基于模糊动态测试的支付平台逻辑漏洞自动挖掘技术研究

（此文章仅为毕业设计开题阶段前结果）

邵之凯

(电子科技大学计算机科学与工程学院)

**摘 要** 随着互联网的发展，线上支付逐渐普及，其安全问题的重要性也逐步提升。而支付软件的开发，总是无法避免软件漏洞的存在。逻辑漏洞便是其中一种较难检测的漏洞类型，并且大量出现在支付软件当中。本文将对现有的动态检测技术进行分析，并了解常见支付流程和漏洞。再此前提下，基于动态模糊测试技术与前后端联合架构，设计一个减少对测试人员影响的、对现有支付平台常见逻辑漏洞通用的检测软件，并完成其部分工作。

**Abstract** With the development of the Internet, online payments are gradually becoming more and more popular. The importance of their security is steadily increasing. The development of payment software has always been inevitable in terms of software vulnerabilities. Logical vulnerabilities are one of the more difficult types of vulnerabilities to detect. They always exist in the payment softwares. In this paper, we will summarise existing dynamic testing techniques and analyse common payment processes and vulnerabilities. Based on this premise, a generic detection software for common logic vulnerabilities in existing payment platforms will be designed. The detection is partially completed based on dynamic fuzzy testing techniques and a joint front and back-end architecture that reduces the impact on testers.

**关 键 词** 支付平台、逻辑漏洞挖掘、动态模糊测试

# 引言

随着互联网普及，以及全方位社会数字化转型，在线支付逐渐成为了人们生活中不可或缺的部分，也是软件开发者、运营者从中获利的重要途径之一。然而，随着在线支付、第三方支付的发展，针对它们的攻击也层出不穷，甚至出现了一些自动化攻击软件，去攻击其中的漏洞。

软件漏洞是程序开发中无法避免的部分，对漏洞的查找和修补也产生了的巨大的工作量。现阶段，虽然也出现了大量的自动漏洞查找工具，但是逻辑漏洞一直是一个难以攻克的难题[5]。逻辑漏洞是应用程序中特定的功能逻辑错误或缺陷导致的可被攻击者利用的程序漏洞，属于一种软件设计上的漏洞。它不存在通用化的特征，因此一般无法直接从源代码中识别出来。支付漏洞是一种常见的逻辑漏洞[6][11][12]，现阶段依旧没有一种很好的检测方式，能够做到通用且有效地对支付模块进行检测，因此该课题存在着较大研究意义。

本文将会通过使用前后端联合扫描架构，利用后台信息进行指向性的动态模糊检测，从而达到有效且减少对测试人员影响地对目标程序进行逻辑漏洞挖掘。

# 背景知识总结

### 动态检测技术现状

2.1.1 动态监测的常用方法

动态检测[9]是一种在操作的同时，依靠控制软件进程的运行，并对程序进行监控，来达到的检测出漏洞的目的。计算机软件程序在设计与运行过程中，无论是系统控制还是数据传输，都具有一定的共性，动态检测就是利用这些共性去完成检测的。

动态测试的种类方面，文章[9]中提到目前常见的动态漏洞检测方法包括模糊测试、动态符号执行和动态污点分析三种方法。

模糊测试的基本思想是借由已有数据或规则，通过修改参数等手法，构造针对关键词的大量测试用例，并予以使用。若程序发生出错、崩溃以及非正常情况下的成功等现象，那就说明可能相应的漏洞。模糊测试一般可以分为两种，其一是根据已有正确合法的数据包进行阵对性的参数修改，这一种可以精准地定位错误来源；其二是根据分析得到的数据格式规则进行数据包或者种子的重编写，这一种灵活性强，可以尝试更多的攻击可能。但是模糊测试由于其随机性，在单一使用时，会导致其路径覆盖率低，作为一种查找代码错误的方式，效率较为低下。动态污点分方法是基于用户输入的检测方法。用户输入作为污点源，通过追踪它们在目标程序执行过程中的传播。当追踪到污点源被安全敏感的模块和函数操作时，就说明此处可能存在漏洞。这种方法无法自己构造数据包和参数种子，它主要用于对软件安全敏感操作的检测。

动态符号执行面向程序路径来构造测试用例，它用抽象符号代替程序变量，在程序运行的同时，通过分析输入种子所遇到的判断语句，去记录每一个约束条件，形成约束条件集。之后对各个条件逐个取反生成新种子再次输入测试，看是否存在程序出错的情况。但是由于动态符号执行过于详细，会存在路径爆破问题，过大或者过复杂的问题都会导致执行的工作量难以接受，并且会导致大量的额外非法数据和错误出现，对有 IP 封锁等功能的服务端的测试行为会受到较大的影响。

