软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(9):2799−2820 [doi: 10.13328/j.cnki.jos.005597]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

移动目标防御的攻击面动态转移技术研究综述*

周余阳, 程 光, 郭春生, 戴 冕

[1](东南大学 计算机科学与工程学院,江苏 南京 211189)

[2](东南大学 网络空间安全学院,江苏 南京 211189)

[3](教育部计算机网络和信息集成重点实验室(东南大学),江苏 南京 211189)

通讯作者: 程光, E-mail: gcheng@njnet.edu.cn

摘 要: 移动目标防御作为一种动态、主动的防御技术,能够通过不断转移攻击面,减少系统的静态性、同构性和确定性,以此挫败攻击者的攻击.随着网络攻击手段的不断发展和变化,深入研究移动目标防御对网络空间安全具有重要意义,而攻击面的动态转移技术作为移动目标防御领域的重点问题,一直受到研究人员的广泛关注.利用攻击面动态转移技术所具有的不确定性、动态性和随机性等优势,实现信息系统的动态防御,可以有效克服传统防御手段的确定性、静态性和同构性的不足.首先梳理了攻击面的基本概念,并具体阐释了攻击面以及攻击面转移的形式化定义;其次,分析了攻击面 4 个层次的动态转移技术——数据攻击面、软件攻击面、网络攻击面和平台攻击面,并对不同的动态转移技术进行分析和比较,分别指出它们的优点和缺陷;最后,还从多层次攻击面动态转移技术的融合、攻击面动态转移的综合评估方法、基于感知的攻击面动态转移方法、基于三方博弈模型的攻击面转移决策等方面讨论了未来移动目标防御中攻击面动态转移可能的研究方向.

关键词: 移动目标防御;攻击面;网络空间安全;动态转移技术

中图法分类号: TP393

中文引用格式: 周余阳,程光,郭春生,戴冕.移动目标防御的攻击面动态转移技术研究综述.软件学报,2018,29(9):2799−2820. http://www.jos.org.cn/1000-9825/5597.htm

英文引用格式: Zhou YY, Cheng G, Guo CS, Dai M. Survey on attack surface dynamic transfer technology based on moving target defense. Ruan Jian Xue Bao/Journal of Software, 2018,29(9):2799−2820 (in Chinese). http://www.jos.org.cn/1000-9825/5597.htm

Survey on Attack Surface Dynamic Transfer Technology Based on Moving Target Defense

ZHOU Yu-Yang, CHENG Guang, GUO Chun-Sheng, DAI Mian

[1](School of Computer Science and Engineering, Southeast University, Nanjing 211189, China)

[2](School of Cyber Science and Technology, Southeast University, Nanjing 211189, China)

[3](Key Laboratory of Computer Network and Information Integration of Ministry of Education (Southeast University), Nanjing 211189, China)

Abstract: As a dynamic and active defense technology, moving target defense can defeat the attacker's attack by constantly shifting the attack surface and reducing the static, isomorphic and deterministic nature of the system. With the continuous development and changes of network attacks, in-depth study of moving target defense is of great significance to China's cyberspace security. As a key problem in moving target defense field, attack surface dynamic transfer technology has attracted wide attention of researchers. The dynamic transfer technology takes advantage of uncertainty, dynamicity and randomness, can realize dynamic defense of the information system and

* 基金项目: 国家自然科学基金(61602114); 国家重点研发计划(2017YFB0801703)

Foundation item: National Natural Science Foundation of China (61602114); National Key Research and Development Program (2017YFB0801703)

收稿时间: 2018-01-10; 修改时间: 2018-04-01, 2018-04-30; 采用时间: 2018-05-09; jos 在线出版时间: 2018-06-07

CNKI 网络优先出版: 2018-06-07 14:53:43, http://kns.cnki.net/kcms/detail/11.2560.TP.20180607.1453.005.html

effectively overcome the certainty, static and isomorphism of traditional defense. In this paper, the basic concept of the attack surface is first laid out, and the formal definitions of the attack surface and attack surface transfer are explained. Then, the attack surface dynamic transfer technologies are introduced from four aspects, including data attack surface, software attack surface, network attack surface and platform attack face. Furthermore, different dynamic transfer techniques, are analyzed and compared, and their advantages and shortcomings are pointed out. Finally, the future possible research directions of attack surface dynamic transfer technology are discussed with emphasis on the multi-level attack surface dynamic transfer technology integration, comprehensive evaluation method of attack surface dynamic transfer, dynamic transfer method of attack surface based on perception and attack surface transfer decision-making based on the three-party game model.

Key words: moving target defense; attack surface; cyberspace security; dynamic transfer technology

互联网技术自问世以来,首先运用于军事、教育和科研领域,后来又迅速向政治、经济、社会和文化等各个领域渗透.在短短的几十年里,网络将人类从工业时代带入信息时代,并且彻底地改变了人类社会面貌和生活生产方式.

