



上海交通大学

SHANGHAI JIAO TONG UNIVERSITY

# KASLR in the age of MicroVMs

EuroSys 2022

Benjamin Holmes, Jason Waterman, Dan Williams

2022 年 4 月 28 日



# 第一部分

## Outline

microVM 很关注启动速度，因此 microVM 通常省略 bootstrap 步骤，直接进入虚拟机 kernel 中启动，但是这使得 vm 缺少了 KASLR 的保护。

这篇文章分析了为什么不能够在 bootloader 过程中实现 KASLR，并且将 KASLR 的功能实现在 monitor 中，取得了能够接受的性能开销。

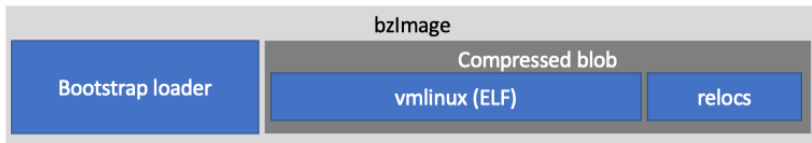
第二部分

# Background

microVMs: 常用于 serverless 中，运行时间短，必须能够快速启动。之所以是 vm 而不是 container 是因为 vm 的安全性更高。

# Booting a microVM

通过 bootloader 启动 bzImage



**Figure 2.** Components of Linux bzImage

# Booting a microVM

## kernel 直接启动



上海交通大学  
SHANGHAI JIAO TONG UNIVERSITY

Modern 的 VM monitors(Fire-cracker, Cloud Hypervisor, QEMU) 都支持跳过 bootstrap 直接启动一个未被压缩的 kernel, 而不是 bzImage。因此直接启动 vm 的 kernel 将不能够给 vm 实现 KASLR。

KASLR 是对在 kernel 启动的时候对整个 kernel 做一个基地址随机化, 来抵抗 ROP 攻击。

KASLR 的两个主要问题不再是问题:

- KASLR 在 kernel 运行过程中只进行一次 (启动时) 随机化: microVMs 本身运行时间很短
- KASLR 粒度太大, 一旦发生了内核地址泄漏, 整个内核的 KASLR 就相当于无用: kptr\_restrict, FGKASLR



# Bring back KASLR to microVM



KASLR 作为防止 OS 受到攻击的关键措施之一，并且自身的很多问题得到解决，所以作者强调 KASLR 还是有用武之地的。作者认为应该为 microVM 实现 KASLR。

第三部分

# Contribution

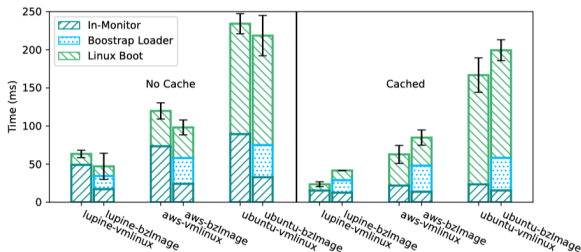
# Contribution 1: 指出 microVM 缺少 KASLR



microVM 缺少 KASLR 的保护，作者指出了这一点，并说这是他们的贡献之一。

## Contribution 2: 分析了为什么不能够在 bootstrap 中实现 KASLR

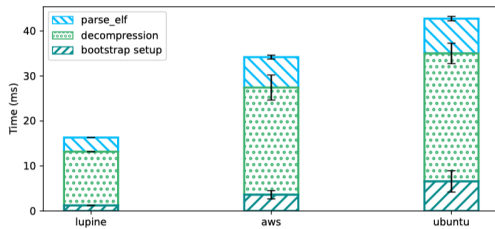
由于 bootstrap 启动的 bzImage 是压缩的，而 direct boot 的 kernel image 是未经过压缩的，作者首先研究 compress 对于 microVM kernel 启动时间的影响。