现阶段还有许多将静态检测与静态检测结合的方法，如蔡军在自己的文中就提到了通过对二进制文件先进行危险函数地址搜索，再以此为指导进行动态符号执行，借此提高效率[1]，但是利用静态检测方法就必须获得项目源码，并且对大项目的适应度会显著降低。

2.1.2 动态监测的常用流程

动态测试的流程方面，Pellegrino等人在文章中[13]根据OWASP的测试指南[14]对测试黑盒环境下的逻辑漏洞动态黑盒测试的四个步骤，包括文档的研究和应用的了解、对预期的工作流和数据流等相关信息的准备、设计测试用例、执行测试并验证结果，提出了一套通过单一的黑箱工具来自动完成上述内容。其步骤主要包括模型推理、行为模式推理、测试用力生成以及测试案例的执行。

模型推理包括资源抽象、资源集约化。资源抽象也即创建资源的综合体，从每个抽象资源中抽取元素，对应于所有可能出现在可用链接中的参数。检查元素的值可以得出与元素本身对应的类型。资源集约化是对请求获得的资源进行分组、整合以及分析拆分。首先，通过异步请求来获取资源，整合同步资源；然后，按照相似性和用户对资源进行分组；之后，若资源为命令而非只携带一个值，就进行拆分。行为模式推理包括数据流模式、工作流程模式。推理中追踪模式主要专注单子节点、多步骤操作以及追踪航点三个模式。单子节点是不会二次访问的节点模式，多步骤操作是以相似流程进行多节点的顺序访问，追踪航点则是用户交互的关键节点。模型模式是根据导航图允许的行动序列模型，包含可重复操作和描述运行轨迹节点的模型航点。测试用例的生成模式包括可重复的单子多重执行、打破多步骤操作、打破服务器的传播链以及绕开交互点。

### 网络应用程序的业务逻辑检测方法

为了能够更加高效地对网络应用服务进行黑盒检测，就需要对其业务逻辑进行检测。Mitra 在曾在文章[15]中提到过一种对网络应用服务进行业务逻辑检测的方法。

业务逻辑检测主要分为两部分。第一步是提取用户导航图。在对原始输入数据进行处理后，仅保留正常访问的GET和POST的相关请求和响应。接着，从预处理的请求和相应中筛选出主要请求和相应的响应，并对末尾请求和响应的主要性进行讨论。对得到的主要请求、响应进行聚类，按照用户行为、HTML结构、HTML结构的子集关系、URL是否为搜索关键词类型、页面是否存在评论行为等相关标准进行分类。聚类过程主要包括以下三个步骤：一、提取页面的属性向量；二、识别类似页面；三、对网络应用程序页面进行聚类；最后便是完成提取用户导航，以图的形式进行保存。第二步是从图中识别应用的业务流程。

### 基于状态机的动态监测方法

网络应用程序的业务逻辑检测还可以通过状态机的形式进行，同时基于它实现动态检测。

在Deepa的文章[15]中提到了一种基于状态机的动态检测方式。在该建模中，网络应用被建模成为了建模成为具有六元组的注解FSM。其中六元组包括了表示应用程序具有URL的页面DOM结构的有限状态集、有限的输入输出集合、提供附加条件的注释（以描述状态过渡的用户、http参数、会话变量三元组）、有限状态集的过渡函数、初始状态及最终状态。对网络应用FSM建模的分析可用于推断参数、角色与控制流的约束条件。参数约束一般包括数据类型、范围约束及长度约束。工作流限制也即对关键页面的搜索，便于揭示缺乏会话变量和CSRF令牌验证而无法维持顺序的页面，关键页面可通过网络应用程序建模为一个有向图来识别。在深度搜索下，可以找到去除后会造成无法访问页面的关键页面，对关键页面的跳过一研究是否存在关键页面绕过的漏洞。

基于上述FSM建模和约束条件的发掘我们可以得到相应描述网络应用行为的状态转换图，也即DetLogic[15]。其运行分为三个阶段，包括学习阶段、攻击生成阶段以及发现阶段。学习阶段是推断网络应用的预期行为，以模型生成器来完成该阶段的任务；攻击生成阶段则是按照修改与附加参数、跳过关键页面和越权与换权访问三种方式生成攻击向量；发现则是通过分析攻击反馈和数据库漏洞案例相似度来得到漏洞结果。

