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

**1 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。

**2 背景知识总结**

**2.1 动态检测技术现状**

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。目前主流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。

动态污点分析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]。但是通过静态

**2.2 支付常用流程**

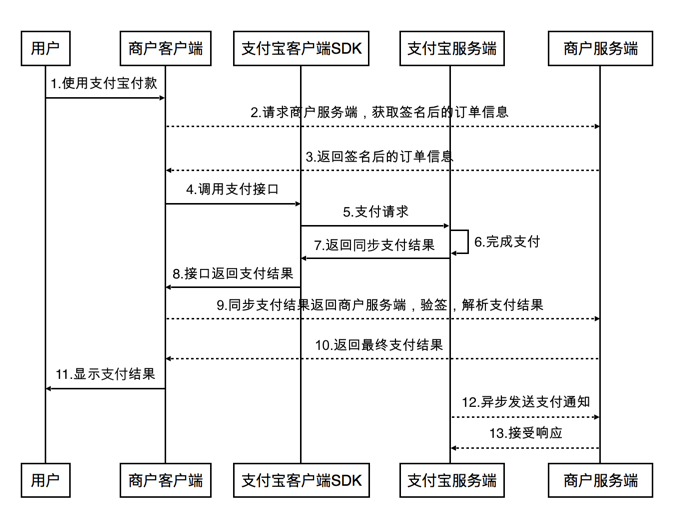
现行主流的支付流程包括支付宝、微信以及其他平台提供[10]的支付方式，下面对部分的支付流程进行简单的叙述：

2.2.1 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

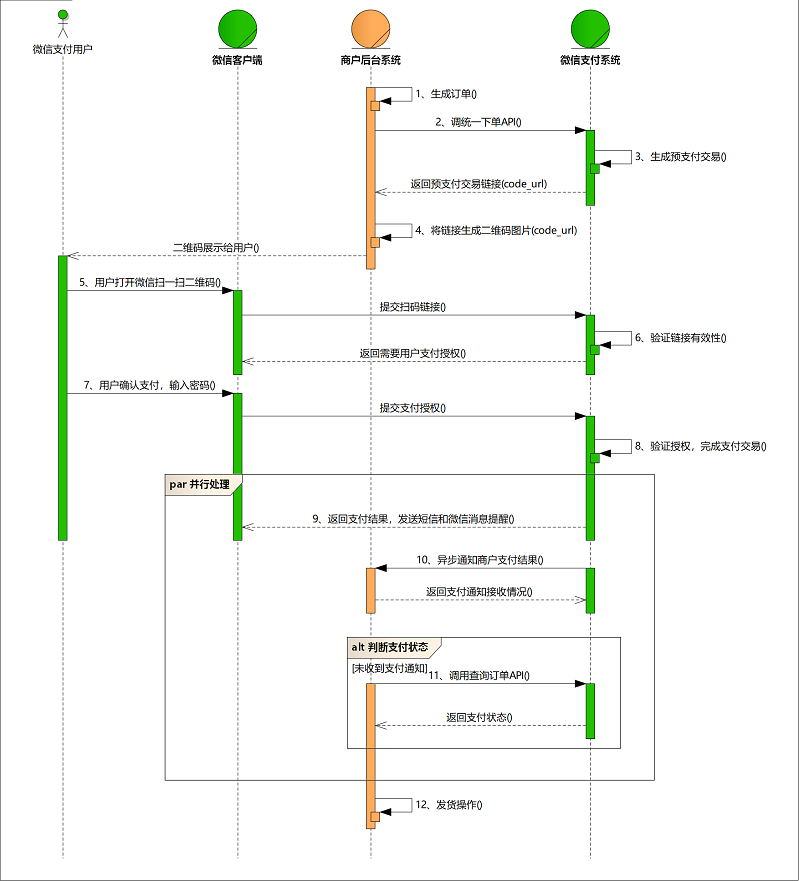
2.2.2 微信的支付流程：

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；

1. 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；
2. 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，并返回支付完成信息；
3. 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

