

本科生毕业设计(论文)

基于二次汇编的ROP攻击防御技术研究与实现

Research and implementation of ROP attack defense based reassemble

|  |  |
| --- | --- |
| 学 院： | 计算机学院 |
| 专 业： | 软件工程 |
| 学生姓名： | 李博 |
| 学 号： | 1120162015 |
| 指导教师： | 孙建伟 |

2020 年 月 日

原创性声明

本人郑重声明：所呈交的毕业设计（论文），是本人在指导老师的指导下独立进行研究所取得的成果。除文中已经注明引用的内容外，本文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。

特此申明。

本人签名： 日 期： 年 月 日

关于使用授权的声明

本人完全了解北京理工大学有关保管、使用毕业设计（论文）的规定，其中包括：①学校有权保管、并向有关部门送交本毕业设计（论文）的原件与复印件；②学校可以采用影印、缩印或其它复制手段复制并保存本毕业设计（论文）；③学校可允许本毕业设计（论文）被查阅或借阅；④学校可以学术交流为目的,复制赠送和交换本毕业设计（论文）；⑤学校可以公布本毕业设计（论文）的全部或部分内容。

本人签名： 日 期： 年 月 日

指导老师签名： 日 期： 年 月 日

基于二次汇编的ROP攻击防御技术研究与实现

摘　要

随着计算机和软件技术的不断发展，使用软件已经是人们日常生活中必不可少的一部分。与此同时，也有越来越多的软件漏洞被挖掘出来，这些漏洞一旦被黑客利用，将极大地威胁系统的安全。面向返回编程(Return-oriented programming)是一种功能强大且用途广泛的代码重用攻击，攻击者可以基于现有的指令片段(gadget)引导程序控制流执行非预期功能，由此引发一系列的安全事件。于是使用有效的ROP攻击防御措施对于系统安全的保护有重要意义和价值。

本文对现有的ROP攻击防御技术进行讨论和分析，提出了一种基于二进制重写技术的ROP攻击防御方法，对Linux的32位和64位可执行文件进行保护。当前的ROP攻击防御方法一部分需要程序的源码，在编译器层面进行修改，然而如今的很多商用软件并不开源；另一部分使用动态监控的方法，会造成很大的性能开销。本文通过使用二次汇编框架Ramblr，可以有效的避免二进制文件没有源码的问题，在此基础上提出了指令替换，无效指令填充和自由跳转保护三种方案。通过对含有0xc2这样可能被解析为retn指令的指令进行等效指令替换或无效指令分隔，减少gadget数量，达到缓解ROP攻击的效果；以及在直接跳转和间接跳转前进行数据校验，保护控制流的完整性，防止攻击者直接调用跳转指令进行控制流转移。

实验部分，通过ROP攻击防御测试，上述方案可以有效地减少gadget数量，并且能够防御基于栈溢出的ROP攻击。结果表明，balabala。

**关键词：**软件安全、面向返回编程、二次汇编、Ramblr

The Subject of Undergraduate Graduation Project (Thesis) of Beijing Institute of Technology

Abstract

Key Words: ROP attack defense, reassemble, Ramblr

目录

[摘　要 - 3 -](#_Toc36888971)

[Abstract - 4 -](#_Toc36888972)

[第1章 绪论 - 6 -](#_Toc36888973)

[1.1 研究背景和意义 - 6 -](#_Toc36888974)

[1.2 国内外研究现状 - 7 -](#_Toc36888975)

[1.2.1 ROP攻击防御技术 - 7 -](#_Toc36888976)

[1.2.2 二次汇编技术 - 7 -](#_Toc36888977)

[1.3 论文主要研究内容 - 7 -](#_Toc36888978)

[1.4 论文结构 - 8 -](#_Toc36888979)

[第2章 相关理论知识 - 9 -](#_Toc36888980)

[第3章 基于二次汇编的ROP攻击防御方法设计 - 10 -](#_Toc36888981)

[第4章 基于二次汇编的ROP攻击防御实现 - 11 -](#_Toc36888982)

[第5章 测试及结果分析 - 12 -](#_Toc36888983)

[结　论 - 13 -](#_Toc36888984)

[总结 - 13 -](#_Toc36888985)

[展望 - 13 -](#_Toc36888986)

