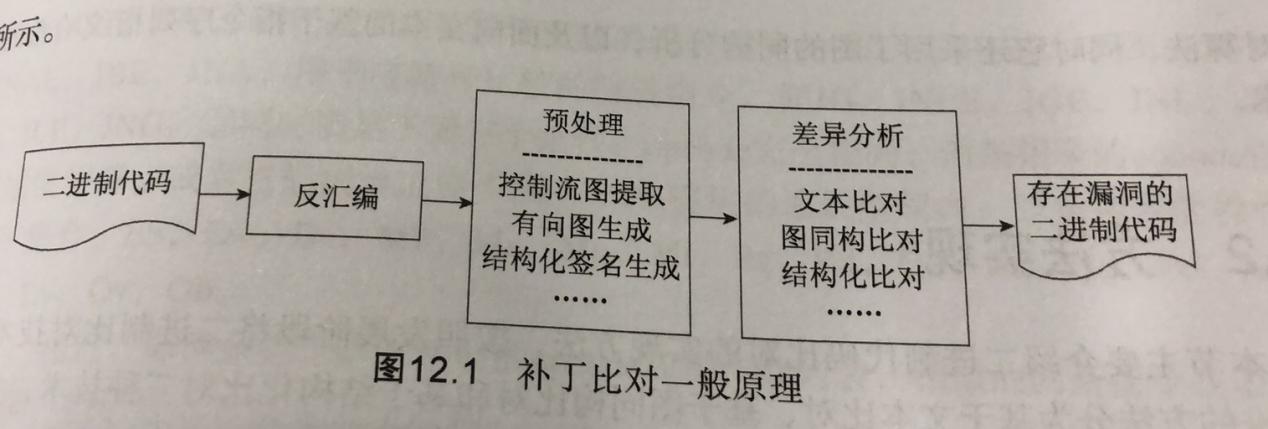
# 《软件漏洞分析》第12章读书报告

518021910698 薛春宇

Chapter Twelve 基于二进制代码比对的漏洞分析

1. 基本原理
2. 补丁比对技术原理：
3. 定义：补丁比对技术是一种通过比对打补丁前后的目标代码，分析发现两个版本的程序之间差异的技术；
4. 过程图：



1. 二进制比对技术原理：

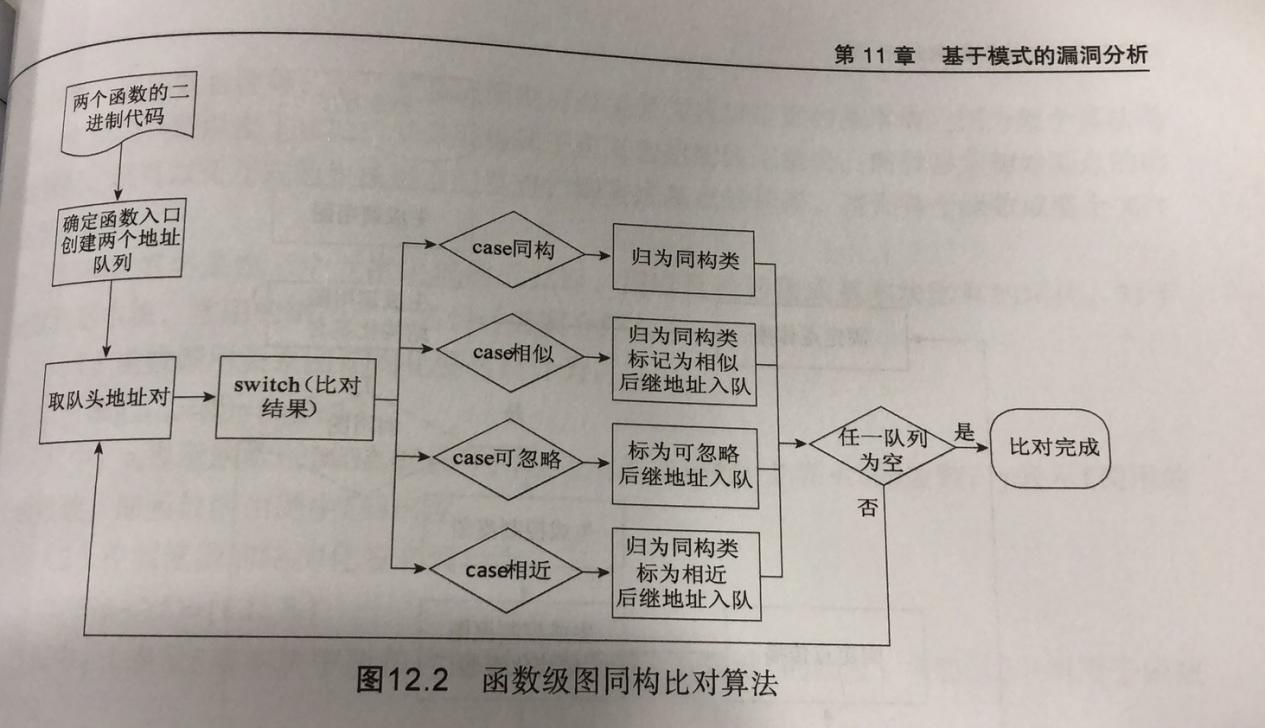
实现原理：

1. 基于文本的比对：对象分为两种（二进制文件和反汇编代码）；
2. 基于图同构的比对：目的（为了克服基于文本比对无法提供二进制程序语义相关信息的情况）；过程（对可执行程序的控制流图进行抽象，将二进制程序转化为一个有向图 -> 根据图论的知识进行解决）；
3. 基于结构化的比对：注重点是可执行文件逻辑结构上的变化，而不是某一条反汇编指令的改变；
4. 综合比对技术：多种比对技术的综合应用；
5. 方法实现
6. 基于文本的二进制代码比对：
7. 二进制字节的对应比较：最常用的算法是最长公共子序列算法（P310）；
8. 反汇编文本比较：实际上是一种指令级别的比对方法，它把汇编指令作为分析的对象，研究指令之间的相似性和差异性，指令之间的比较关系可以分为“相似”（两条指令的语义完全相同）、“相近”（即两条指令具有相同的opcode和操作数列表，相似的两条指令在语义上不完全相同，但已经非常接近完全同构）、“可忽略”（如果某条指令为NOP指令，或者是只有唯一后继节点的JMP指令，那么该指令则可以被标记为“可忽略”）、“不同”（两条指令中的一条被标记为“可忽略”）；
9. 基于图同构的二进制比对：
10. 指令级图同构比对算法：基本思路（两个需要比较的可执行文件分别构造成图，以指令、数据常量、函数调用指令等作为顶点，以控制流图的边作为图的边。对生成的两个图做同构识别，用同构算法找到最相似的两个部分作为同构部分。然后，对两个图的非同构部分继续识别其是否同构，直至全部识别结束。不同构的部分就可能是两个版本可执行文件的不同之处）；

