Безопасность микроядра. Нужны ли бинарные митигации, если следуешь принципу secure-by-design?

Мелехова А.Л., АО «Лаборатория Касперского», anna.melekhova@kaspersky.com

Abstract

Когда речь заходит о системах, разработанных по принципами secure-by-design, то разработчики задумываются о количестве и качестве митигаций от бинарных уязвимостей. С одной стороны эти митигации - ответ на хорошо изученные угрозы, а с другой стороны затраты на secure-by-design в сумме с тратами на обычные митигации дают астрономические усилия на безопасность. Сегодня поговорим, как видоизменяются бинарные харденинги в безопасной микроядерной системе KasperskyOS, которая строилась исходя из принципов secure-by-design. Рассмотрим KASLR, W^X, KPTI, CFG, stack canary. Доклад ориентирован на техническую аудиторию, однако поясним все аббревиатуры. Содержит следы ассемблера.

Тезисы

(1) **Микроядро, или почему Linux-у так сложно стать безопасным?**

- Поверхность атаки у микроядра и у гибрида/монолита
- Драйвера, пару примеров Linux, когда “вскрывали” через драйвера. Уязвимости в Mali GPU драйвере
- Секреты в ядре
- Карта харденингов от Попова, проект PAX
- SELinux и Android binder для надстройки reference monitor

(2) **Харденинги (== митигаций от бинарных уязвимостей) ядра**

Харденинги будем рассматривать по шаблону - от какой атаки защищаемся, как защищается Linux, как защищается Kaspersky OS. В тезисах лишь наметки.

(2.1) **Stack canary / SSP = shadow stack pointer**

Стековая канарейка митигирует переполнение буфера на стеке. Митигация нужна и для KasperskyOS, но в ограниченном размере. Взаимодействие user-kernel space ограничены, типизированы и проверяются в модуле безопасности.
Подчеркнем, что имплементаций канареек много. Per-thread, per-system call, per-system, per-task. Чуть намекнём как выбирать.

(2.2) KASLR (kernel address space layout randomization)

KASLR для микроядра нужен в меньшей степени, чем для монолита (отсылка на статью Таненбаумовских студентов). И вероятность утечки адреса меньше, ведь нет сторонних драйверов.

(2.3) CFG/CFI (control flow integrity/guard)

CFG, как и канарейка, митигирует построение JOP/COP/ROP chains. Но при включении KASLR уже сложнее строить цепочку. Присоединяем к этому отсутствие в микроядре драйверов и сторонних библиотек, маленький размер ядра - и получаем чисто технически сложности с построением цепочки. И типизация параметров усложняет атаку. Таким

образом, CFG/CFI рекомендуем включить при аппаратной поддержке и отсутствии оверхеда. Потому что эксплуатации, от которых она защищает трудно достижимы.

(2.4) KPTI (kernel page table integrity)

KPTI защищает от спекулятивных атак на чтение секретов из ядра. Но эта митигация вносит ощутимый оверхед, особенно при частных системных вызовах. Вспоминаем, что в ядре секретов мало, только сообщения на время передачи и если они не шифрованные. При этом вычитка даже одного байта в meltdown/spectre занимает время, а сообщения быстро передаются туда-сюда. И плюс секреты еще и найти нужно (KASLR). Поэтому строгой необходимости делать KPTI нет.

(3) Пример бинарных митигаций недоступных монолитам/гибридам.

Полноценный W^X. В целом, если процессу позволяется иметь страницы writable+executable, то это идеальная цель для создания payload-а. Так давайте запретим процессам получать такие страницы. Ок, но в Unix DAC оригинальной модели такого per-process разделения не предусмотрено. Ну а ядро? Запрещено ли ему выделять W+X? Нет, не запрещено. Потому что bpf. Да, ebpf имеет верификатор и процессы, создающие bpf запросы, должны иметь соответствующие CAPABILITIES. Однако, факт остается фактом - ядро Linux выделяет W+X kernel pages.
В случае же микроядра с reference monitor-ом и deny-by-default конфигами на стадии разработки политик лишь ограниченному числу процессов даются права на аллокации W+X страниц (web assembly например). А ядро себе W+X страниц не выделяет.

Выводы

Харденинги в secure-by-design системах нужны в меньших объемах. Для каждой ОС нужно строить threat model и изучать возможные варианты проникновения, чтобы оценить при каких условиях до вашей митигации “дойдет очередь”. Однако, в secure-by-design систем возможны такие бинарные митигации, которые полноценным монолитам лишь мечтаются.