软件安全

研究报告

东北大学

题目名称	软件安全研究进展之检测技术
班　　级	软信-1503
学　　号	20155362
姓　　名	薛旗
日　　期	2018.05.10
成　　绩	
评 阅 人	

软件学院

软件安全研究进展之检测技术

引言：

随着信息技术的发展，软件作为人与机器交互的重要媒介，发挥着越来越重要的作用。软件在为人们生活提供便利的同时，也带来了新的威胁，如信息泄露，信息篡改等。加强软件安全管理，及时发现并处理软件缺陷漏洞，是目前软件管理部门及用户追求的方向。软件安全检测技术是发现软件问题的重要环节，是修复软件的基础，其已成为软件行业非常重要的一项工作。本文将介绍软件安全检测技术的最新研究进展。

软件安全检测技术及研究进展：

一、基于模型的软件安全检测技术

基于模型的安全性测试是对软件的结构和行为进行建模，生成相应的测试模型，再由测试模型自动生成测试用例，以驱动安全性测试。

在安全分析的过程采用的关键技术主要包括模型表示技术、软件抽象技术和安全检测技术。其中模型表示技术是为了描述软件中出现的缺陷和脆弱点，把现实软件中的安全特性或缺陷表示成公式化的形式；软件抽象技术是软件结构按指定要求进行简化，提高软件安全检测的效率；安全检测技术要求在软件安全检测的过程中，首先通过模型表示技术即安全建模技术把安全特征表示成安全模型，然后该模型与软件抽象技术结合对软件进行简化，最后对简化后的软件实施安全分析，产生安全报告。

常用的软件测试模型有有限状态自动机、UML 模型和马尔可夫链等。以 UML 模型为例，主要方法流程如下：

1.分析被测试软件，根据测试目的，确定测试对象和测试特征；

2.选择和构造 UML 模型，该 UML 模型表述了需求所表述的所有可能行为；

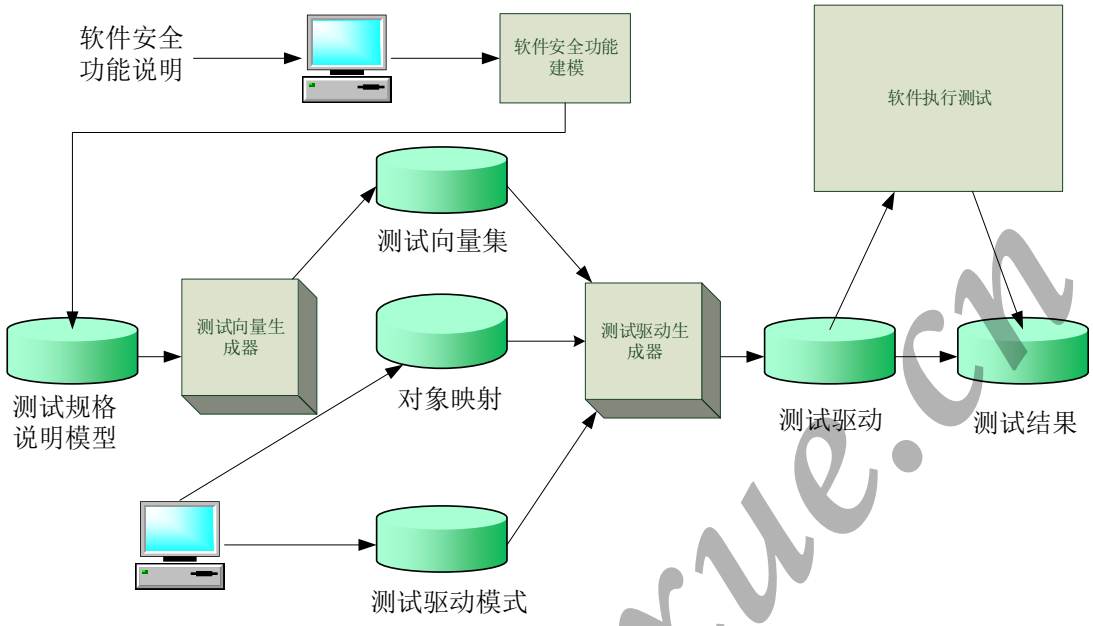
3.对 UML 模型进行验证，排查 UML 模型构造时可能出现的有界性、安全性、死锁和状态可达性，确保 UML 模型的正确性；

