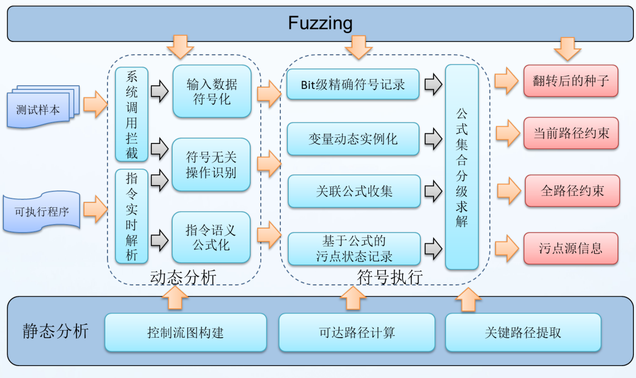
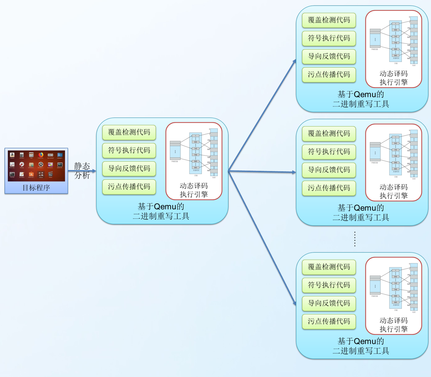
ReWrite-Fuzz:基于二进制代码重写的导向Fuzzing系统

杨轶，苏璞睿
{yangyi, purui@is.iscas.ac.cn}

ReWrite-Fuzz系统针对现有二进制代码模糊测试工作中面临的问题研发。主要包括：1）使用盲目数据变换，难以有效导向目标路径；2）路径翻转的求解过程依赖于全路径约束，易被非必经节点产生的约束干扰。3）Fuzzing和符号执行过程数据提取基于插桩实现，每条指令均中断，并与预定义的地址或指令数组进行对比，性能损失大。

ReWrite-Fuzz系统仅关注程序中可能将路径导向偏离目标的分支条件；提出二进制指令语义直接解析、符号无关指令识别方法，动态忽略了80%以上符号无关指令；提出了基于公式的污点状态记录方法，在不增加复杂度的条件下，同时实现符号执行与污点传播；提出了基于全路径约束、当前路径约束的分级求解方法，消除了非必经节点约束干扰。实现了基于动态重写的符号执行、污点传播、路径覆盖检查代码嵌入方法，将代码插桩转变为嵌入在二进制程序中的回调函数，进一步利用Linux系统fork机制，极大提高了系统分析效率。

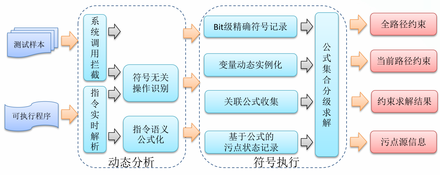




系统主要包括程序预处理、控制依赖分析、输入数据符号化、关联逻辑公式收集、公式集合分级求解、符号数据源回溯等功能，具备针对Linux操作系统上的大型应用软件的动态逆向分析能力。系统具有如下的技术特性：

- 基于bit级符号化表示与求解，分析精度高；
- 结合符号执行精确翻转关键路径，降低符号执行的复杂度；
- 支持命令行参数、文件、网络输入变量符号化，数据获取能力强；
- 全约束求解与简化约束求解分级实现，不受前序路径中非必要约束干扰。

该系统目前已经应用于漏洞挖掘，在发现了binutils、gpac、libpng等广泛使用的应用和程序库中的多个未知漏洞，已经得到开发人员确认。


测试样本
可执行程序
系统调用拦截
指令实时解析
符号无关操作识别
指令语义公式化
动态分析
Bit级精确符号记录
变量动态实例化
关联公式收集
基于公式的污点状态记录
公式集合分级求解
符号执行
全路径约束
当前路径约束
约束求解结果
污点源信息


目标程序
覆盖检测代码
符号执行代码
导向反馈代码
污点传播代码
基于Qemu的
二进制重写工具