Motivation

解决什么问题？

设计并实现针对USB驱动的可扩展的模糊测试框架。

为什么解决？

USB(Universal Serial Bus)凭借其高带宽、即插即用和电源供给等特性，获得了广泛的应用并且USB驱动直接在内核或高优先级进程中运行，因此其安全性十分关键。

USB驱动在开发时，默认信任设备（即没有考虑到恶意设备的存在，但是现在很多可编程的USB设备导致恶意设备是常见的 BADUSB），并且USB驱动开发好后，也通常没有经过充分安全测试。

WirelessUSB、USPIP和USBRedir等技术，还使得USB设备可以实现远程连接和访问，进一步增大了USB驱动的攻击面。

现存的保护USB驱动免受恶意设备攻击的防御机制，效果有限，部署复杂，可扩展性差。例如，基于包过滤的防御机制（LBM）只能对已知的攻击进行有效防护，无法抵御未知攻击。

因此，保护USB驱动的最好方法就是，直接对USB驱动进行漏洞挖掘和修复。

为什么困难？

对USB设备驱动进行模糊测试，需要通过设备向驱动输入随机输入样例。

1. 利用特定的可编程设备，来辅助模糊测试（umap2）

缺点：昂贵、不可扩展、难以实现自动化模糊测试（不断插拔），无法获取测试覆盖信息

1. 修改内核注入数据，来辅助模糊测试

（syzkaller的扩展usb-fuzzer，利用software host controller将随机输入插入到USB驱动的栈中；PeriScope则通过修改MDA和MMIO接口来插入随机数据）

缺点：不可扩展；需要对操作系统有很深的理解；部分代码无法覆盖。

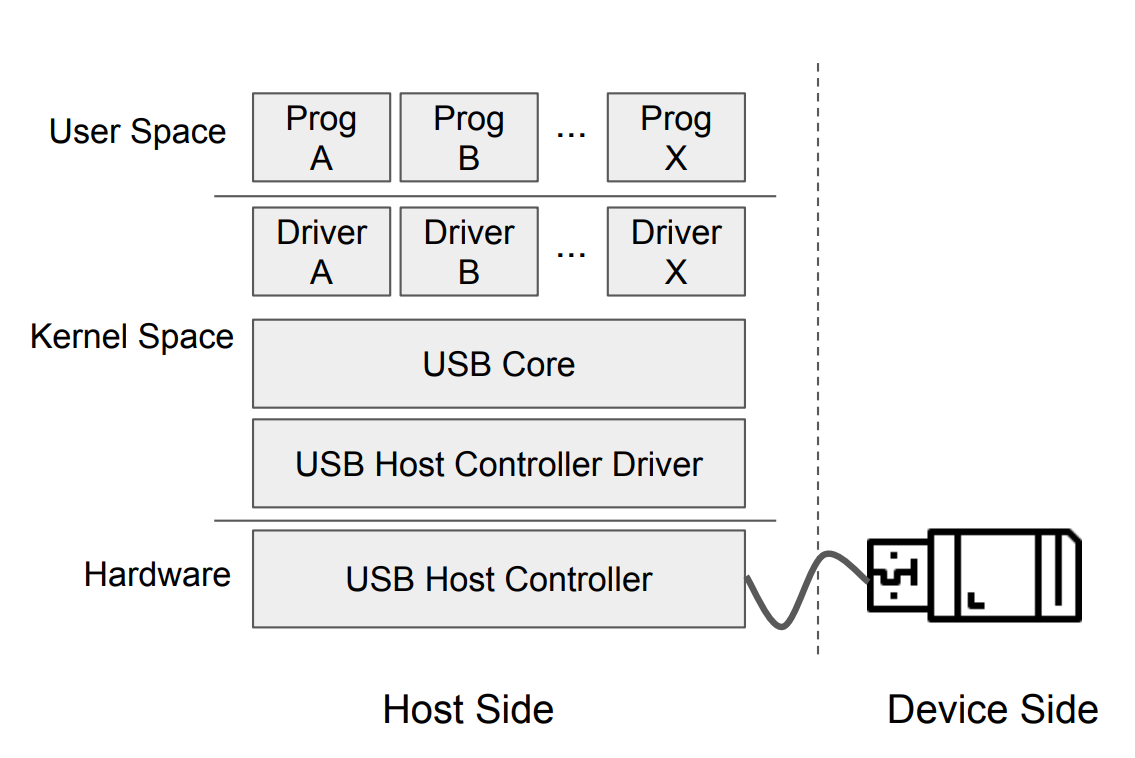
共同缺陷：

只能覆盖测试Probe routine，无法覆盖Function routine

Key Idea：

背景知识：

USB架构：



当USB设备插入主机时，USB Core会向设备发送请求，读取其device descriptors and configuration descriptors，从而决定选取那个具体的设备驱动与其交互。

USB攻击类型：

1. 系统和驱动对设备的默认信任的攻击：

BadUSB、USB Rubber Ducky、USB Driveby

1. 电子攻击：

利用USB的电源总线，对主机发送高电压造成物理破坏

1. 针对软件漏洞的攻击：  
   利用USB设备驱动的漏洞进行攻击

USB-MIDI device CVE-2016-2384 BF漏洞

三种攻击的根源在于，假设：硬件（设备）是难以被篡改并且是值得信任的。

最近的研究挑战了这一基本假设。设备固件的漏洞可能导致攻击者远程控制设备，WirelessUSB、USBIP等技术，使得USB接口可远程访问，可编程USB设备使得恶意设备的制作更加简单。

存在的疑问：

1. Fuzzing device在于驱动的交互中，是如何决定每次返回给驱动多少字节的数据的？（文章中只提到了顺序地返回Fuzzer提供的输入文件）
2. User Mode Agent，如果一轮模糊测试，没有触发漏洞和异常，它如何通过系统日志，判断一轮测试的结束？难道也是固定时间？
3. AFL或者修改的Kcov到底是怎么插桩计算覆盖率的？怎么就体现出对单进程的改进了？