**Figure 4.** Boot times for compressed (with LZ4) and uncompressed kernels

## 实验 1

分析有无 cache 的情况下，bootloader boot 和 direct boot 哪个更快

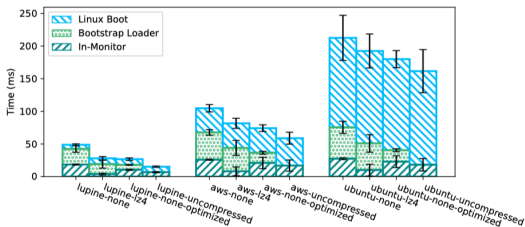


**Figure 5.** Breakdown in cost of in the Linux bootstrapping process

## 实验 2

分析 bootloader 中时间花在了哪些步骤

既然 bootloader 中的主要时间是花在了解压缩上，而 bootloader 又是现阶段 KASLR 基于的工具，那一个直接的想法就是 bootloader 不需要解压缩，直接 load 一个 uncompressed kernel，进行地址随机化就行。这样启动时间就会比 bzImage 好很多，事实上是否如此呢？



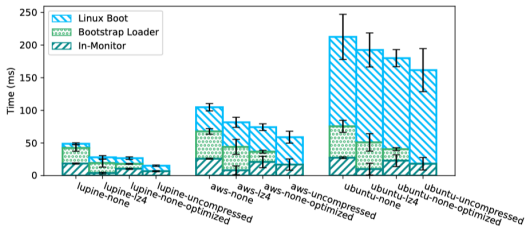
**Figure 6.** Bootstrap timings comparing LZ4 to compression "None".

### 实验 3

bootloader 加载未经过压缩的 kernel(不解压, 直接 copy)

性能甚至比加载压缩的 kernel 还要差, 作者修改了 bootloader 后, 跳过了两次多余的 image 拷贝后, 再次测试:





**Figure 6.** Bootstrap timings comparing LZ4 to compression "None".

## 实验 4

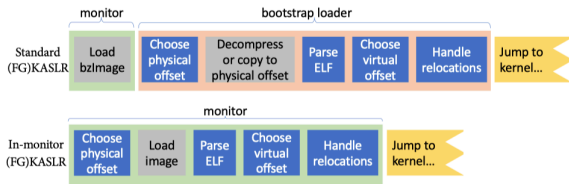
### 优化 bootloader 后的启动时间

none-optimized 依旧比 uncompressed 要慢，作者认为 bootloader 中实现 KASLR 是不可行的。于是引入了他们的设计 in-monitor KASLR。

## Contribution 3: 提出了在 monitor 中为 vm 实现 KASLR 的设计, 并在 Firecracker 中实现

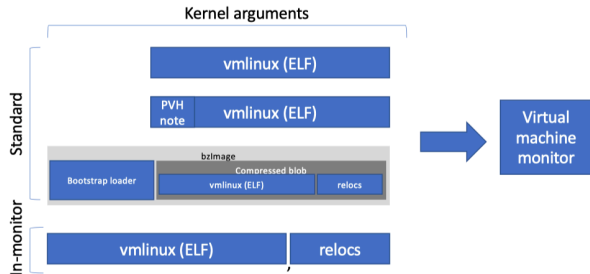


核心的设计是在 monitor 中完成原来在 bootloader 中完成的 parse ELF 和重定位这两个事情。然后直接跳转到 guest kernel 中运行。



**Figure 7.** In-monitor KASLR eliminates expensive/redundant bootstrapping.

相当于除了解压缩其他 bootloader 做的事情都在 monitor 中完成



**Figure 8.** Possible kernel input to microVMs. In-monitor KASLR takes an uncompressed kernel with relocation information.

为了完成 relocation 需要额外的 input 信息 (bzImage 里面的 relocs)

第四部分

# Implementation

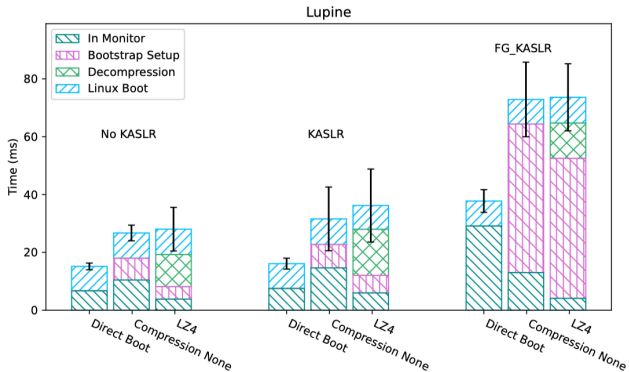
在 Fire-cracker v0.26 中实现了 in-monitor 的 KASLR 与 FGKASLR。  
KASLR 不超过 200 行, FGKASLR 不超过 1000 行。

