学号: DZ1933026

姓名: 王国畅

技术

- java.lang.instrument.Instrumentation
- javassist
- org.objectweb.asm

挑战与解决方案

1. 在类加载时劫持回调函数：使用Instrumentation框架，实现自己的classFileTransformer，通过javaagent静态加载jar包来修改字节码。
2. 在类加载时获取字节码：使用ASM框架，在MethodVisitor中override visitInsn可以获取所有被执行的字节码，override visitFieldInsn可以从参数获取Field相关指令的更多参数。
3. 在visitInsn/visitFieldInsn内，可以通过调用super.visitInsn等API来插入Opcode，从而可以通过jvm的opcode实现向trace的调用。
4. 在实现对trace的函数调用时，需要获取当前Insn的Opcode操作数信息，如IALOAD的arrayReference和index信息。在已知Opcode的情况下，通过查询ASM用户手册可知当前堆栈情况，从而调整栈顶帧，将需要的参数信息复制到栈顶。例如LASTORE和DASTORE:

```
//这里想起了ics的pa
    case Opcodes.LASTORE://80
    case Opcodes.DASTORE://82
        //stack top: o,i,w
        mv.visitInsn(Opcodes.DUP2_X2);
        //stack top: w,o,i,w
        mv.visitInsn(Opcodes.POP2);
        //stack top: w,o,i
        mv.visitInsn(Opcodes.DUP2_X2);
        //stack top: o,i,w,o,i
        mv.visitMethodInsn(Opcodes.INVOKESTATIC, "mtrace/wgc/demo/MTrace",
"traceArrayWrite", "(Ljava/lang/Object;I)V", false);
        break;
```

5. 由于trace函数实际不存在于被插装程序的searchpath中，这里通过javassist向待插装程序的classPool添加trace所在类的classPath信息。

```
    ClassPool classPool = ClassPool.getDefault();
    classPool.insertClassPath(new ClassClassPath((MTrace.class)));
```

6. 由于运行时对库函数插装会带来不必要的麻烦，这里过滤了以java，sun，jdk，mtrace开头的class:

```
if(className.startsWith("java")||className.startsWith("sun")||className.st
artsWith("jdk")||className.startsWith("mtrace")){
        return null;
    }
```

附录

可执行文件说明

```
mtrace-1.0-SNAPSHOT.jar -> 已打包源代码
tracetest.jar -> 简易testcase
```

testcase运行结果

```
W       1       63961c4200000000        int[0]
W       1       63961c4200000001        int[1]
W       1       63961c4200000002        int[2]
W       1       63961c4200000003        int[3]
W       1       63961c4200000004        int[4]
W       1       63961c4200000005        int[5]
W       1       63961c4200000006        int[6]
W       1       63961c4200000007        int[7]
W       1       63961c4200000008        int[8]
W       1       63961c4200000009        int[9]
W       1       3e3abc8800000078        SomeClass.x
W       1       3e3abc8800000079        SomeClass.y
R       1       63961c4200000000        int[0]
R       1       63961c4200000003        int[3]
W       1       3e3abc8800000078        SomeClass.x
W       1       3e3abc8800000079        SomeClass.y
R       1       6ce253f10001af4e        java.lang.System.out
R       1       3e3abc8800000078        SomeClass.x
R       1       3e3abc8800000079        SomeClass.y
Hello World! someobj at x: 0 y: 0
R       1       6ce253f10001af4e        java.lang.System.out

R       1       6ce253f10001af4e        java.lang.System.out
W       1       53d8d10a00000000        java.lang.Object[0]
0000000002
R       1       6ce253f10001af4e        java.lang.System.out
type1 is string: true
W       1       7229724eb95be7b5        RunnableDemo.threadName
W       1       7229724f05fb28d2        RunnableDemo.index
R       1       6ce253f10001af4e        java.lang.System.out
R       1       7229724eb95be7b5        RunnableDemo.threadName
Creating Thread-1
R       1       6ce253f10001af4e        java.lang.System.out
R       1       7229724eb95be7b5        RunnableDemo.threadName
Starting Thread-1
R       1       7229724f00000074        RunnableDemo.t
R       1       7229724eb95be7b5        RunnableDemo.threadName
W       1       7229724f00000074        RunnableDemo.t
R       1       7229724f00000074        RunnableDemo.t
```