[致谢 - 14 -](#_Toc36888987)

[参考文献 - 15 -](#_Toc36888988)

1. 绪论
   1. 研究背景和意义

随着网络技术的快速发展，计算机正不断受到越来越复杂的攻击者的威胁。在网络中，攻击者发送恶意制作的数据包，这些数据包利用软件错误来获得未经授权的控制。防止此类错误存在的研究一直在进行中，但是到目前为止，仍未能为可利用的安全漏洞问题提供一个周全的方法。但是，针对攻击本身的不同方面的解决方案已经取得了一些成功，这导致了恶意攻击者和安全研究人员之间的“军备竞赛”。

最早的攻击技术之一是代码注入攻击，攻击者将精心编写的代码写入有漏洞的程序的内存中，然后利用一个漏洞将控制流重定向至代码所在的内存，从而执行代码。攻击者注入的代码能够损害数据库的完整性和安全性，还能够窃取数据来绕过访问和身份验证控制。针对这一问题，早在1961年，Burroughs 5000推出时就为可执行空间保护提供了硬件支持，在实现带标记的体系架构时，内存中的每个字节都带有一个关联的隐藏标记位，用于指定其是代码或数据，确保了内存可写或可执行。Microsoft公司在2004年将这种技术首次应用在Windows XP Service Pack 2，可执行的空间保护在Windows中也称为数据执行保护（Data Execution Prevention，DEP），开启此功能后，计算机将拒绝执行位于用户可写内存区域中的任何代码，从而防止攻击者将代码写入堆栈并通过返回地址覆盖跳转到堆栈。但是，早期的数据执行保护依旧不是完美的，攻击者可以使用代码重用攻击，例如return-into-libc技术，通过栈溢出覆盖返回地址将控制流转移到libc库中现有的代码片段，调用system函数或mprotect函数创建可写的可执行内存区域来绕过DEP。于是Windows系统又添加了地址空间布局随机化（Address Space Layout Randomization, ASLR）技术，用于将虚拟地址随机分配给正在运行的程序中的代码和数据，防止攻击者利用事先从程序中获取的静态地址来执行代码。

面向返回编程（Return-Oriented Programming, ROP）是一种基于代码重用的攻击，它也是栈溢出的高级形式，最初由Shacham在2007年提出[1]。通过这种方法攻击者可以劫持程序的控制流，并执行计算机内存中已经存在的指令片段（gadget），从而绕过DEP。像栈溢出一样，面向返回编程可以滥用缓冲区溢出漏洞来执行恶意指令，而不受安全措施DEP的阻碍。



图 1-1 国家信息安全漏洞库-漏洞新增数量统计图

图1-1是国家信息安全漏洞库从2019年9月至2020年2月关于漏洞新增数量的统计图，近六个月来平均每月漏洞达到1432个，2020年2月份采集安全漏洞共1246个，其中缓冲区错误类型的漏洞最多，数量为150个，占比12.04%，并且在超危漏洞中缓冲区错误类型的漏洞也有20个，数量最多。这些漏洞可以为ROP攻击提供很好的利用条件。于是，及时的检测并防御ROP攻击就显得极为重要。基于此，本文提出基于二次汇编的ROP攻击防御技术，通过二次汇编技术可以对可执行文件进行反汇编生成汇编文件，并对汇编指令进行混淆和插桩，然后将汇编文件重编译成新的可执行文件。此过程不需要程序的源代码，并且可以在指令级别进行处理，在不对程序进行外部监控的情况下自动化的完成ROP攻击防御，减少了性能开销。

* 1. 国内外研究现状

1.2.1 ROP攻击防御技术

ROP攻击通过将目标程序中现有的指令序列链接在一起，可以使远程攻击者执行具有图灵完备的计算，而无需注入任何恶意代码。正是由于其巨大的威胁，近年来产生了许多针对ROP攻击的防御方法。

一些安全研究人员在程序源代码的基础上进行ROP攻击的防御。

Kaan Onarlioglu等人[2]提出了基于减少gadget和保护自由跳转分支的防御方法，通过修改编译器使源代码进行编译生成中间代码时对中间代码进行处理，包括指令替换、无效指令（nop指令）以及自由跳转保护，以达到减少gadget数量和保护跳转指令不被攻击者滥用的目的。