### 支付常用流程

现行主流的支付流程包括支付宝、微信以及其他平台提供[10]的支付方式，下面对部分的支付流程进行简单的叙述：

* + 1. 支付宝的支付流程：

根据支付宝开发文档[2]所述，连接支付宝的支付流程如下：

* + - 1. 用户向商家提交订单申请，商家生成订单发给商户服务端，商户服务端返回订单信息；
      2. 商家发出支付请求，支付宝服务端完成支付后返回向商家同步支付消息；
      3. 商家客户端向服务端发送支付结果后验证签名，验签、分析支付结果，完成后返回支付状态，完成支付流程；

具体的支付流程见下页图 1。

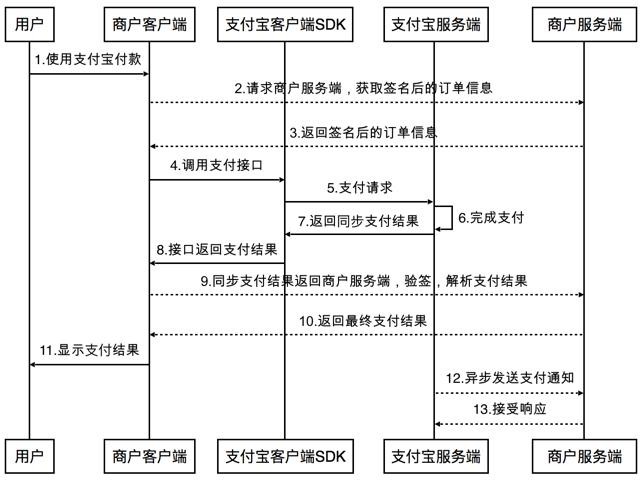


图-1 支付宝支付流程图

* + 1. 微信的支付流程：

根据微信开发文档[3]所述，连接支付宝的支付流程如下：

* + - 1. 前端客户端根据用户操作生成相应订单，由商家后台提交至微信支付系统服务端（下称服务端），在后端生成该交易后返回支付链接；
      2. 用户利用步骤一中的链接向服务端提供链接，服务端验证链接正确性后返回支付权限；
      3. 用户利用授权输入密码进行支付，验证交易正确性，进入并行支付完成阶段，并返回支付

成信息；

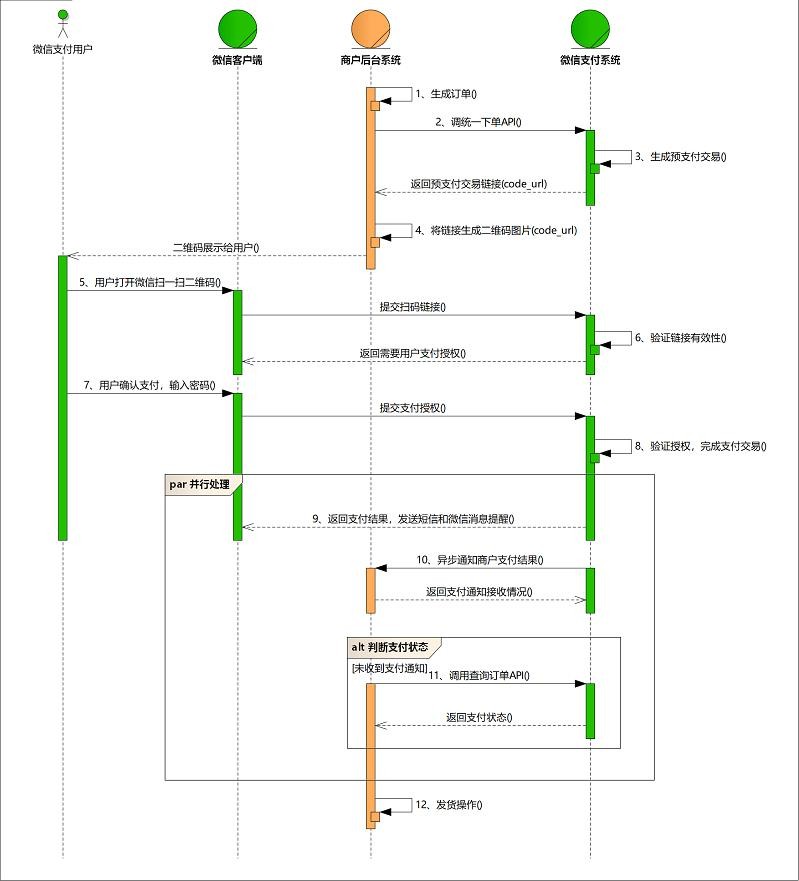
* + - 1. 向商家推送支付结果，若商家未收到则主动拉去订单支付状况；具体支付流程见下页图 2。