互联网一方面给人们带来了诸多的便利和好处,而另一方面,信息化的迅猛发展也带来了诸多网络安全威胁,蠕虫、木马、拒绝服务攻击以及近年来兴起的高级持续性威胁(APT)攻击等,让人们不得不去面对互联网安全问题的严峻现实,而近年来爆发的各类安全事件(Stuxnet 病毒、“棱镜门”事件、heartbleed 漏洞、WannaCry 勒索病毒等),也使得人们感受到各行各业中潜伏的网络安全威胁无处不在.

虽然现有的防御方法已经发展得相当成熟,但是威胁的未知性和持续性使得互联网安全面临着易攻难守的严峻挑战[1].网络系统体系结构上由于设计的缺陷或人为的疏漏,存在诸多固有的安全威胁.同时,网络组成的确定性和网络结构的静态性为网络攻击提供了依存环境,使得攻击者相比较于防御者存在着时间和信息不对称的优势;此外,网络组成要素的单一性和同构性使得攻击者具备成本优势,一旦攻击者成功攻击了某系统,其可以轻松利用很小的攻击开销将攻击放大、扩散.因此,现有的防御方法难以有效地应对愈加复杂的网络入侵与攻击,防御者自始至终处在被动的劣势地位.

为了改变这种攻防不对称的劣势局面,移动目标防御(moving target defense,简称 MTD)便是为了“改变游戏规则”而提出的防御研究方向[2],其核心思想是创建和部署多样的、变化的机制和策略,以此增加攻击者实施攻击的复杂度和攻击开销,限制漏洞被暴露和被利用的机会.通过对目标系统的攻击面实施多层次、动态持续的转移,减少系统的静态性、同构性和确定性,迷惑或误导攻击者,增加攻击者实施攻击的成本和难度,降低入侵系统的成功率直至迫使攻击者放弃攻击,从而提高了目标系统的安全性.

移动目标防御并不是某一种具体的防御方法,而是一种设计指导思想[3].目前,在该思想的指导下,世界范围内的许多研究者进行了大量的研究工作,产生了很多攻击面的动态转移技术.攻击面动态转移技术主要通过改变特定系统资源属性或属性对外的呈现信息,使其攻击面发生变化,从而迷惑或误导攻击者,促使攻击者攻击错误目标或丢失攻击目标,改变网络防御被动的局面,以此提高系统的安全性.

已经有部分学者进行了相关技术的分析和研究,但从公开发表的科研论文和资料来看,国内外对攻击面的动态转移技术的相关分析研究还不够全面和深入.2011 年和 2012 年,Jajodia 等人[1,4]编写了两本关于移动目标防御的书籍,但是这两本书按照不同学者的研究工作划分章节进行编排,每位仅对自身的攻击面转移技术进行深入探讨,相互之间关联不足,缺少各个攻击面动态转移技术的分类和比较分析;2013 年,Okhravi 等人[5]对现有的攻击面动态转移技术进行介绍和分析,且有一定的分类,但是该文献更偏向于研究报告,对于技术细节的描述很多,对于转移技术的分析十分简单,且缺乏不同技术的横向对比;2016 年,蔡桂林等人[2]对当前的移动目标防御技术的研究进展进行了介绍,其中,攻击面转移技术部分的分类方式只是按照不同的具体技术进行分类,缺乏统一性.

因此,本文尝试对攻击面的动态转移方法进行全面的归纳和总结,按照数据、软件、网络、平台这 4 类攻击面进行分类,对不同的动态转移技术分析其优势及缺陷并进行横向对比,最后探讨了未来可能的研究方向,为进一步研究作参考.

本文第 1 节介绍攻击面及其动态转移的基本概念与形式化定义.第 2 节按照 4 类攻击面的分类对攻击面的

动态转移技术进行阐述、分析和横向对比,同时也对攻击面动态转移的策略进行单独介绍.第 3 节对未来研究方向进行介绍和展望.第 4 节对全文工作进行总结.

1 攻击面及其转移的基本概念

1.1 攻击面的基本概念

目前,学术界对于攻击面尚无一个明确、统一的定义.Howard 等人[6]将攻击面描述为软件系统遭受攻击的能力,包括 3 个维度:攻击目标和促成因素、通道和协议、访问权限.Manadhata 等人[7,8]根据针对操作系统的许多攻击案例的分析,总结出攻击者可以使用系统函数、通道和系统环境中的数据项对系统进行攻击,并据此将系统攻击面定义为方法、通道和数据三元组的子集.Kurmus 等人[9]在 Linux 系统内核的研究中,将程序中公开的函数和函数直接、间接调用的组合定义为调用图,并在此基础上将攻击面定义为调用图、攻击可调用函数集和阻碍攻击函数集三元组的子集.Peng 等人[10]将攻击面定义为所有可外部访问资源的总和.Foreman 等人[11]、Sun 等人[12]和 Cybenko 等人[13]将网络攻击面定义为攻击者实施攻击、连接到系统、破坏系统的方法和手段.而 Bopche 等人[14]则将其定义为网络配置与攻击者可利用漏洞等网络资源的子集.

