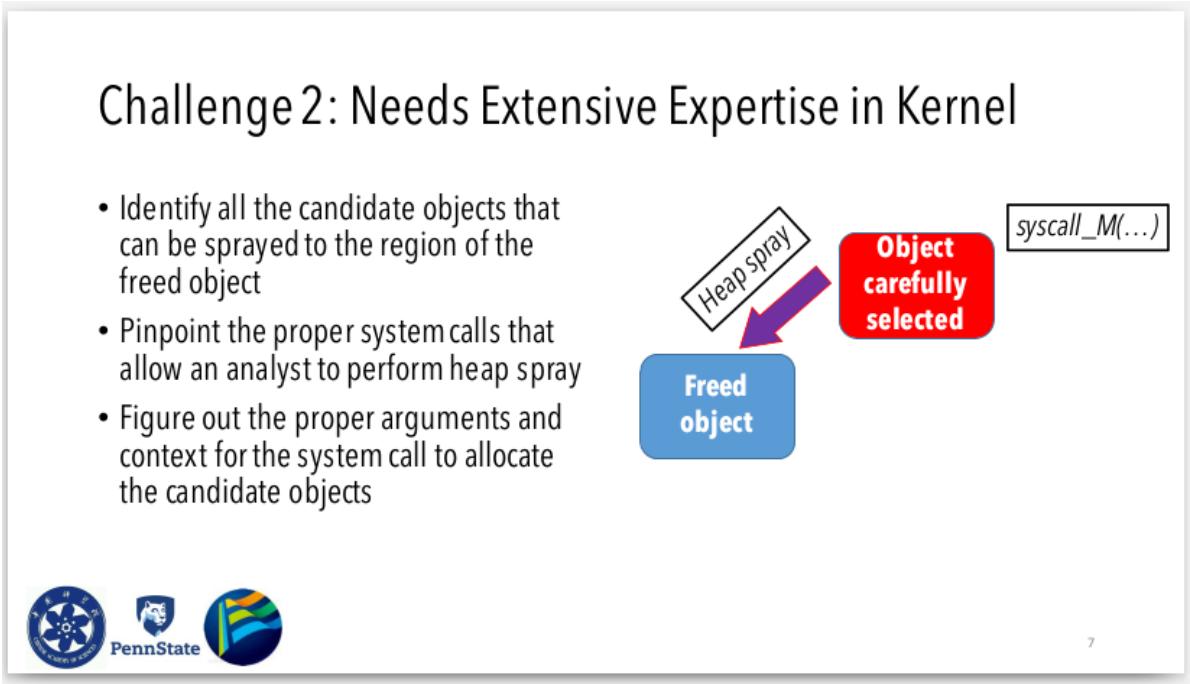
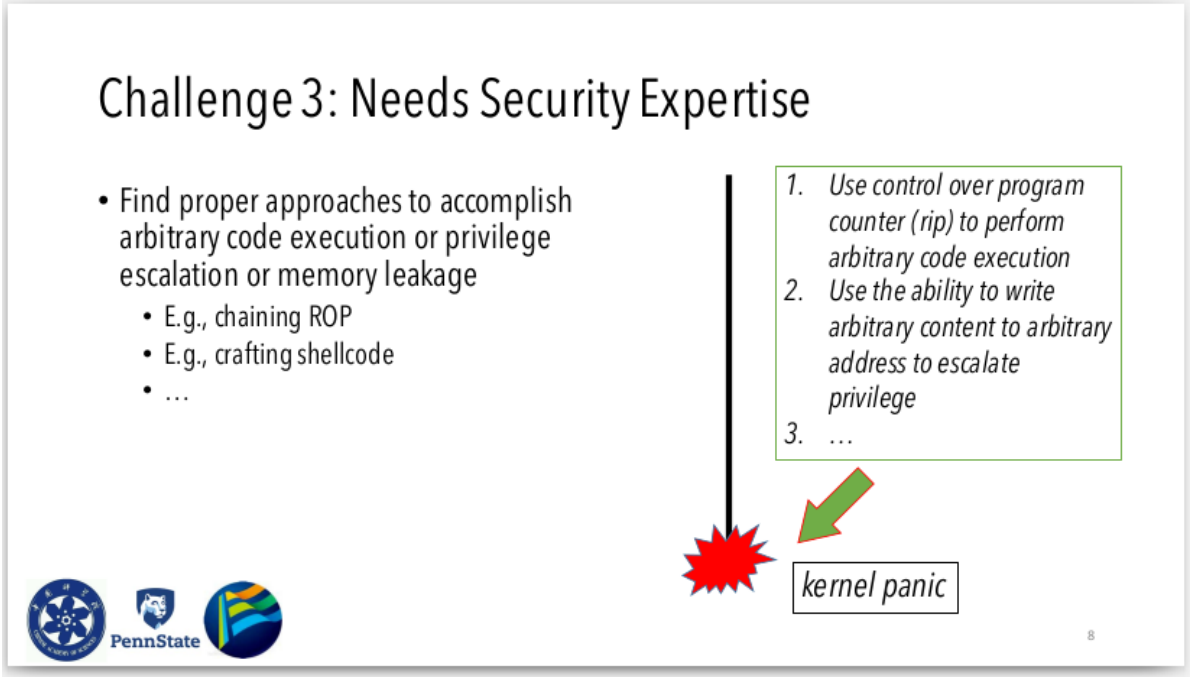
启发