图-2 微信支付流程图

* + 1. 其他常见的简单支付流程介绍

根据网上常见的第三方支付流程，可以整理出以下流程：

* + - 1. 由用户提交订单信息，服务器生成订单并返回相关参数；
      2. 用户提交付款验证信息，服务端验证后返回成功信息；
      3. 服务端向商家端推送或商家主动向服务端拉取订单支付状态，完成支付流程；相应流程可下页见图 3。

除此之外，还有一些网站会使用本站的用户电子积分进行买卖，失去了第三方的确认流程，可能会产生更加简单的支付流程，产生一些额外的漏洞。

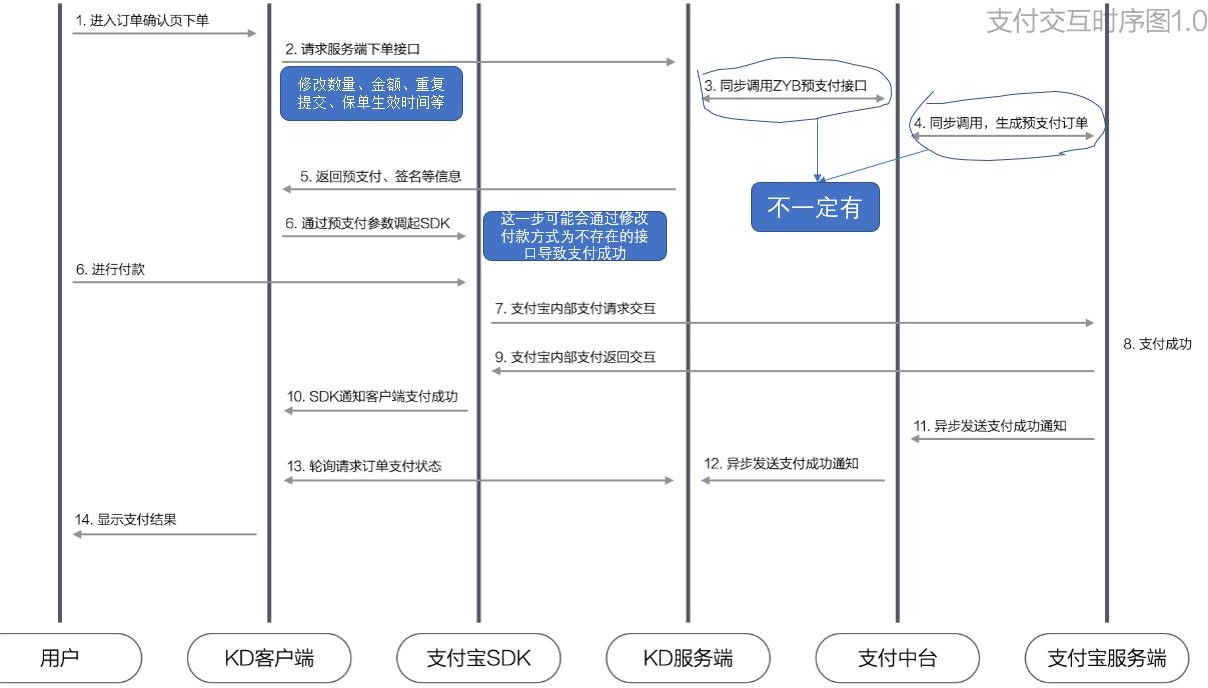


图-3 一般软件支付流程图（及与支付宝的区别）

### 支付常用漏洞总结

根据部分博客和网站的手工支付逻辑漏洞挖掘经验[4]，下文将总结现行常见的支付平台的逻辑漏洞：

1. 修改支付价格、数量：

在完整的支付流程中，订购、确认信息、付款时的价格如果没有进行确认，就可以对价格或者商品数量进行修改，完成价格并不符的支付，其中要考虑有一些平台对数据进行了一些基本的阈值设置，所以对以上数据的修改测试要在一定范围内进行，防止因此漏掉相应的逻辑漏洞，而还有一些数据会因为数据溢出等未知原因在较大数时产生异常支付情况。

