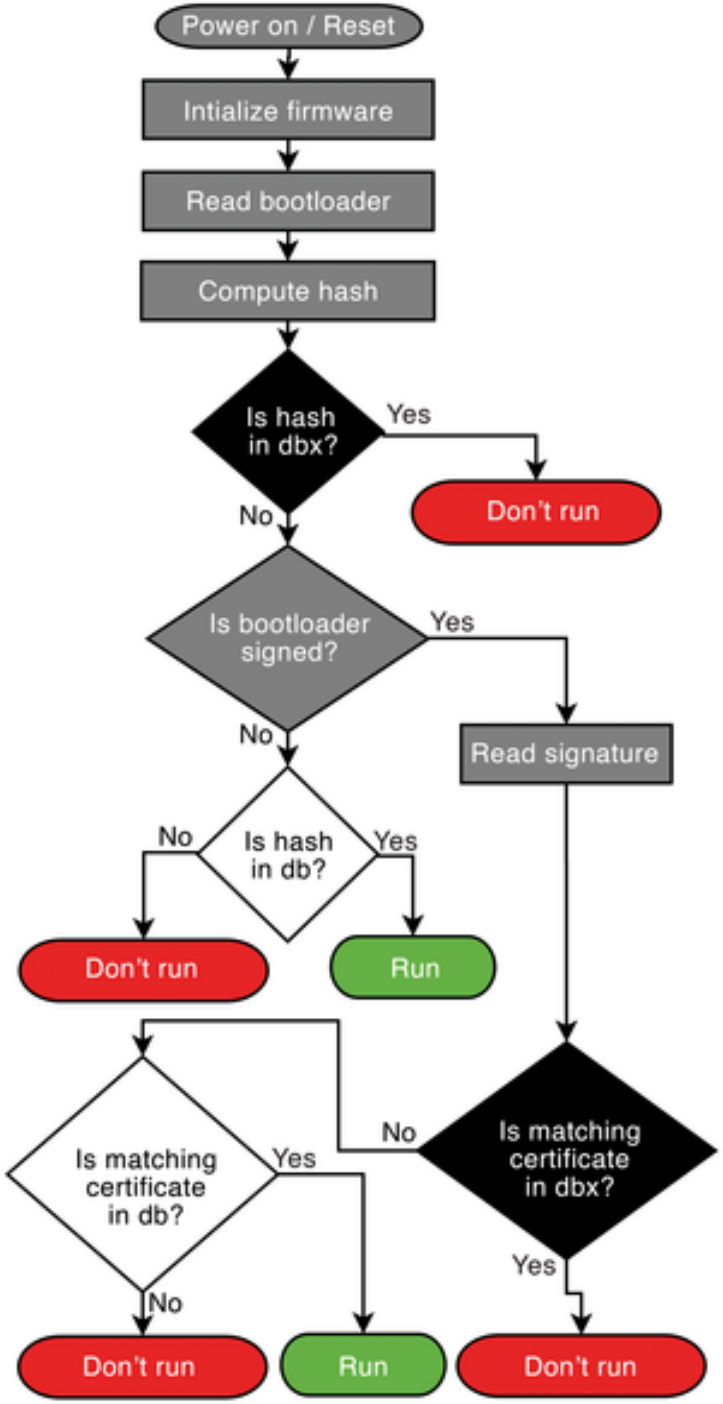
Allows verification of bootloaders

Prevents execution of bootloaders

UEFI Firmware

with certificate store





Что не так с UEFI Secure Boot? (II)

- Механизм проверки подписи, требующий парсинга PE
- Нет стандартной защиты от даунгрейда прошивки и ОС (нестандартные часто не работают или обходятся)
- Нет стандартного способа восстановления после взлома
- Отсутствуют механизмы персонализации ОС
- Отсутствуют политики безопасности (откуда загружена ОС, в каком режиме она запускается, кто инициатор запуска...)
- Все Linux системы и OPROM'ы (GPU, NIC) имеют один CA
- Производитель перестаёт обновлять прошивки

Работа с устройствами (IV)

1. Устройства могут иметь полный доступ к оперативной памяти через DMA (github.com/Cr4sh/s6_pcie_microblaze)
2. Устройства поставляются с драйверами (OPROM), которые исполняются в прошивке и подписаны одним CA

- Физический уровень защиты (имело смысл до Thunderbolt)
- IOMMU (Intel VT-d) — ограничение доступа к памяти
- Не исполнять внешние драйвера устройств вообще.
- Создать ограниченную песочницу для устройств. i.blackhat.com/USA-19/Thursday/us-19-Krstic-Behind-The-Scenes-Of-IOS-And-Mas-Security.pdf

Обновление прошивки (III)

1. Инициация обновления аутентифицируется
2. Проверка цифровой подписи в доверенном хранилище
3. Обновление прошивки несовместимо с запуском ОС
4. Прошивка недоступна для записи в ином режиме (SMM)
5. После обновления установка старой версии невозможна
6. Откат к старой версии невозможен даже при физ. доступе
7. Обновление персонализировано со сроком действия

Запуск операционной системы (V)

1. Загрузчик должен проверять любой файл (включая ядро) (eclypsium.com/2020/07/29/theres-a-hole-in-the-boot)
   Пример: vaulting в OpenCorePkg.
2. Загрузчик должен защищать память (свою и ядра)
   Примеры: W^X, MAT, VBS (docs.microsoft.com/windows-hardware/design/device-experiences/oem-vbs)
3. Конфигурация должна зависеть от платформы
   Пример: TPM Unseal с Intel TXT Measured Boot (docs.microsoft.com/en-us/windows/security/information-protection/tpm/tpm-fundamentals)

Запуск операционной системы (V)

4. Не следует доверять родительской платформе. То, что частично доступно пользователю (e.g. CMOS, NVRAM, Storage) или имеет сложную спецификацию (e.g. ACPI, SMBIOS) может не соответствовать ожиданиям.
5. Используйте рандомизацию адресного пространства (wikipedia.org/wiki/Address_space_layout_randomization)
6. Удаляйте ключ шифрования носителя из памяти при сне (S3) и шифруйте образ RAM при гибернации (S4).
7. Загрузчик ОС имеет меньше доверия, чем прошивка, ошибки запуска ОС должны перезапускать платформу.

Технические меры защиты (VI)

1. Стековые канарейки со случайным SEED
2. Аутентифицированные указатели (если есть)
3. Automatic variable initialization (-ftrivial-auto-var-init)
4. Безопасное стирание памяти (вместо memset)
5. Ретполины (-mretpoline) и SMAP/SMEP, если есть песочница (llvm.org/docs/SpeculativeLoadHardening.html)

Спасибо за внимание!

Вопросы?