吴炜师兄的这篇论文Wu, W., Chen, Y., Xing, X., & Zou, W. (2019). Kepler: Facilitating control-flow hijacking primitive evaluation for linux kernel vulnerabilities. *Proceedings of the 28th USENIX Security Symposium*, 1187–1204. 中提到了一些与我的思路相似的东西，我关注的主要是下图challenge2提到的问题





即面对一个UAF漏洞，如何找到合适的结构体去做堆喷射并完成利用。

吴炜师兄论文中采用的方法大致为：输入一个引发kernel panic的PoC，通过Kernel fuzz找到能够引发另一个kernel panic状态的系统调用序列，然后使用符号执行确定引发panic的指令来自悬空指针的读、写还是执行，从而评估fuzz找到的系统调用序列是否能够完成利用。

思考

最近看到的kernel pwn中UAF漏洞利用相关的题目都跟特殊的结构体和相关执行流有关，比较典型的就是corCTF 2021的fire of salvation和0CTF 2021 final中kernote。出题人的出题思路都是首先发现了一个内核结构体的利用方法，然后通过构建一个带有UAF漏洞的内核模块考察对这个结构体的利用，通常漏洞还有一些特殊的限制条件。

由此我想到，（至少在kernel pwn中）UAF漏洞可以抽象为由以下几种特性决定的一种模式：1.UAF控制的chunk的大小；2.能够读取chunk中若干字节；3.能够改写chunk中若干字节。从而，UAF漏洞利用的问题可以看作：给定一个UAF漏洞的模式，如何在内核中确定一条可以利用这个漏洞的系统调用序列（内核结构体和执行流）。

解决这个问题的方法似乎可以很大程度上借鉴吴炜师兄的这篇论文，用fuzz的方法找到能够做堆喷射的内核结构体，使用符号执行的方法判断响应的系统调用序列能否完成利用。

我希望设计这样一个系统：给定UAF漏洞的模式，给出能够利用这个漏洞的系统调用序列，同时给出使用该序列需要满足的特定限制条件（由漏洞程序提供）。

这个问题是可以说是吴炜师兄论文解决的问题的一个子集，但至少在以下两点上具有优势：1.输入是一个抽象的漏洞的模式，不需要从PoC程序中确定悬空指针产生和引发panic的悬空指针解引用指令的位置，一方面更加准确且容易实现，另一方面也增加了应用的范围；2.吴炜师兄论文中的方法，在fuzz过程中只寻找那些能够产生不同panic的系统调用序列，而没有关注UAF漏洞特性和系统调用互相作用的利用模式：如UAF漏洞具有如下两个性质：（1）能够控制一个cred struct；（2）提供任意字节写，显然可以通过fork()堆喷射cred结构体然后修改其中用户权限控制的字段完成利用，但在吴炜师兄论文中提出的方法无法识别这样的利用模式。

在实现上，吴炜师兄论文中为了提高fuzz的效率，只选择了部分与PoC漏洞产生强相关的系统调用做fuzz，这样必然会漏掉相当一部分可能能够完成利用的执行流。