综上所述,本文将攻击面定义为系统中可被利用、遭受攻击的系统资源集合,攻击者可通过攻击面实施攻击,达到其窃取系统资源、破坏系统的目的,并参照文献[7]给出了图 1 所示的示意图.

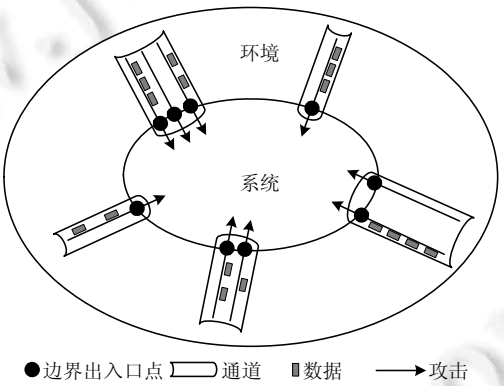

Fig.1 A diagrammatic sketch of the elements on the attack surface

图 1 攻击面的组成示意图

1.2 攻击面动态转移的基本概念

移动目标防御是一种需要防御者不断转移系统攻击面的防御方法.直观来讲,防御者可以通过不断改变攻击面上的资源或者改变各类资源的作用力,以实现攻击面的转移.然而,并非所有的改变都能够转移系统攻击面,防御者需要至少改变所利用的某一类攻击面资源(例如,防御者可以修改 IP 地址、系统开放的端口数量等),或至少改变某一类攻击面资源对攻击面的影响力大小(例如,防御者可以修改执行某些命令所需的权限等),从而才能达到转移攻击面的目的.

在其他条件保持不变的情况下,即在不引入新的脆弱性、带来新安全风险的前提下,若原有攻击所利用的资源已经发生上述改变,则原有攻击将不再有效.如此一来,攻击者就需要更高的攻击成本、花费更多的攻击时间才可能实现原有攻击,或者可能需要额外寻找其他的攻击途径.于是,本文将上述攻击面资源的改变定义为攻击面的动态转移,其示意图如图 2 所示.

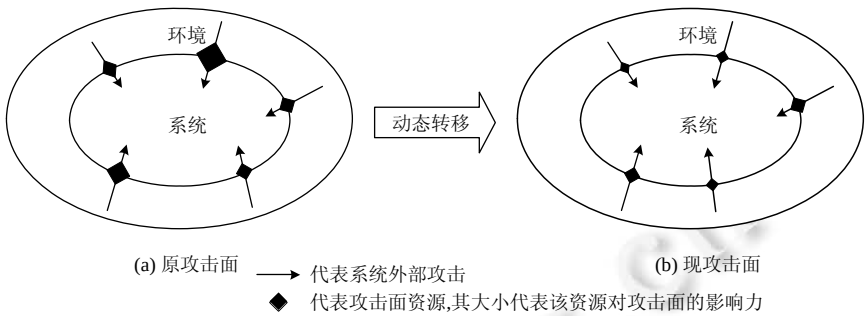

Fig.2 A diagrammatic sketch of the dynamic transfer process of the attack surface

图 2 攻击面动态转移过程的示意图

2 攻击面的动态转移技术

攻击面动态转移技术主要通过改变特定系统资源属性或属性对外的呈现信息,使其攻击面发生变化,从而迷惑或误导攻击者,促使攻击者攻击错误目标或丢失攻击目标,改变网络防御被动的局面,提高系统的安全性.目前,攻击面的动态转移技术主要包括 4 类:基于数据攻击面的动态转移技术、基于软件攻击面的动态转移技术、基于网络攻击面的动态转移技术以及基于平台攻击面的动态转移技术,具体的分类框架如图 3 所示.同时,为使攻击面动态转移技术更具有效性和可用性,学术界对于攻击面动态转移策略的研究也正如火如荼地进行,故这一部分内容也将在本节最后单独介绍.