Tyler Bletsch等人[3]提出了基于控制流锁定的防御方法，通过一个大小为1比特的变量k来标记此时状态是解锁、间接调用、来自间接调用函数的返回以及来自直接调用的函数返回，从而防止跳转指令被攻击者滥用。

以上方法的实现需要重新编译程序，意味着每个使用者不得不使用新的程序来增加安全性。然而重新构建一个新的程序往往是一个巨大的工程，并且如果源代码丢失，那么程序的安全性就更无法得到保障。于是一些安全研究人员在不修改程序的基础上通过外部监控的方式进行ROP攻击的防御。

Lucas Davi等人[4]使用二进制指令框架Pin来监控栈内容达到防御ROP攻击的目的。通过复制一个程序栈，在程序动态运行时，在处理器执行指令之前拦截指令并检查指令类型。若为call指令，则将返回地址复制一份压进栈，若为ret指令，则比较复制的栈与程序栈的栈顶内容是否相同，从而判断该跳转指令是否被攻击者使用。

Vasilis Pappas[5]等人提出了一种基于异常控制流转移的动态检测方法。利用处理器提供的上次分支记录技术（Last Branch Recording，LBR），可以获取上一次执行分支的记录，从而在系统调用返回时，判断之前是由call指令还是ret指令跳转到系统调用的，若是call指令则是善意的系统调用，否则是恶意的系统调用。

Ivan Fratrić[6]等人提出了六种不同的检测方法来判断一个函数调用是否是恶意的。包括检验栈指针是否在正确的边界内、检查栈上是否有特定函数的入口地址、验证返回地址是否有效（可执行并且前面有call指令）、验证调用栈是否有效、模拟程序执行流程以及特定函数检验（VirtaulProtect和LoadLibrary函数）。

Yueqiang Cheng等人[7]使用LBR技术和预运行的方法。在预运行阶段导出所有指令的偏移、类型和对齐方式，并收集程序和共享库中可能的gadget，以此构建一个数据库;在运行阶段通过滑动窗口机制限制可利用的指令数量，并通过LBR技术识别先前执行的指令和传入的指令是否是ROP攻击的gadget。

以上的方法大部分依赖于ret-call指令之间的联系，并通过CPU的LBR技术在程序动态运行时获取控制流的转移信息，从而判断是否受到ROP攻击。然而外部监控的方法会带来很大的性能开销，并且Nicholas Carlini等人[8]已经对kBouncer和ROPecker两种防御方法提出了破解的思路及实现方法。

1.2.2 二次汇编技术

* 1. 论文主要研究内容
  2. 论文结构

第2章 相关理论知识

# 第3章 基于二次汇编的ROP攻击防御方法设计

# 第4章 基于二次汇编的ROP攻击防御实现

# 第5章 测试及结果分析

# 

结　论

## 总结

## 展望

# 致谢

# 参考文献

1. H. Shacham, “The Geometry of Innocent Flesh on the Bone: Returninto-libc Without Function Calls (on the x86),” in Proceedings of the 14th ACM Conference on Computer and Communications Security, 2007.
2. Onarlioglu K, Bilge L, Lanzi A, et al. G-Free: defeating return-oriented programming through gadget-less binaries[C]//Proceedings of the 26th Annual Computer Security Applications Conference. 2010: 49-58.
3. Bletsch T, Jiang X, Freeh V. Mitigating code-reuse attacks with control-flow locking[C]//Proceedings of the 27th Annual Computer Security Applications Conference. 2011: 353-362.
4. Davi L, Sadeghi A R, Winandy M. ROPdefender: A detection tool to defend against return-oriented programming attacks[C]//Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security. 2011: 40-51.
5. Pappas V. kBouncer: Efficient and transparent ROP mitigation[J]. Apr, 2012, 1: 1-2.
6. Fratrić I. ROPGuard: Runtime prevention of return-oriented programming attacks[R]. Technical report, 2012.
7. Cheng, Yueqiang, et al. "ROPecker: A generic and practical approach for defending against ROP attack." (2014): 1.
8. Carlini, Nicholas, and David Wagner. "{ROP} is Still Dangerous: Breaking Modern Defenses." *23rd {USENIX} Security Symposium ({USENIX} Security 14)*. 2014.