安卓应用中资源泄漏的自动修复

刘洁瑞, 吴添勇, 严俊, 张健

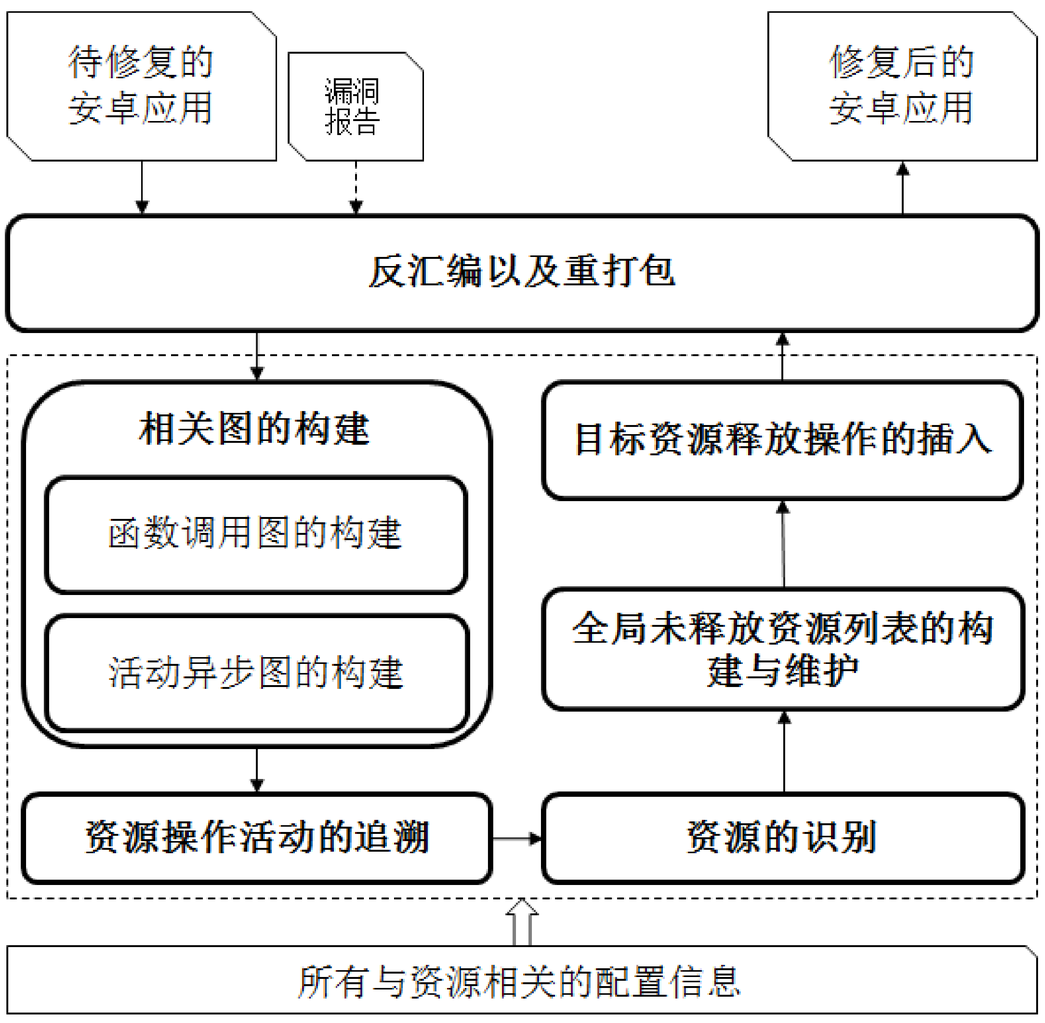
Fixing Resource Leaks in Android Apps with Light-weight Static Analysis and Low-overhead Instrumentation, ISSRE 2016

计算机科学国家重点实验室
{liujr, wuty, yanjun, zj}@ios.ac.cn

背景介绍

目前，安卓系统在移动市场上拥有越来越多的用户，成为越来越受欢迎的移动操作系统。但是，由于测试不充分、开发团队规模小、应用迭代周期短等各种各样的原因，安卓应用的质量依然令人们担忧。资源泄漏是普遍存在于安卓应用里的问题之一。资源泄漏会造成严重影响用户体验的问题，比如电池浪费、内存泄漏甚至程序崩溃。我们提出了一个通过程序分析和代码插桩技术来自动化修复安卓应用中的资源泄漏的方法。该方法保证了加入的补丁代码不会影响原有程序的运行逻辑，也不会引入新的问题和漏洞。

总体框架



基本步骤

相关图的构建：在分析和修复之前，该模块静态地构造函数调用图和活动异步图。

资源操作活动的追溯：对于可能导致资源泄漏的每个资源申请操作，修复过程需要将相应的资源释放操作插入到它资源操作活动的相应生命周期回调函数中来避免资源泄漏。因此，该模块找出所有泄漏资源请求操作的资源操作活动，函数调用图和活动异步图用于实现这一目标。

资源的识别：要确定插入的释放操作应该接收的参数，修复过程需要跟踪身份参数的值。对于每个资源相关的操作，该模块引入一些辅助变量来记录其身份参数的值。

全局未释放资源列表的构建与维护：为了保证修复是安全的，已经申请但是还没有释放的资源应该被记录。该模块插入一些代码来创建和维护一个列表。应用在运行时会将所有已申请但是还没有释放的资源记录在这个列表中。

目标资源释放操作的插入：该模块在适当的位置为每个报告出的资源申请操作插入相应的资源释放操作以及相应的保护条件来完成修复。保护条件依据全局未释放资源列表列表来确定是否执行释放操作。

实验评估

应用名称	泄漏操作数	指令数量			DEX 大小		
		修复前	修复后	增量 (%)	修复前	修复后	增量 (%)
Bluechat	1	4218	4380	3.8	38.3	39.8	3.9
GetBackGPS	5	13177	15320	3.2	104.3	115.0	2.0
FooCam	1	11366	11818	3.9	574.1	576.9	0.4
ErWeiMaL	1	92329	92494	0.2	1443	1444	0.1
QiCaiScan	4	228205	233168	0.5	1192	1202	0.2
CheDengWo	1	200493	200655	0.1	1240	1241	0.1
WebPCSuite	2	321713	323619	0.3	2764	2769	0.1
FromCat	12	92236	93691	0.1	1023	1030	0.1
MaMa	1	95730	95895	0.1	576.0	577.3	0.2
XiaoMiWiFi	1	56554	57178	1.1	488.6	492.3	0.7
SuperTorch	3	177185	179103	0.3	1264	1265	0.1
FontMaster	1	68729	70243	2.1	932.4	940.3	0.8

修复真实安卓应用所带来的额外开销

应用名称	建图花费时间	插入指令花费时间	总花费时间
Bluechat	0.026	0.016	0.042
GetBackGPS	0.083	0.009	0.092
FooCam	0.164	0.070	0.234
ErWeiMaL	0.647	0.012	0.659
QiCaiScan	0.590	0.019	0.609
CheDengWo	0.760	0.034	0.794
WebPCSuite	0.535	0.004	0.539
FromCat	0.681	0.046	0.727
MaMa	0.515	0.015	0.530
XiaoMiWiFi	0.347	0.021	0.368
SuperTorch	0.976	0.007	0.983
FontMaster	0.637	0.013	0.650

修复过程花费的时间