4.通过深度优先搜索算法遍历 UML 模型，自动生成测试用例，根据充分性准则计算相关的覆盖率，完成对测试用例的评估；

5.根据待测程序和所述 UML 模型得到的测试用例生成测试脚本，自动执行所述测试脚

本，并保存执行测试脚本得到的实际输出结果；

6.根据测试用例的实际输出与预期输出的比较，得出测试结果，再根据测试目标与预先设定好的停止准则，决定是否需要修改模型或修改待测程序。

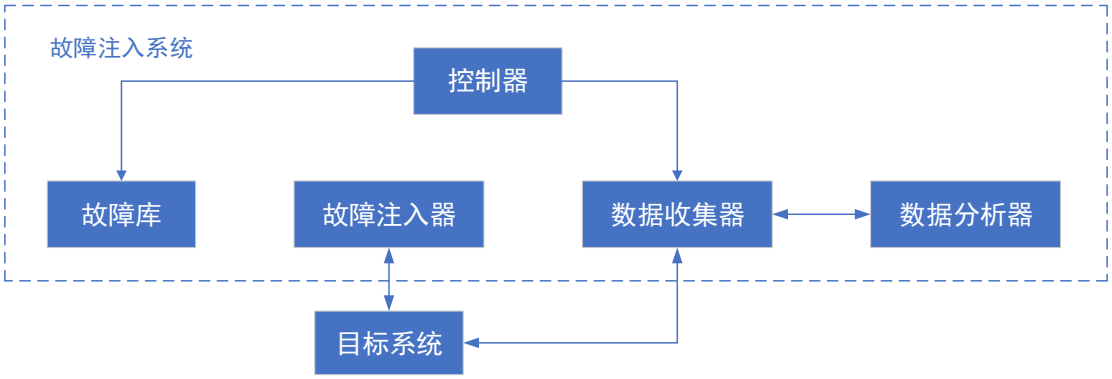


自动化安全测试处理流程

基于模型的软件安全性技术适用范围取决于安全功能的建模能力，对授权、访问控制等安全功能测试比较适用。

二、基于故障注入的软件安全检测技术

故障注入是评测容错机制的一种有效方法，其原理是在机器指令可以访问到的范围内，通过人为方式将故障引入到系统当中，用软件方法通过修改软件或硬件的状态变量或数据来模拟故障的产生，加速系统的失效。故障注入可以有效地模拟各种异常程序行为，通过故障注人函数能强制使程序进入某些特定状态，而这些状态在采用常规的标准测试技术时是无法达到的。针对应用与环境的交互点，主要包括用户输入、文件系统、网络接口、环境变量等引起的故障。



故障注入系统的一般结构图

软件故障注入不需要增加昂贵的附加硬件，不会损坏目标系统的硬件环境，容易回收数据，且能够方便地跟踪目标程序的执行，既能注入软件故障也能注入硬件故障，而且还能把故障注入于操作系统之中。当前故障注入的主要方法有如下几种:

1.仿真故障注入：需要以较好的目标系统仿真模型为基础;

2.硬件故障注：需要专业的硬件设备;

3.软件故障注入：对目标系统硬件环境没有任选损坏，能方便地跟踪目标程序的执行并回收数据，系统开稍小，且有较好的可移植性;

4.基于环境混乱的故障注入：把应用程序和其运行环境都纳入系统的范畴，通过改变正常的环境因素(如文件，网络等)来测试系统对环境故障的客错能力。

根据注入方式，软件故障注入可以分为动态注入和静态注入。静态注入指的是在目标程序的内存映象被加载和执行前，通过程序插桩方式改变被验代码，通过按预设策略的插桩代码变异将故障注入到目标程序中；动态注入是在程序运行期间，在特定的状态或条件下，通过某种触发机制，触发故障注入，使程序将要执行的指令和数据等发生变化。