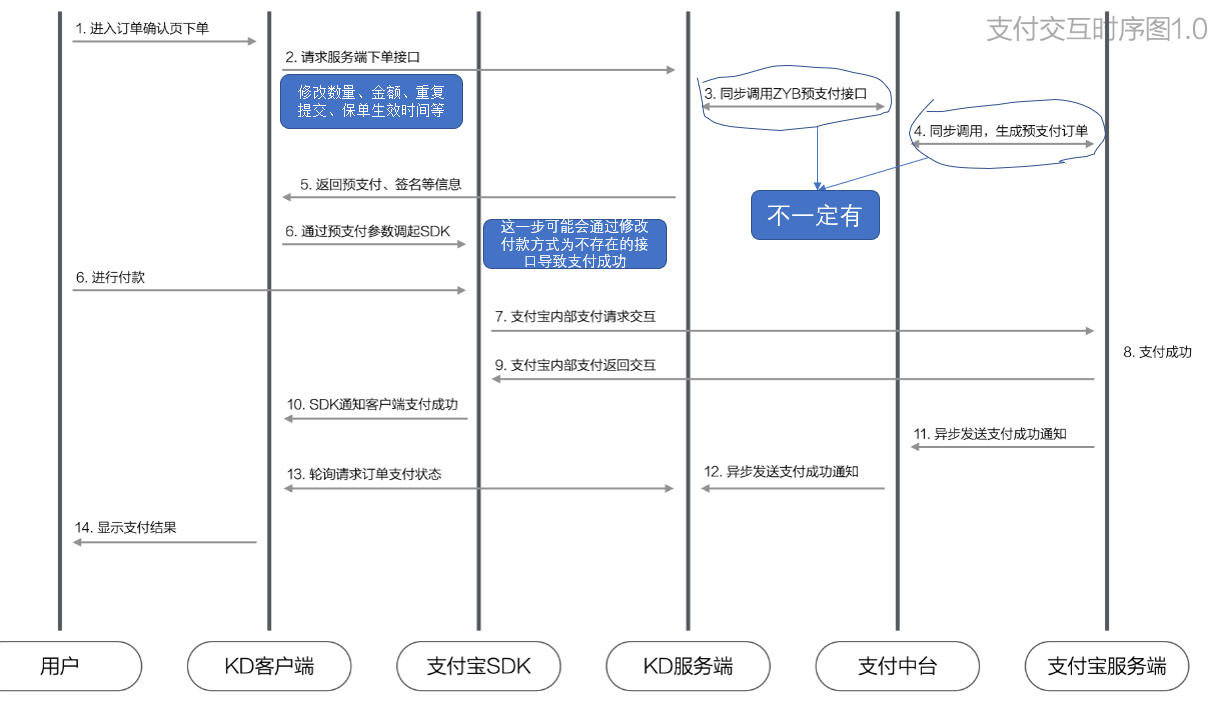
2.2.3 其他常见的简单支付流程介绍

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）

**2.3 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。

**2.3 使用的开源软件简介**

参考已有的渗透软件设计[7]，本实验中为了更好地实现数据包的截获，本实验将使用Burpsuite和W13scan软件作为基础开发。

Burpsuite是一款开源的集成化渗透测试工具，集合了多种渗透测试组件，帮助我们快速完成应用的渗透测试和攻击检测。它的软件本体主要分为9个模块，包括proxy，target，intruder，comparer，repeater，decoder，extendere，sqlmap，csrf，通过代理的方式对目标进行数据包的截取、处理和使用。

W13scan是一款常用于web端静态漏洞检测的开源扫描工具，具有高功能性以及较强的编程灵活性，它可以通过编写连接软件，将其与Burpsuite进行对接。通过Burpsuite截获的报文可以直接发往W13scan中，由编程者编写的插件模块进行处理，进而完成漏洞挖掘和扫描。

**3 检测软件设计**

* 1. **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。同时为了减小对检测人员的影响，让报文能够更有指向性地完成检测，可以考虑从后台先获取我们可能需要重点关注的字段进行检测，同时也可以避免考虑如何对返回包进行检测，确认漏洞攻击成功与否。因为可以大致确认需要测试的字段，所以本文将选择动态模糊测试作为实验的测试基础。

根据常见支付逻辑漏洞的收集情况，本实验将检测的内容分为两个部分：

1. 直接修改单笔订单相关信息的实验，通过直接修改与订单直接相关、和用户电子财产直接相关的数据以及和用户确认信息相关的部分来进行检测，观察是否存在修改支付价格及数量、修改支付状态、修改其他属性以及越权使用用户信息等漏洞；
2. 通过使用多组已有的合法支付报文去覆盖和篡改待测报文的部分内容的方式，尝试对重放漏洞、多重替换支付以及重用额外信息等相关漏洞；

同时，为了应对没有后台权限的情况，本软件还是会集成会报文分析模块，通过分析来去数据报文，查找关键词，进行可选择地属性替换，也即完全的模糊测试，对软件进行漏洞挖掘，实现软件的灵活性。