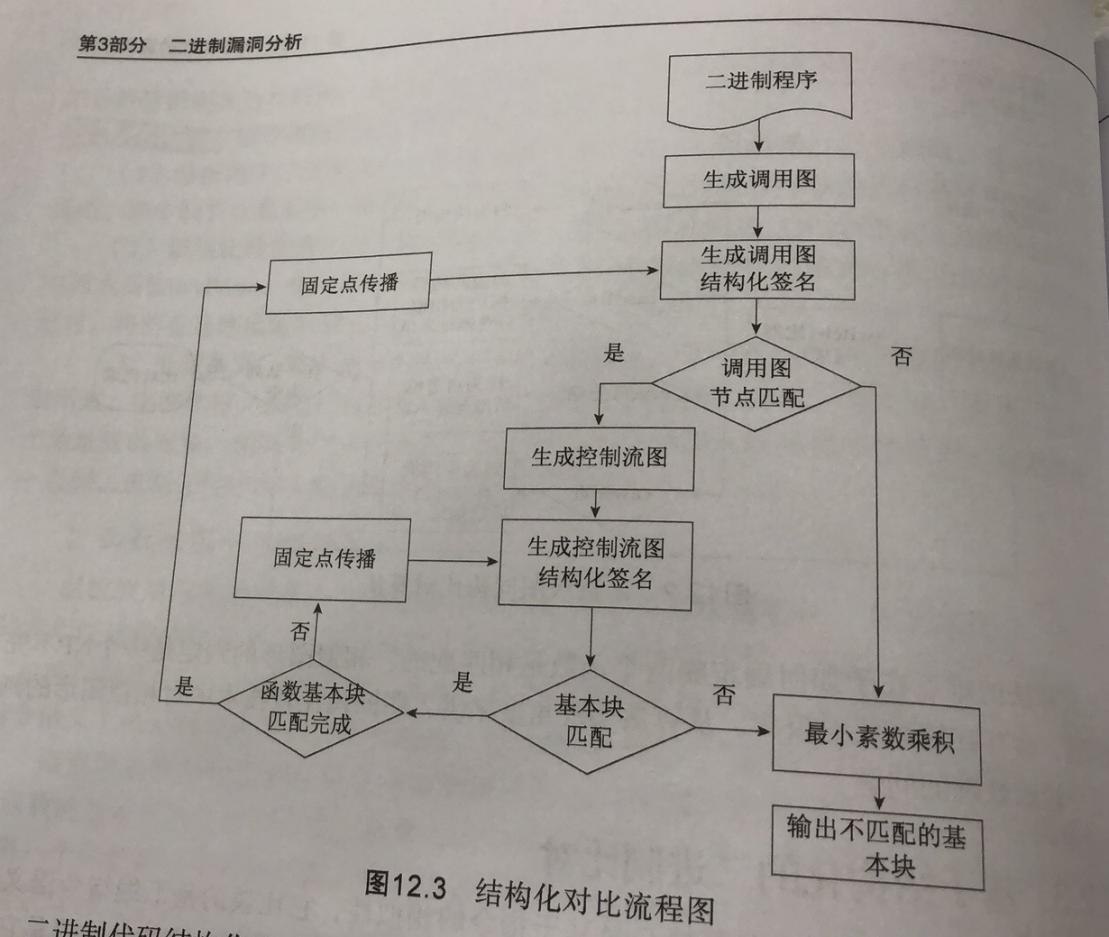
|  |
| --- |
| 指令级图同构比较算法： |
| （1）分析两个二进制文件，获得函数、引用表、字符串等。对于函数，生成函数流程图。图中的节点表示单一指令，边表示指令间所有可能的执行顺序； |
| 1. 识别比较的开始点。对于PE文件，可以用IMAGE\_OPTIONAL\_HEADER结构中的成员EntryPoint，也可以分析导出函数表，匹配相应的导出函数，作为比较的起始地址对。将所有这些地址对放入一个分析队列中； |
| （3）处理队列，对于每一个需要比较的地址对，分析该地址处对象的类型：函数、引用表、字符串等，并进行对应的比较。如果对象是不同的类型，则失败；否则，比较两个地址处的对象，相同则加入到同构图中，并向队列中加入新的比较地址对。如此类推一直到二进制文件中没有相应的地址为止。 |

1. 函数级图形化比对算法：

|  |
| --- |
| 1. 以两个函数的入口地址为开始，维护一个地址对队列； |
| （2）如果队列非空，从队列中弹出一对地址，比较两个地址处的指令：如果已经同构，转至第（2）步；如果地址对应的指令“相似”，归为同构一类并标记为“相似”，然后将两个对应的后继地址添加到队列中，转至第（2）步；如果两条中的指令有一条为“可忽略”，将忽略的指令放入忽略节点集合中，并将忽略指令的后继指令的地址与另一条指令地址组合成一个地址对加入队列，转至第（2）步；如果两条指令是“相近”的，归为同构一类并标记为“相近”，然后将两个对应的后继地址加入队列，转至第（2）步； |
| （3）两个函数比对完成，判断两个函数对应的图形是否同构； |



1. 基于结构化的二进制比对：



1. 基于结构化签名的二进制比对；
2. 基于可信基点的结构化签名二进制比对；
3. 软件补丁的二进制比对技术（P317）；
4. 实例分析（P318）；
5. 典型工具：
6. Bindiff；
7. Eye Binary Diffing Suite；