与其他故障注入方法相比，软件故障注入有实现简单、灵活、工作量小、评测范围大、不损害原系统等特点，有广泛的应用前景。

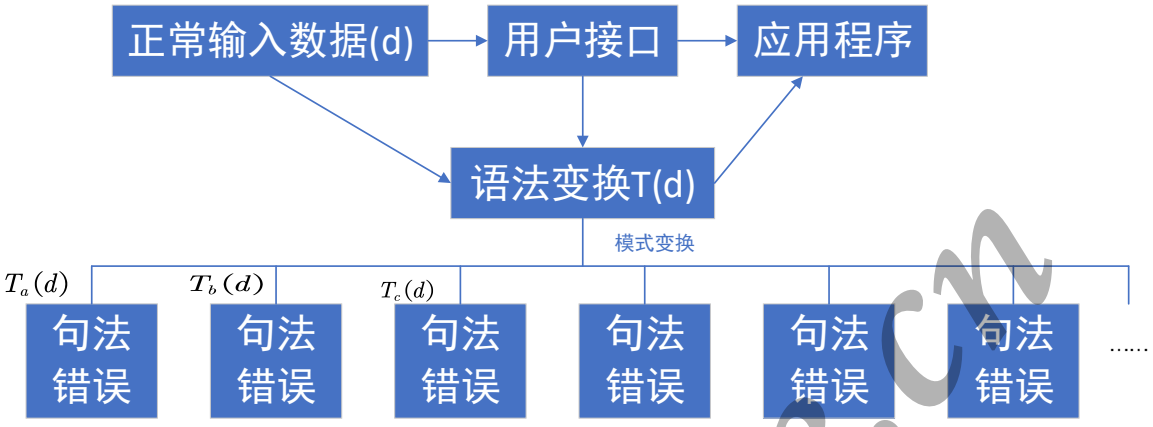
三、基于语法的软件安全检测技术

语法测试是根据被测软件的功能接口的语法生成测试输入，检测被测软件对各类输入的响应。测试输入接口可以有多种类型，如命令行、文件、环境变量、套接字等。语法测试适用于被测软件有较明确的接口语法，易于表达语法并生成测试输入的情况。语法测试结合故障注入技术可得到更好的测试效果。语法测试步骤如下:

1.识别被测软件接口的语言，定义语言的语法;

2.根据语法生成测试用例并执行测试。生成的测试输入应当包含各类语法错误，符合语法的正确输入，不符合语法的畸形输入等；

3.通过察看被测软件对各类输入的处理情况，确定被测软件是否存在安全缺陷。



四、基于 XML 的软件安全检测技术

软件静态检测是从软件代码和结构中找出安全缺陷的重要手段。从安全规则的角度，提出了基于 XML（eXtensible Markup Language）中间模型的静态检测方法。该方法将 C/C++源代码解释为 XML 中间模型，将安全规则转化为缺陷模式，利用 Xquery 查询表达式对软件安全缺陷进行定位。基于该方法的原型系统检验结果表明：该方法能够有效地检测出违反安全规则的软件缺陷，并具有安全规则可定制的特点。

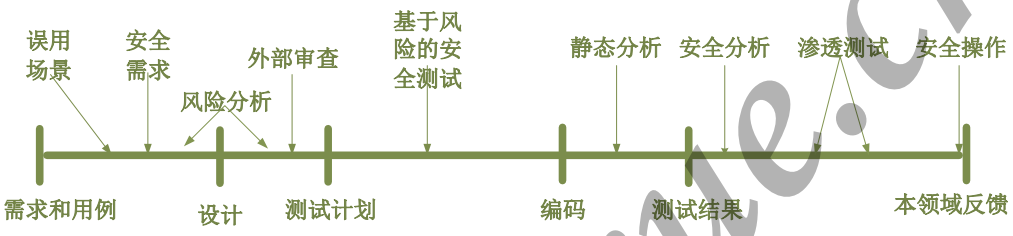
我们知道，C/C++语言并不是一种安全编程语言，一个重要的原因在于其标准中存在大量未定义行为和不安全用法，使用不当将产生严重的安全隐患。当前，避免这些安全隐患的通常做法是制定针对 C/C++语言编程的安全子集，在编写代码阶段加以限制和规范。同时，研究开发代码静态检测工具，通过对源代码的分析自动检测安全隐患，既能提高检测效率，也可降低检测成本。