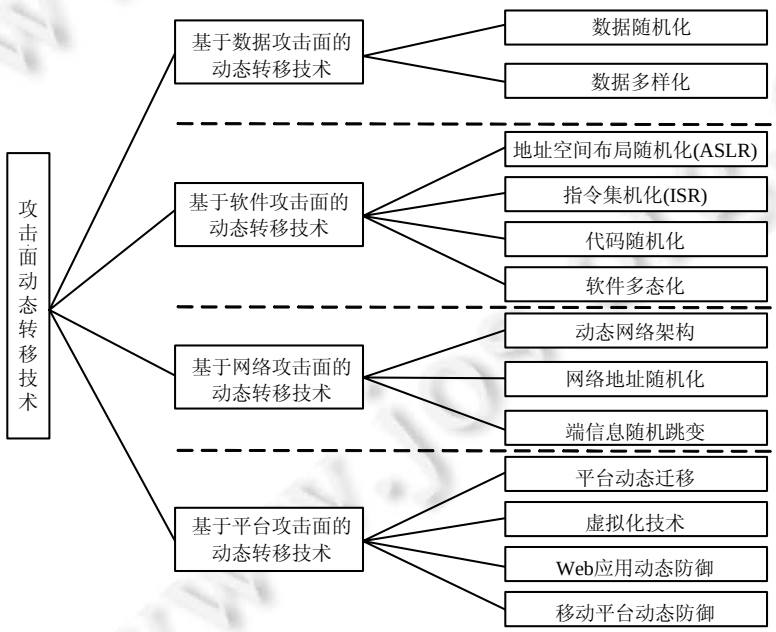

Fig.3 Classification framework of the attack surface dynamic transfer technology

图 3 攻击面动态转移技术的分类框架

2.1 基于数据攻击面的动态转移技术

数据作为系统中最基本的组成成分,也是攻击者优先进行攻击的攻击目标,是攻击者对系统发动攻击所依赖或者所使用的主要系统资源之一.通过篡改、窃取系统中的重要数据,攻击者可以干扰甚至破坏系统的正常

运行,从而达到其攻击的目的.区别于后文中软件攻击面动态转移中的地址空间布局随机化和指令集随机化中的资源对象存在一定的相对固定性,数据本身便存在多样性与变化性.基于数据攻击面的动态转移技术并不是对任意数据进行动态化、随机化和多样化,其根据系统的防御需求,需要在保持数据语义不发生变更的前提下,通过对选定数据的形式、编码、格式、排列等进行动态变换,实现脆弱性的规避或者攻击行为的发现,从而提升系统的安全性.而根据实施方式的不同,基于数据攻击面的动态转移方法可以分为数据随机化、数据多样化两大类.

2.1.1 数据随机化

数据随机化的基本思路主要是改变数据在存储器中的存储方式[15],利用随机生成的密钥,通过异或操作对不同的数据对象(例如指令操作数、数组、指针等)进行随机化操作,使得存储器中的数据形式呈现随机动态变化的特性,从而使得攻击者攻击时访问内存错误或无法跳转至恶意代码指向地址,以挫败缓冲区溢出攻击.数据加密技术作为一种典型的数据随机化方法,对数据进行了随机化处理,可以借助密钥的变化,实现数据的动态变化,能够达到系统防御的效果.因此,根据前文基于数据攻击面的动态转移技术的概念,其应属于数据动态转移技术的语义范畴,但通用的数据加密技术已经广泛应用在数据安全存储和保密通信等领域,技术实现和理论研究也都有相关的详细描述,故本文不对数据加密的一般性实现做详细介绍,而是以数据加密技术为基础,介绍相关的数据随机化的动态转移方法.

对于指令操作数的随机化方法,Cadar 等人[15]为源代码划分等价类,对不同等价类分配掩饰码,通过异或操作对各个操作数进行加解密操作,并通过代码插桩来加解密内存访问,从而实现程序中数据的随机化保护.但是该方法存在着一定的缺陷,在其指针分析算法中没有区分结构体和数组等内部的变量,而是将其作为一个整体进行处理,分析的精度不够高.而在该研究的基础上,蔺羽佳等人[16]针对传统的数据随机化技术静态分析精度不高的问题,提出一种基于域敏感度指针分析算法的数据随机化方法,将随机化操作细化到结构体等聚合类变量的内部,采用域敏感的指针分析技术使得随机化更加细粒度,提高了保护效果.然而即便如此,上述研究中由于对同一等价类分配相同的密钥,仍然存在攻击者对于同一类名下不同数据项进行溢出攻击时,随机化采用的加解密技术失效从而无法抵御攻击的可能性.

Fen 等人[17]提出的数据随机化方法,并不是直接对数据进行加密,而是对数组和指针变量使用 32bit 的随机密钥进行异或加密,使得即使发生缓冲区溢出攻击,由于数组和指针均已被加密,并不会指向攻击者的恶意代码所指向的地址.但是,为了保证程序加密后的正常运行,密钥均是保存在每个程序头开始的 32 个 bit 中,攻击者一旦破解了加密方式,仍然可以成功实现缓冲区溢出攻击.

而近年来,全同态的加密方法的兴起,实现了对数据在加密条件下的操作,解决了数据随机化过程中保密性与可用性的兼容问题.Gentry[18]提出了第一个基于理想格的全同态加密方法,虽然该方案的现实可行性不高,但是也为后来的研究揭开了序幕.Brakerski 等人在上述研究的基础上提出了基于整数环上模切换(modulus switching)技术的 BGV 全同态方法[19],其减少了密钥的存储空间大小,计算效率高.随后,又提出了基于 LWE (learning with errors)的全同态加密方法[20],进一步提高了运算效率.随着研究的不断深入[21−23],加密的计算效率和安全性能都在不断提高,使得数据的随机化可以以更短的时间和更小的系统开销实施,从而通过保证数据的动态特性,使得攻击者难以利用数据项进行有效攻击.