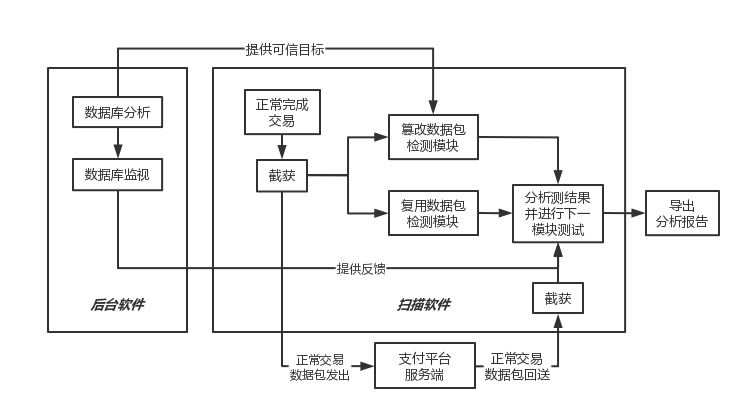
* 1. **检测程序具体设计**

根据程序概要设计和常见框架设计[8]，本测试软件将分后端监视软件和前端检测软件。

后台程序通过编写python脚本分析并监控后台数据库，第一要得出具体的攻击目标，第二要同时监控相应的数据库变化，来对前端的扫描行为做出反应和总结。

扫描程序为了能更有效地实现数据包的截获，因此本事将使用开源的Burpsuite以及W13scan软件，并通过二者的联动，实现对目标数据包的截获和使用。通过截获正常且完整的订单报文，我们可以同时得到攻击样本和分析样本，软件会将这些报文分别送达篡改数据包检测模块和复用数据包检测模块。篡改数据包检测模块会通过是否存在后台程序，来选择性地使用后台程序提供地攻击目标作为主要参考，再对报文关键字进行分析，针对这些关键字进行各个区段的模糊测试，并等待返回结果；复用数据包检测模块会将已有的报文按照不同测试需求设定复用到以此新的未发生的交易报文中并发出，等待结果。待结果从会送数据包或者数据库后台软件返回时，分析器得出各个数据包的结论，从而给出相应的漏洞检测报告。

具体的程序流程图4如下：

图-4 检测程序概要流程图

**4 实验结果**

由于本实验属于开题阶段目前仅包括重放测试和部分数据包篡改，且未完成后台分析监视部分。由于本实验并未完成数据库检测部分，现在仅对服务器返回包检测。

* 1. **软件正确性测试**

为了测试软件的正确性，我们对Burpsuite软件给出的逻辑漏洞靶场进行测试。因为不存在后台分析部分，这里先对重放攻击进行测试。

测试过程图5与结果图6如下：

图-5 程序正确性测试界面

图-6 程序正确性测试报告

* 1. 软件平台测试

由于没有找到合适的漏洞网站进行实战测试，本文为了能更好地对在线支付环境进行测试，使用了github上选择的fastadmin-pay开源支付插件作为测试对象进行测试。通过对fastadmin-pay的源码进行分析，对订单金额、订单数量、订单用户等部分的检测进行了相应的修改，并在每次修改后进行漏洞扫描，实验结果如下表表1。

|  |  |  |
| --- | --- | --- |
| 被去除的检测模块类型 | 检测到的漏洞 | 是否检测到对应漏洞 |
| 订单金额 | 订单金额漏洞 | 是 |
| 订单数量 | 订单数量漏洞 | 是 |
| 订单用户 | 重放漏洞、订单用户漏洞 | 是 |
| 订单积分 | 订单额外信息漏洞 | 是 |
| 订单优惠券 | 无 | 否 |

表 -1 fastadmin-pay支付插件上各漏洞的检测情况

实验结果表明在金额、数量、用户信息的漏洞检测中，该软件已经可以初步完成任务。但是在部分订单额外信息漏洞上，软件性能表现不佳，未能将漏洞找出，也可能存在软件中有本实验里未发觉的验证逻辑消除了漏洞的存在。

**5 结论**

本文总结并分析了主流的支付平台的支付流程，并且收集了手工挖掘主要针对的支付平台逻辑漏洞脆弱点，从而明确了本文要检测的具体目标。同时，本文对现有动态测试软件进行了总结，按照减少对测试人员影响的要求选择模糊测试作为测试的主要动态测试技术。本文还设计了前后端联合实现关键词检测和结果检测，借此方法实现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。

由于本文仍处在开题阶段，程序开发仍在进行，因此仅完成了部分的扫描程序内容并做测试。关于后台扫描得到的关键词与扫描端的关键词匹配和复用报文的使用规则等细节仍需要进一步考虑。

在已完成部分，经过实验发现，由于仍有一些具体逻辑并未细分，或者像优惠券一类存在具体虚拟账户内存在校对的实物财产存在更多的捆绑逻辑校验，因此在数据包直接修改部分仍有许多漏洞没有检测出来，在后期的实验中软件需要进一步地改进。

**6 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?chapter=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
4. 交易支付逻辑漏洞总结[OL].HTTP://www.secpulse.com/，2016.12
5. 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
6. 姜洋.渗透测试关键技术研究[D].西安：西安电子科技大学，2014
7. 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
8. 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.
9. 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.
10. 罗成,武玥.iOS应用内支付安全分析[J].电信网技术,2013(11):20-22.
11. 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.
12. 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.