目前，国外在 C/C++语言安全子集和代码静态检测方面已进行了大量的研究，定义了许多有代表性的安全子集，并设计了基于安全子集的代码静态检测工具，应用在航空、医疗以及运输等安全关键领域。通过对航天型号软件安全性标准《GJB 5369-2005 航天型号软件 C 语言安全子集》的深入研究，提出了相应的代码静态检测方法。通过对源代码进行语法制导的解析，利用 XML（eXtensible Markup Language）在数据存储和数据交换中的优势，将源代码转化成 XML 中间模型。把安全子集中的每一条规则抽象为缺陷模式，使用 Xquery 查询语言将缺陷模式转换成 Xquery 表达式，利用 Xquery 表达式查询和定位 XML 中间模型中与缺陷模式匹配的节点，通过缺陷重定向机制完成缺陷从 XML 中间模型到源文件中的精

确定位。基于此方法开发的自动化检测工具 CSV 的实验表明，该方法能够有效地检测出违反安全子集的所有软件缺陷。通过系统提供的规则定制接口，也实现了安全规则的自由配置与扩展，增强了系统的实用性。

五、基于风险的软件安全检测技术

风险是错误发生的可能性和造成的危害程度的结合。基于风险的软件安全检测技术是以软件安全风险作为测试的出发点和测试活动的主要参考依据，把风险分析与管理、安全测试以及软件开发过程统一起来，在软件开发的各个阶段中就把有风险的安全漏洞考虑在内，将安全测试与软件开发同步进行。



通过误用模式、异常场景、风险分析以及渗透测试等技术来处理具有风险的安全问题。基于风险的测试以软件模块的质量风险为主要参考依据，来进行测试力量的分配，可以尽早地发现尽可能多的潜在安全问题，以最少的资源、最短的时间有效地达成用户需求，并确保合适的软件品质以避免大量的后期维护工作。

六、基于渗透的软件安全检测技术

渗透测试(Penetration Testing)是一个评估主机系统和网络的安全性时模仿黑客特定攻击行为的过程，安全测试工程师尽可能真实地模拟黑客使用的漏洞发现技术和攻击手段，对目标的安全性作深入的探测，发现系统最薄弱环节的过程。渗透测试一般可分为两类，被动攻击和主动攻击。被动攻击不采用直接进入目标系统的方式去收集信息，主动攻击则直接侵入目标系统或网络内部收集所需要的信息。

一次完整的渗透测试步骤为：

1.收集信息；

2.抽取出对网络渗透有用的信息；

3.制定渗透策略并进行漏洞测试；

4.模拟攻击的真实过程；

5.最后整理收集到的信息并提交测试报告。

渗透测试比较真实有效，发现的漏洞一般都是真实存在的，而且较为严重，但是由于模

拟的测试只能达到有限的测试点，所以渗透测试的覆盖率较低，且漏报率较高。

结语：

随着互联网技术的不断创新与发展，倒逼软件安全检测技术的向前推进。随着人们对软件安全检测的重视，软件安全检测已成为一项极为重要的工作。现在软件安全检测技术已有比较大的发展，比较成熟的算法、技术、理论都开始逐渐出现，相应的检测技术也越来越高。但总体而言，软件的安全检测技术发展还不够成熟，还有很大的发展和改进空间，新的算法、理论或者猜想将会进一步完善软件的安全检测，需要我们继续研究和探索。