OS 论文阅读（3）

刘炜
2014210878

June 15, 2015

1 CleanOS: Limiting Mobile Data Exposure with Idle Eviction [1]

1.1 思路

这篇文章关注的移动终端上的敏感信息本地存储的问题。作者通过调研分析 14 款知名应用，发现大多数都有本地存储的问题，有些存的是通信内容，有的存的是用户名口令。因此，作者希望系统能够相应的机制来保护敏感信息的安全，作者给出的解决方案是系统提供一套 API 帮助应用管理敏感信息，将这些信息 push 到一个安全的云端。作者为敏感信息设计相应的数据结构，系统通过污点分析来发现敏感信息，一旦发现污点打上了，就将信息发到云端。如果需要访问敏感信息，就将云端去数据解密后供应用使用。其中污点分析的部分是用 TaintDroid 的工作来实现的。

1.2 讨论

老实说，我不认为这篇文章的工作做得很好。理由有以下几条：

1. 从出发点来讨论，在本地存储用户名口令本来就是非常不安全的实践，而在本地存储其他的内容，比如聊天记录，甚至并不会被视为漏洞，即使如此，也可以通过对 sqlite 的数据库用 sqlcipher 进行加密或者对数据进行加密来进行保护。

2. 从提出的解决方案来看，首先这个云存储的安全需要论证。另外，应用是否愿意将自己的用户数据往这个云上放也是一个大问题。

3. 从提出的实现方法上来看，由于用到了污点追踪来分析敏感信息，那我们首先要看污点分析的效果。现在的污点分析只能说找出来的东西有一些信息量，但不能保证漏报的情况，用来做漏洞挖掘是有效的，但要求它保证一定的漏报率，这个是很难的，尤其是现在的污点分析只能针对 Dalvik 层的 Java 代码，对 native 层的代码大多没有进行处理。

1.3 体会

我觉得这个文章提到的一些问题是不是需要系统做出那么大的改动来完成这样的功能就很值得探讨。如果要实现类似的功能为什么不能系统在 framework 里实现一套 API 来做这件事，而要用污点来把这件事搞得这么复杂，这个问题本来就是应用上的问题，如果可以有轻量级的解决方案，没有必要上升到系统的层面。

2 DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis [2]

2.1 思路

DroidScope 这篇文章是针对 Android 系统实现了一个动态插装的系统，插装的层级从 Linux Kernel 到系统调用到 Dalvik 虚拟机都有，可以说是插装能做到的层级基本上都覆盖了，因为作者使用的插装方法是通过插装 Qemu 的 TCG 代码段。Qemu 的 TCG 代码段可以认为是与真机上的 Machine code 有相同层级的代码，因此这篇文章理论上能做到插装粒度已经做到了极致。

基于对 QemuTCG 代码段的插装可以为一些 Trace 的分析工具提供了必要的信息。需要指出的是，DroidScope 的工作不仅仅可以提供通常的插装工具提供的方法调用前和方法调用后的插装，还提供了类似寄存器读写前后和内存读写前后等的插装时机，这是由 DroidScope 的插装非常底层带来的好处。

DroidScope 的工作一个需要特别注意的问题是 JIT，因为在插装 Dalvik 指令的时候，依赖于 Dalvik 指令翻译成机器码这个过程的插装，JIT 可能会使得系统调用之前缓存的代码来执行，使得插装失败。一种比较暴力的方法是编译的时候完全禁掉 JIT，另一种方法也是文章中的方法就是选择性地禁 JIT，通过对 code regine 的判断来选择性地禁 JIT。

2.2 讨论

插装是很实用的分析方法，现在 Android 上可用的插装工具很多，比如有 adbi/ddi, Xposed, Substrate, Injectso, Frida 等等。这些工具与 DroidScope 相比，都是较为轻量级的工具，插装的层次没有 DroidScope 底层，将 DroidScope 和这些工具比较，可以发现 DroidScope 有以下的优点与缺陷. 优点:

1. 插装粒度细，不仅仅是 hookAPI
2. 插装比较底层，潜力无限，便于进一步拓展，做大规模分析

缺点:

1. 开发不如其他工具友好，对于小局部的插装不如其他工具方便
2. 模拟环境下运行，可能会被恶意软件检测
3. 与真机环境相比，运行效率和性能以及通用性有待进一步提高

2.3 体会

这个工作是非常不错的工作，但这篇文章还远没有把这部分的工作做完，比如污点分析 Tracer，作者的实现也是在 Dalvik 层上的，没有考虑到如何去把 control flow 层级的污点分析做起来，更加没有把 native 层的污点进行分析，但基于 qemu 的方法是有可能把这个难点问题解决的。

References

[1] Y. Tang et.al, "CleanOS: Limiting Mobile Data Exposure with Idle Eviction", In OSDI'2012

[2] L. Yan et.al, "DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis", In Usenix Security'2012