Checking Intent-based Communication in Android With Intent Space Analysis

APR 21ST, 2016

论文下载

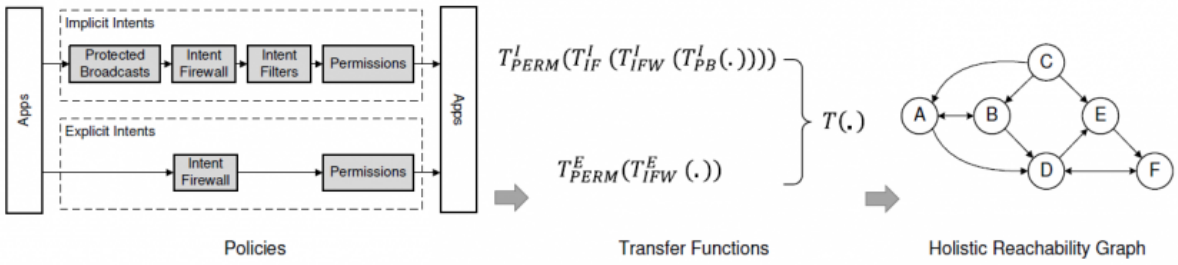
- Intent在Android中是APP之间主要的交互方式。处于安全考虑，需要限制部分APP之间的intent交互。这些限制模块可以称为security extensions。
- 系统内部的security extensions包括：Intent Filter,Intent Firewall(可在系统路径/data/system/ifw/*.xml下查看，会记录哪些APP发送的怎样的intent会被block),Permissions,Protected Broadcasts(某些系统广播只能由系统APP接受，发送等)。
- 除了系统内部的，目前也有很多额外的安全增强模块被提出来，比如：Saint,TISSA,CRePe,APEX,SE Android等等。
- 由于security extensions太多，各有各的规则，各有各的逻辑，就导致很难明确的知道到底系统中，每个APP之

间到底能不能用intent通信。同时，每个security extension，很难证明自己真正起到了效果。

- 因此，作者提出了**IntentScope**这个工具。这个工具会综合系统中所有的**security extensions**的规则，最后得出一个完整的,所有**APP**之间能否互相以**intent**通信的关系。

要说明APP之间复杂的限制规则，首先就需要一个统一的模型。作者提出了**intent space**的概念。每个intent有自己的属性，比如component name,action,scheme,authority,type,category等等。每个属性可看作一个向量，每个属性的值可以理解为向量的值。这样每个intent就有了自己的向量空间。默认的，每个APP所能发送的intent的空间都是无限大的。而security extensions会减小这些空间。

IntentScope通过针对性的分析每个security extension，将其限制规则转化为自定义的关于intent向量空间的运算，这样就可以用数学计算的方式来求出每个APP所能发送的intent的空间，最终可以生成APP之间的关系图。IntentScope的工作流程如下图:



应用这个工具，作者做了实际的部署及实验。选了两种机型和四个系统。作者通过实际生成的效果图对结果进行了分析，找了一些安全问题，但都是已知的问题。同时，作者给出了IntentScope的效率。但是并没有证明自己工具的准确率及误报率漏报率。

实验的几个点：

1. 没有任何的权的APP还是能和很多APP通信（其实就是组件暴露）。个人认为这个结果并不能说明什么问题。因为这些路径只是可能路径，APP实际上可能并没发送这些恶意的intent.
2. IntentScope能够检验某些隔离APP安全模块的有效性。
3. 这到系统中所有的Multi-app Workflows。Multi-app Workflow就是类似：下载图片-编辑图片-发送图片 这样的工作流程。由于IntentScope生成的图每个边和点都有对应的参数记录下来。比如这个intent的action是什么之类的参数。因此找到所有的Multi-app Workflows不困难。
4. 发现权限泄漏，也是老问题。