1. 修改支付状态：

在一些情况下通过篡改报文中的支付状况，让商家或者服务器端误认为订单已经完成支付，从而跳过支付步骤。

1. 修改其他属性：

由于许多网站存在支付时的附加属性，比如积分抵扣、折扣券，为了同时完成折扣的相应修改，所以订单支付中会附带相关信息回送，如果对校对不严的相关值进行修改，就可以进行一定

的获利。

1. 支付接口漏洞：

考虑到现在不止存在微信支付宝等第三方平台提供支付服务，存在大量设计不当的支付接口，在阻止一些报文的收发后，支付流程会正常运作，达到跳过支付等目的。

1. 重放漏洞：

有一些支付操作后，收到的订单完成包没有进行核实与检测，可能导致重复使用完成支付后的报文就可以让服务端以为订单又被执行了，从而免费获取订单中的商品。

1. 多重替换支付：

有些支付平台的单笔订单的审查较为完整，对应的验证信息可以保证订单支付完成。通过将成功支付的订单 A 信息，复制到订单 B 的报文中，来间接完成订单 B，实现用订单 A 的低价格来完成其他高价订单的支付。在此基础上，如果对商品进行了绑定验证，可以通过利用一些试用、优惠活动的相关报文信息，重用到眼下的订单中，造成低价甚至免费完成目前订单的状况。

1. 重用额外信息：

部分订单中可用账户内的某些抵用券进行购买，再未完成的支付的情况下，这些券可能并未被扣除，便可以进行复用。

1. 越权使用用户信息：

用户信息验证不完善，则会导致用户通过篡改报文中的 user\_id 等相关信息让他人代付订单。

# 检测软件设计

### 检测软件概要设计

由于许多支付平台软件的体量庞大，所以不适合使用静态或动态结合静态的检测方式，因此本实验想要设计一个基于动态检测的实验。同时，为了帮助软件公司在不影响检测人员的前提下进行高效的漏洞挖掘，因此最好不要使用动态符号执行和动态污点测试。基于上述内容，本文将实现一个结合FSM的网络应用行为检测的动态检测软件。

# 实验结果

# 参考文献

1. 蔡军,邹鹏,熊达鹏,何骏.结合静态分析与动态符号执行的软件漏洞检测方法[J].计算机工程与科学,2016,38(12):2536-2541.
2. 微信支付开发文档.Native 支付开发指引 [EB/OL](2021.1.15)[2021.11.5]https://pay.weixin.qq.com/wiki/doc/api/native.php?chapt er=6\_5
3. 支付宝文档中心网页&移动应用.服务端接入流程

[EB/OL](2021-11-17)[2021.11.20]https://opendocs.alipay.com/open/203/106493

1. 交易支付逻辑漏洞总结[OL].[HTTP://www.secpulse.com/](http://www.secpulse.com/)，2016.12
2. 李建军.基于逻辑的网络安全技术研究[D].郑州：解放军信息工程大学，2012
3. 姜洋.渗透测试关键技术研究[D].西安：西安电子科技大学，2014
4. 裴兰珍.网络安全漏洞渗透测试框架综述[J].电子信息对抗技术，2016，31(02)：10-13
5. 李永华,窦春轶.谈计算机安全漏洞动态检测的原理方法与实践[J].数字技术与应用,2010(07):153-154.DOI:10.19695/j.cnki.cn12-1369.2010.07.099.
6. 蔡荣文.基于动态检测技术的软件设计安全漏洞查找方法[J].山东农业大学学报(自然科学版),2019,50(05):873-876.
7. 罗成,武玥.iOS 应用内支付安全分析[J].电信网技术,2013(11):20-22.
8. 何博远.逻辑漏洞检测与软件行为分析关键技术研究[D].浙江大学,2018.
9. 冯丹.Web 应用业务逻辑漏洞检测技术研究[D].浙江工商大学,2017.
10. G. Pellegrino, D. Balzarott. 2014. Toward Black-Box Detection of Logic Flaws in Web Applications. NDSS 14, SanDiego, yCA, USA, 23-26.
11. The OWASP Foundation, “OWASP Testing Guide,”December 2008. [Online]. Available: https://www.owasp.org/index.php/OWASP Testing Project
12. Mitra Alidoosti. Alireza Nowroozi. Ahmad Nickabadi. July 2019. BLProM:A Black-Box Approach for Detecting Business-Layer ……
13. DetLogic: A black-box approach for detecting logic vulnerabilities in web applications