**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十章内容 |
| **本周实际完成进度：**了解动态污点分析技术的一些关键内容，及两个典型的工具。 |
| **详细内容：**   1. **基本原理**   **1. 基本概念**  污点分析技术基本原理：是将来自于网络、文件等非信任渠道的数据标记为“被污染的”，则作用在这些数据上的一系列算术和逻辑操作而新生成的数据也会继承原有数据的“被污染的”属性。通过对这些属性进行分析可能够得出程序的某些特性。可分为静态污点分析和动态污点分析。  动态污点分析技术：该分析技术是在程序运行的基础上,对数据流或控制流进行监控,从而实现对数据在内存中的显式传播、数据误用等进行跟踪和检测。动态污点分析技术可以从流和使用范围两个方面来分类。  根据流：基于数据流的分析技术和基于控制流的分析技术。  根据使用范围：用户进程级的动态污点分析和系统级的动态污点分析。  **2. 动态污点分析基本原理**  定义source、sink点。动态污点分析技术与静态分析技术唯一的区别在于静态污点分析技术在检测时并不真正运行程序，而动态污点分析技术需要运行程序并同时传播和检测污点标记。  一般的分析流程：污点数据标记、污点动态跟踪、污点误用检查。  方法特点（见于p251页）   1. **方法实现** 2. **污点数据标记** 3. 重要的污点数据：从粒度上讲,监控主要分为指令级监控和函数级监控,目前普遍采用的方法是在**关键函数**处实行监控。在漏洞分析过程中,污点数据通常主要是指软件系统所接受的外部输入数据,在计算机系统中,这些数据可能以内存临时数据的形式存储,也可能以文件的形式存储。**当软件系统需要使用这些数据时,一般通过系统调用来进行数据访问和处理**。以 Windows系统为例,几乎所有软件都最终会通过 Windows的API来访问文件。因此只需要对文件访问的API进行监控,就可以分析软件读取或者输出了什么污点信息。常用的API函数有：NtReadFile、NtWriteFile、Send 4. 污点数据标记的方法   识别出污点数据之后，要对污点进行标记。引入污点生命周期的概念，即在该生命周期内，污点被定义为有效。在动态污点分析中有污点创建和污点删除。  污点创建  污点删除   1. **污点动态跟踪** 2. 污点数据传播规则   当污点数据从一个位置转移到另一个位置，则认为产生了污点传播。具体的污点传播规则可参见表10.1（见于p255）  (2)污点跟踪数据结构  针对对象级信息，是在函数监控的基础上进行视图重建，比如获取文件对象和套接字对象的详细信息。  (3)污点动态跟踪的实现  文章中用污点传播树来描述污点之间的传播关系，污点传播树的构建过程对应于污点的传播和处理过程。在污点分析过程中,采用污点分析中常用的基于影子内存的污点分析方法,将影子内存用于存放当前时刻存在的有效污点。所谓影子内存,是指在程序动态执行过程中,记录下的某些真实内存的镜像。基于影子内存的污点分析技术的基本思想,是将程序执行过程中产生的污点全部用影子内存的方式记录下来,以便分析各污点之间的动态传播关系。  为了快速、准确地利用记录的轨迹信息构建污点传播树,可以利用内存hash的方法来实现影子内存中污点存放位置的快速查找。当遇到会引起污点传播的指令时,首先对指令中的每个操作数都通过污点快速映射查找影子内存中是否存在与之对应的影子污点从而确定其是否为污点数据,然后根据前面介绍的污点传播规则得到该指令引起的污点传播结果,并将传播产生的新污点添加到影子内存和污点传播树中,同时将失效污点对应的影子污点删除。  (4)污点净化规则。污点净化,就是在处理某些指令时,虽然有操作数的取值是污染的,但是结果却可能是不污染的。从数据流分析的观点来看,污点数据通过该指令操作后,得到了净化。在对污点数据进行传播分析时,一定要对这些可能进行污点净化的指令进行特殊考虑。例如逻辑运算类指令的污点传播情况，需要考虑结果值是否依赖于受污染的输入数据，如and、or、xor等。  (4)动态代码插桩  根据时机不同，可分为：源代码插桩、编译时插桩、链接时插桩和运行时插桩。   1. **污点误用检查** 2. 污点敏感点   即sink点，主要可分为：跳转地址、格式化字符串、系统调用参数，这三种检测可捕获大部分基于漏洞的攻击。另外，还有两种类型的检测：跟踪标志位是否被感染，检测被感染的标志位是否用于改变程序控制流 检测数据移动类指令的地址是否被感染   1. 污点误用规则   在进行污点误用检查时，通常需要参照一些规则，这些规则称为“漏洞模式”。比如，对于函数原型“char \*strcpy(char \* dest,char\* src)”，一般检查参数“src”是否被污染，而不是检查参数“dest”是否被污染。常见软件漏洞类型有缓冲区溢出（栈溢出和堆溢出）、整数溢出、命令注入、格式化字符串、sql注入、跨站点脚本等。   1. **典型工具**   **1. TaintCheck**  **2. Argos** |
| **相关文件及参考资料：** 软件漏洞分析技术的第十章内容 |
| **遇到的问题及该阶段学习的想法：这个章节的内容跟之前静态污点分析是非常相似的，只不过动态污点分析是存在程序执行的，更加具有难度。** |
| **接下来的进度计划：** 阅读第十一、十二章 |