2.1.2 数据多样化

数据多样化的基本思想主要是通过对数据集进行多样化处理,构造等价或等语义的多个数据集,再监测测试过程中的行为与输出,以鉴别系统或程序运行过程中是否存在攻击行为,解决其自身的设计缺陷问题.目前,数据多样化的主要方法包括多副本运行和多变体数据两大类.

对于多副本运行的方法,Ammann 等人[24]针对软件在特定输入下发生故障的问题,分析引起故障的数据集,并构造其等价数据集,在该软件的多个副本上并行测试,再使用投票机制决定输出,试图找到使得系统正常运行的数据集,以此解决系统自身设计缺陷,增大攻击者恶意篡改程序输出的难度.但是,可使原有系统正常运行的等价数据集较难获取、等价数据集的输出差异不明显、对投票机制的依赖,都可能导致该方法无法产生所需的

输出结果.

对于多变体数据的方法,Nguyen-Tuong 等人[25]将 *N* 变体架构的思想引入数据多样性的方法中,通过对特定类型的数据进行多样化处理,构造出与原程序语义一致的多个变体,监测并比较输入值经过不同变体的中间行为和最终输出,根据发生的分歧判定输入的合理性,以此鉴别是否存在攻击行为.但是该方法中,构建相同语义的变体难度较大,对系统和程序的代码实施修改也会导致很大的开销.

在随后的发展过程中,数据多样化方法更多地应用于软硬件的随机测试中.随机测试主要是指在利用被测系统很少量信息的情况下,以较少的代价自动生成大量的测试数据集,通过测试发现被测系统潜在的故障,评估被测系统的有效性.随机测试的优势在于其自动化程度高、易执行,能够避免测试人员的主观意识,但也存在测试用例质量差、可信度较低的缺陷.于是,其衍生而来的改进方法自适应随机测试[26]和准随机测试[27]近年来受到了研究者的广泛关注,相比较于传统的随机测试方法,该两类方法的测试数据集在输入空间中的分布更加均匀,能够降低生成测试用例的盲目性,从而有效提升发现系统错误的效率,有助于尽快实施错误的修复,提升被测系统的安全性和可靠性.

2.2 基于软件攻击面的动态转移技术

在各类攻击中,针对软件的攻击占有很大的比重,攻击的手段也十分多样,例如代码注入攻击[28]、缓冲区溢出攻击[29]、ROP(return-oriented programming)攻击[30]、数据泄露攻击[31]、软件恶意篡改[32]等.但是,大部分防御方法受先验知识和系统静态性的制约,只能被动地对攻击进行防御.面对当前攻击方法动态性、随机性、多样性的特点,基于软件攻击面的动态转移方法则通过密码技术和编译技术,对软件代码在地址空间布局、指令集、内存空间布局、程序的结构布局等方面采取动态化、随机化、多样化的处理,改变了软件的同质化现象;通过动态转移软件攻击面的方式,力求增加攻击者的攻击难度,增强对于上述攻击手段的有效防御.目前,基于软件攻击面的动态转移方法主要有地址空间布局随机化、指令集随机化、代码随机化、软件多态化技术等.

2.2.1 地址空间布局随机化

地址空间布局随机化(ASLR)最早是由 Forrest S 等人[33]提出的,其认为可以通过随机化进程地址空间的方法来实现计算机系统的多样化.而针对缓冲区溢出攻击,ASLR 也是目前最广泛使用的软件攻击面的动态转移方法[34−37].ASLR 可以在编译时随机化,以生成同一软件的不同版本,也可以在运行时随机化,对程序组件的地址进行动态布局.

ASLR 的随机化对象主要包括堆地址、栈基地址、进程环境块(process environment block,简称 PEB)地址和线程环境块(thread environment block,简称 TEB)地址、动态链接库地址等.堆地址随机化主要通过在堆上动态随机分布内存块,从而避免使用默认堆地址,使得攻击者难以预测下一次分配的内存块位置;栈基地址随机化主要通过在编译时在栈顶填充数据,随机调整函数栈帧的大小,或在加载时更改变量位置,使得攻击载荷对某些变体的溢出攻击失效;PEB/TEB 随机化主要通过修改内核模式下某些函数,使得原有指向固定地址的指针指向随机地址,从而使得 PEB/TEB 的地址随机化;动态链接库地址随机化主要通过 HOOK 技术获取到动态链接库的默认装载基地址,并进行随机化修改,从而攻击者无法正常调用该动态链接库.

然而,ASLR 方法自身仍然存在着如下的一些局限性:(1) 由于 ASLR 方法只随机化部分组件,攻击者利用未采用 ASLR 方法的程序组件,可以绕过 ASLR 进行溢出攻击[34];(2) 由于地址空间的有限性,可随机化的空间过小而导致攻击者可以采取暴力破解的方法获得跳转地址[35];(3) 通过覆盖存在溢出漏洞的函数的部分返回地址,使其与基地址的相对距离固定,寻找可用的跳转指令进行攻击[35];(4) 内存信息泄露的情况下,攻击者可能通过部分内存信息推导出地址发动攻击[36];(5) 通过处理器的内存管理单元(memory management unit,简称 MMU)中的缓存,攻击者可能实施侧信道攻击以获取虚拟地址,并进行真实地址的转换[37].

