**软件漏洞分析技术读书报告**

姓名：仝志欣

|  |
| --- |
| **本周预期完成任务：**阅读软件漏洞分析技术的第四章内容—数据流分析 |
| **本周实际完成进度：**了解了数据流分析检测程序漏洞和数据流辅助漏洞分析的方法实现过程；了解三款采用数据流分析技术进行源代码漏洞分析的典型工具。 |
| **详细内容：**   1. **基本原理**   **1. 基本概念**  该分析是一种用来获取相关数据沿着程序执行路径流动的信息分析技术，分析对象是程序执行路径上的数据流动或可能的取值。包括流不敏感、流敏感、路径敏感分析，其中后两者考虑较为周全，常被采用。还可分为过程内分析和过程间分析，字面意思。  **2. 技术框架**    建模图  ①静态漏洞分析：将程序代码模型作为分析对象，将漏洞分析规则作为检测程序漏洞的依据。  ②处理分析结果：为追求分析效率，漏洞分析工具同时应用多种检测规则对代码进行检测。此外，每个分析过程得到的精确程度不一样，这是由于会使用近似分析的情况。  3. 辅助支持技术  构建相对完整的程序调用图、分析变量的别名、数据流分析完善程序模型。数据流分析作为辅助分析时该分析主要用于构建调用图、分析变量别名。   1. **数据流分析方法实现** 2. **使用数据流分析检测程序漏洞**   主要是将经过基本分析和辅助分析得到的程序代码模型和预先定义的分析规则作为输入，根据匹配输出初步的漏洞分析结果。在这一过程中结尾处，静态漏洞分析方法比较重要。包括程序代码模型、程序建模、漏洞分析规则、具体静态漏洞分析几个模块。  (1) 程序代码模型  ①抽象语法树  描述程序语句的句法结构。  ②三地址码  将运算语句全部转化为只含有一个运算符的语句  ③静态单赋值形式  通常指静态的三地址码，除此之外，抽象语法树或者程序的源代码也可以表示为静态单赋值形式。  控制流图  控制流图是描述控制流的模块图，经典的控制流图的节点是基本块。在绘制控制流图时要考虑异常的话就会比较复杂。  调用图：调用图是描述程序中国城之间的调用和被调用关系的有向图。  (2) 程序建模  程序建模是构建上述模型的过程。包括代码解析和辅助分析两个部分。代码解析是指词法分析、语法分析、中间代码生成和过程内控制流分析等代码分析，辅助分析指控制流分析。控制流分析导出控制流图和调用图。  (3) 漏洞分析规则  漏洞分析规则主要包括规则的自动状态机和变量取值的相关检测规则。前者对应程序变量检测，后者针对取值范围。  (4) 静态漏洞分析规则  这一块比较复杂，但主要包括两个方面即过程内分析和过程间分析。  过程内分析：分析时对不同类型的程序语句进行不同的处理：赋值语句、控制转移语句、过程调用语句。另一方面，在分析时要选择路径，在这点有两种遍历方法：深度优先、宽度优先。  过程间分析：运用调用图。有三种思路：遇到调用语句就进去分析，完成分析之后再回到原来的程序继续分析；第二个思路是针对第一种思路中的递归，就是分析不动点；也可以使用队列进行。  处理分析结果：机器分析过之后要进行人工处理；分析结果要考虑程序漏洞的危害性；可以对发现的漏洞进行简单的分类。   1. **数据流分析作为辅助技术的漏洞分析**   其实这一点我觉得有点难以理解的，我的理解是：指向分析等是特殊的数据流分析，这些分析用来使调用图或者控制流图更加精确，而这种图对于常见的漏洞分析很重要，所以在这里数据流分析作为辅助。在指向分析中难点在于在进行过程间分析时使用摘要对上下文进行区分，具体以看课本为主；在取值分析中，文章给出两种方法来分析变量：常量传播、逆向分析变量。两者具有不同的适用范围。   1. **典型工具**   **1. Fortify SCA**  Fortify SCA 是一个软件源代码缺陷景台测试工具。它通过分析用用程序可能会执行的所有路径，从源代码层面上识别软件的漏洞，并提供完整的分析报告。它由四部分组成：前端、分析引擎、分析规则库、控制管理界面。其中前端是用来转化程序，转化成该引擎能够处理的数据结构，分析引擎是判断漏洞的主要部分，工作台是负责整个分析过程，对从引擎得到的检查结果进行处理。使用该工具的步骤：选择扫描对象、处理FPR文件并查看漏洞分析结果。  **2. Coverity Prevent**  该工具与第一个工具原理上比较相似。步骤：编译器处理每一个源文件，编译器解析代码后在中间目录中生成代码的模型，然后分析引擎分析源代码生成缺陷报告，最后这些报告提交数据库，管理人员可以通过浏览器访问缺陷，并可以和其他缺陷管理系统集成。   1. **FindBugs**   该工具是针对Java语言的的静态代码分析工具，并不专门针对安全漏洞的分析和检测，它侧重于有效的发现程序中存在的设计问题，它还涉及一些非安全相关的程序设计缺陷，如线性竞争问题和性能问题等。分析目标是Java字节码，工作流程如下：接受的对象如果是Java文件并没有进行编译操作，那么实际测试对象为.class文件，如果没有.class文件，无法进行检测。接到对象后工具会将测试对象构造成一个工程然后根据选择的检测器形成分析计划。根据分析计划，FindBugs调用检测器对.class文件进行检测。  其中检测器有很多种分类包括类结构和类的层级结构的分析、线性扫描代码、程序结构分析、数据流分析等。该工具所采用的分析方法较为简单，更倾向于使用简单的、应用范围更广的技术，而不是使用复杂的、针对特定程序缺陷的技术。该工具并未进行较为深入的数据流与控制流分析，主要依靠将代码与缺陷类型进行模式匹配来检测缺陷。 |
| **相关文件及参考资料：** 软件漏洞分析技术的第四章内容 |
| **遇到的问题及该阶段学习的想法：**第四章介绍了用数据流技术检测漏洞和用数据流技术作为辅助技术静态分析漏洞的方法。文章比较详细地介绍了各种数据流分析技术的具体思想，并结合实例帮助理解，最后还介绍了三种分析工具，这三种分析工具都是与本章密切相关的工具。用数据流分析检测漏洞仍然会存在漏报和误报。尽管如此它仍是一些主流漏洞分析系统使用的主要技术。 |
| **接下来的进度计划：** 下一周通读第五、六章 |