简要介绍下实现上的一些点:

- 重定位后不立刻更新 `/proc/kallsyms`, ORC stack unwinder table(用于 stacktrace)。因为 micro-VMs 的应用程序其实不会访问它。
- `vmlinux.relocs` 由 kernel build 程序直接生成, monitor 直接读就好
- 直接使用 Rust 语言提供的随机数
- hardcode 内核启动参数 (实现还不完备)

第五部分

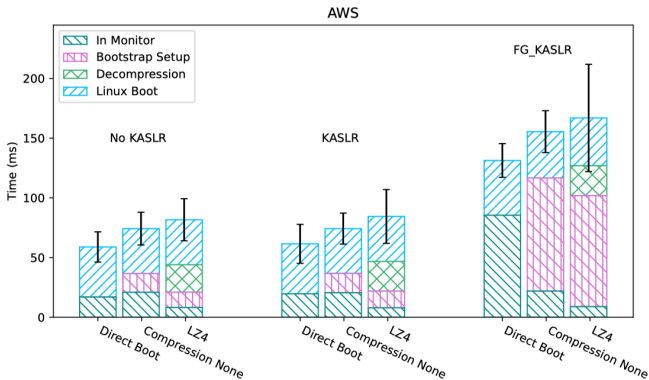
# Evaluation



(a) Lupine kernel config

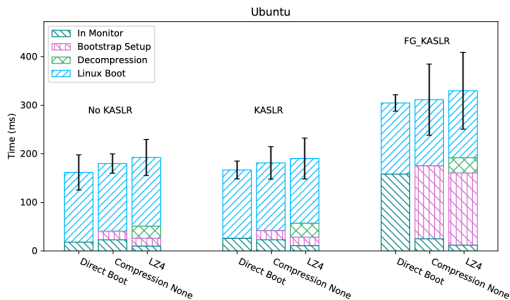
图: lupine kernel 代表 single-purpose 很小的 kernel config, 用于 unikernel-based environments





(b) AWS kernel config

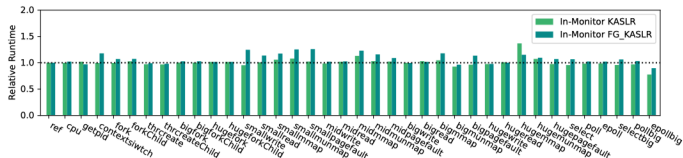
图: AWS kernel 指 FireCracker 上 vm 用的 kernel config, 代表 micro-vm 的配置



(c) Ubuntu kernel config

图: Ubuntu 是 Ubuntu 18.04 的 config, 代表 kernel 发行版的配置

in-monitor KASLR 相对于之前提到过的修改过的 bootloader 实现 KASLR 的方式比起来在 Luping, AWS 和 Ubuntu 分别有 15ms, 13ms, 16ms 的启动时间提升。相比于传统的 bzImage boot 则分别有 20ms, 23ms, 25ms 的启动时间提升。



**Figure 11.** Relative runtime on LEBench kernel microbenchmarks between the baseline *aws-nokaslr* and a KASLR-enabled kernel needed by in-monitor KASLR *aws-kaslr* shows low impact on overall system performance.

通过 in-monitor 中实现 KASLR 对整体性能的 overhead 很小，通过 LEBench microbenchmarks 测试得出 KASLR 仅有不到 1% 的运行时间影响，FGKASLR 会慢上大约 7%。

第六部分

# Discussion

当前的工作是建立在有 cache 的情况下未压缩的 kernel image 比 bzImage 启动更快的前提下，但是在 unikernel 的情况下，每个 app 对应一个 kernel，这时候没有足够的 cache 缓存 kernel image。这可能导致这种情况下重新在 bootloader 实现 KASLR 更佳

第七部分

# Conclusion

# 总结

- 为 microVM 在 monitor 中实现了 (FG)KASLR
- 在 monitor 中实现 KASLR 是一个非常直接的想法
- 在 monitor 中 KASLR 实现是一个比较工程的问题
- 论文也并没有表现出它们有哪些创新的设计，反而将很大的篇幅介绍背景，motivation 和测试步骤。



上海交通大学  
SHANGHAI JIAO TONG UNIVERSITY

谢谢

Benjamin Holmes, Jason Waterman, Dan Williams · KASLR in the age of  
MicroVMs