2.2.2 指令集随机化

指令集随机化(ISR)最早是由 Thimbleby[38]提出的,是主要应用于防御注入攻击的一类技术,通过对指令集进行特殊的随机化操作(例如内核修改、脚本语言随机化、Runtime 随机化、指令地址随机化等),实现软件攻击面的动态转移.随着研究的不断推进,衍生出了不同的研究方向,本文主要将其分为抵御 Web 攻击的 ISR 方法

和抵御二进制代码注入攻击的 ISR 方法两大类.

在抵御 Web 攻击方面,Geneiatakis[39]提出了一种对端用户透明的 ISR 方法,通过对运行环境引入随机化和范随机化的方式,实现了指令集随机化,并通过实验验证了该方法对 SQL 注入攻击较好的防御效果;Ping 等人[40]在分析当前防御 SQL 注入攻击方法的基础上,构建了一套基于数据库管理系统代理(DBMS proxy)的 SQL 关键字随机方法,由 DBMS proxy 分析 SQL 语法以鉴别是否发生攻击,并将随机化的 SQL 转换为正常状态后送至 DBMS,该方法达到了开销低、效果好的防御目标.

在抵御二进制代码注入攻击方面,Wartell 等人[41]引入了新的二进制代码混淆技术,通过在加载时自身随机化指令地址,由加载时间决定基本块地址,从而输出新的的二进制代码,并搭建出原型系统,在 Windows 和 Linux 平台上通过了测试,能够有效抵御代码注入攻击;Venkat 等人[42]基于异构指令集架构进行指令集随机化,并由运行时间决定是否执行随机化迁移,在很大程度上限制了攻击者利用二进制代码短段(gadget)的攻击.此外,该方法还部署有程序状态迁移模块,对注册表及内存信息进行迁移以防止攻击者利用.实验证明,该方法对抵御注入攻击与代码复用攻击有非常好的效果.Sinha 等人[43]提出一种名为 Polyglot 的 ISR 方法,该方法为了规避弱加密带来的隐患,采用 AES 和 ECC 算法对内存页进行加密,并通过代码、库文件的共享降低性能开销,能够在较低开销的情况下抵御注入攻击与代码复用攻击.

然而,ISR 虽然在理论上可以抵御各类代码注入型攻击,增大攻击受保护程序的难度,但是 ISR 方法也不可避免地存在着这样一些问题:(1) ISR 很多情况下需要构建硬件的执行环境,需要对处理器进行修改,即使通过修改软件执行环境实现,也会对性能产生较大的影响;(2) ISR 方法由于密钥与进程的关联性,使得应用程序在加载和运行过程中较难适应动态链接库;(3) ISR 方法在面对攻击者从内存中读取密钥或恢复出加密代码段的攻击时,将无法防范.

2.2.3 代码随机化

尽管前面介绍的 ASLR 和 ISR 方法能够在一定程度上抵御注入攻击,但是也催生出了新的攻击手段,返回导向编程(return-oriented programming,简称 ROP)攻击[44].ROP 攻击属于代码复用攻击,起源于 return-into-libc (RILC)攻击[45].ROP 攻击利用程序中已存在的代码,选取多个由 ret 指令结束的二进制代码短段(Gadget),按照一定方式组成攻击单元;或者利用间接跳转(indirect jump)的方式,选取内存中与 ret 类似的其他指令序列进行攻击,例如 POP-JMP 和 JOP(jump-oriented programming,跳转导向编程)[46].此外,近年来出现的即时返回导向编程(JIT- ROP)攻击,更是只需要很少量的泄露地址便可绕过细粒度的 ASLR 防御手段,其对软件攻击面的动态转移技术提出了更加严峻的挑战.于是,代码随机化方法便应运而生,针对 ROP 攻击等一类代码复用攻击,通过随机化方式修改二进制代码,使得攻击者无法利用程序中原有代码,试图以此抵御代码复用攻击.

Pappas 等人[47]提出了一种就地代码随机化的方法,在不插入额外代码、不依赖符号调试信息的情况下,只修改能够从编译的二进制文件中安全抽取的部分代码,由于保留了原有指令长度和基本程序块,使得代码的语义并不会被破坏,却降低了攻击者对内存的知悉程度,从而抵御了 ROP 攻击,但是该方法并不能保证攻击者无法使用非随机化的 Gadget 生成有效负载;Koo 等人[48]在上述方法的基础上提出了新的代码随机化方法,通过将未随机化的 Gadget 移动至随机位置,并用陷阱指令代替原代码以诱导攻击者,该方法提高了代码的随机化覆盖程度,有效解决了 Pappas 等人方法的不足;Chen 等人[49]针对攻击者获取到足够的 Gadget 便可绕过现有的代码随机化方法实施攻击的问题,提出了一种名为 CodeArmor 的代码随机化方法,通过对代码空间的虚拟化,使得内存中的代码指针不会泄露其相应的具体代码空间,并持续对虚拟化后的代码空间到真实代码地址的映射随机化.该方法能够显著减小软件攻击面,使得攻击者无法利用 Gadget 进行攻击,能够有效抵御即时的返回导向编程(JIT-ROP)攻击.

但是上述的方法也不能一劳永逸,Snow 等人[50]提出了一种动态生成 ROP Gadget 链的攻击方法,可以绕过上述代码随机化的防御方法,不过,该方法需要在线读取内存中大量可执行代码来实现代码复用攻击,存在一定的局限性;Carlini 等人[51]提出了 3 种新的 ROP 攻击方法,分别针对 ret 指令、gadget 以及伪造历史记录的方式达到攻击目的;Maisuradze 等人[52]提出了攻击者可以在未读取任何代码片段的情况下,通过即时编译器注入可

预见的 Gadget 来触发控制流指令中的位移,从而实施 JIT-ROP 攻击.

2.2.4 软件多态化

由于现有的软件开发部署主要从成本和可用性角度考虑,造成源代码采用同样的编译器、同样的方法进行编译链接,再进行分发销售.于是,攻击者只要发现一个漏洞就可以很轻松地运用到同一版本的所有该软件上,软件存在很大的单一性和静态性.由于软件中存在这类安全隐患,由此产生了软件多态化技术[53].其通过在软件开发过程中为同一源代码生成同一功能、不同结构的大量软件实体,使得每位用户的同一软件都存在着内部结构的差异性,从而使得攻击者攻破不同用户的难度加大.目前,主要的软件多态化方法主要分为反向堆栈技术、寄存器随机化技术、指令修改技术、多变体运行技术等.

- 反向堆栈技术主要是通过扩展原有的堆栈操作指令,构建出一个具有向上增长堆栈的变体[54].通过改变堆栈的增长方向,使得修改后的堆栈中包含完全不同的变量,以抵御缓冲区溢出攻击和典型的堆栈粉碎攻击;
- 寄存器随机化技术[55]主要是通过交换寄存器中的内容,使得由于攻击者依赖原先寄存器中的固定内容发生变化,而导致攻击失效.但是,还没有支持随机化处理过的寄存器的硬件架构,所以需要在指令运行前提前交换寄存器中的内容.或者,只改变存储变量和临时变量的寄存器,将是一种较轻量级的方法;
- 指令修改技术主要是利用指令调度,调用内联、循环迭代分布、部分冗余删除等方法来改变生成的机器代码,可以用来抵御 ROP 攻击和其他一些依赖特定指令存在于特定位置的攻击.此外,还包括向代码中插入无操作序列(no operation sequence,简称 NOP)、引入偏移量从而改变后续代码序列位置的方法[56]和为现有指令进行等价指令替换的方法[57];
- 多变体运行技术[58,59]主要是通过同时运行多个保持相同语义的变体,并观察和监测各个变体在同步点的行为,在保证输入一致的前提下,监控程序监测到与其他变体发生不一致的行为时,对该变体是否存在攻击行为进行分析,并终止该变体继续运行,同时选用其他变体的输出作为程序执行的结果.

然而,这些方法无疑会加大软件开发的成本,且由于分发给客户不同版本的软件,也会带来管理维护成本的相应提高.同时,不同的软件版本也会带来软件校验方法无法适用的问题.而多变体同时运行除了带来更大的性能开销外,也存在着灵活性不高、重建变体相对困难等问题.

2.3 基于网络攻击面的动态转移技术

网络攻击主要是通过信息收集、分析、整理之后,发现网络、目标系统或应用中的漏洞,有针对性地对逐步渗透到网络内部,对目标系统进行资源入侵与破坏、机密信息窃取,同时监视和控制目标系统的活动.目前,网络攻击手段的种类越来越多,自动化能力越来越强,大大增强了攻击的强度和传播速度,为网络空间安全带来了极大的安全威胁.

现有的网络防御方法通常采用防火墙、入侵检测系统、用户认证、数据加密和解密、漏洞扫描、防病毒软件等,但任何单一安全防护技术已经不能确保网络和系统的安全,而且大部分安全防御技术是被动、滞后的.由于网络架构和配置的静态性,这些方法在检测攻击时依赖先验知识,通过静态特征对攻击进行匹配,而攻击者可以持续地收集和分析网络和系统的信息,使得其有充分的时间找到目标中存在的缺陷和漏洞,从而对目标发动攻击.

基于网络攻击面的动态转移方法,主要的目的便是切断攻击者网络侦查和探测目标漏洞,阻碍攻击者访问目标节点这一环节.迫使攻击者不断追逐攻击目标,在不中断正常网络通信的前提下,阻止攻击者连接到目标系统或者引导攻击者连接到虚假、错误的目标,消除攻击者攻击时间、空间上的优势,阻止其进行后续攻击.

目前,基于网络攻击面的动态转移方法主要有动态网络架构、网络地址随机化、端信息跳变等方法.

2.3.1 动态网络架构

在传统的静态网络架构中,主要采用的是静态网络安全技术,架构内各组件的防御或检测能力也是静态的.且各组件孤立工作,不能实现有效的信息共享、协同工作.为了解决静态网络架构中存在的不足之处,研究者们提出了动态网络架构的思想,通过实施网络攻击面资源的转移,动态变换网络资源及配置,同时保持架构内各部

分组件的信息交换和协同转移,以此提高应对网络攻击的动态防御能力.

对传统网络架构的研究中,可变网络(mutable networks,简称 MUTE)[60]、自屏蔽动态网络架构(self-shielding dynamic network architecture,简称 SDNA)[61,62]、MOTAG 架构[63]、移动自组网络(mobile ad hoc networks,简称 MANETs)[64]均在文献[2]中有详细说明,故本文不再赘述.

对软件定义网络(software-defined networking,简称 SDN)架构的研究中,Kampanakis 等人[65]在 SDN 框架上实现了网络攻击面的动态转移,由 SDN 控制器抽象出当前网络状态,并按照一定的时间间隔随机改变主机地址、路由等网络变量,通过分析网络和配置信息提取出实时数据信息,以对当前所遭受的威胁和攻击进行评估;Wang 等人[66]提出了基于 SDN 的嗅探反射(sniffer reflector)架构,该架构主要分为扫描传感器、反射器、暗网这 3 个组成部分,通过扫描传感器监测目标主机与网络之间的流量,当监测到攻击者的扫描流量时将其送至反射器,随后将该流量重定向至暗网,通过转移诱导攻击者攻击错误目标,提升目标网络的安全性.

尽管目前的动态网络架构都存在着各自的优点,但是大多应用在不同的场景下,应对不同的攻击手段,并没有出现通用的、部署容易、优化开销的架构方案.

2.3.2 网络地址随机化

网络地址随机化的思想主要是为网络地址引入动态变换更新机制,通过对网络数据包进行特殊的处理或通过一定机制协同变换网络地址,使得攻击者始终无法确定通信双方的真实地址,从而破坏了攻击者的嗅探攻击,实现了对主机的隐私保护,以此提升通信双方的安全性.

Jafarian 等人[67]提出了一种采用 OpenFlow 的 IP 地址动态变换方法,基于可满足性模理论(satisfiability modulo theories,简称 SMT),由 OpenFlow 控制器分配虚拟 IP 地址(virtual IPs,简称 vIPs),等概率分配或由管理员分配权重后,在一定的时间间隔后重新分配.但是,该方法只能部署在 SDN 架构下,并不能部署在传统网络.后来,该团队又提出了适用于传统网络的时空地址动态变换方法[68],该方法中,每台主机都与一个独立的临时 IP 地址集相关联,且在随机时间间隔后不断发生地址变换.每次更新的 IP 地址由原主机身份和时间决定,使得用户无法知晓通信对方的实际地址,也增大了攻击者连接特定目标主机的难度.Wang 等人[69]针对传统网络架构下网络地址转移范围的局限性,提出一种基于 SDN 的网络地址动态转移方法,在终端主机之间通信时,需要通过 POX 控制器进行网络地址和 DNS 响应信息的分发和管理.该方法有效地突破了传统网络地址转移的范围限制,增大了攻击者获取真实主机地址的难度.Makanju 等人[70]针对 SDN 网络地址转移方法的转移范围过大而存在无效转移地址的问题,引入进化算法对大范围的地址空间进行检索,并选择合适的网络地址进行转移.该方法提供了一种新的研究思路,但也存在着搜索时间可能较长,导致攻击者有机会获取系统信息以及网络环境中组成元素与进化算法原应用场景中人口元素不能完全对应等问题,值得后续跟进研究.

在实际工程化时,网络地址随机化的方法还可以做进一步扩展,采用更多的干扰方式,包括改变系统发送数据分组的时序、更改数据分组的大小、伪造数据分组等方法,或者可以采取对数据载荷加密的方式,使得攻击者技术截获也无法分析数据分组的内容,从而提高本方法的有效性和可靠性.此外,随着 SDN 的发展,以及越来越多的设备支持 OpenFlow 协议,也使得网络地址随机化方法在实际网络环境的工程实现和应用更加便捷.

2.3.3 端信息随机跳变

端信息跳变主要是指在端到端的数据传输中,通信双方或一方按照协定伪随机地改变端口、地址、时隙、加密算法甚至协议等信息,从而实现网络的主动防御.

林楷等人[71]提出了基于消息篡改的端信息跳变技术,在数据消息离开主机系统之前,仅对其源或目的端信息进行增量式修改并重新计算校验和,实现消息篡改,从而迷惑攻击者.并建立了跳变栈模型,其中包括用户层跳变、内核层跳变和网络层跳变,同时,分析了各层跳变方案的优势和不足,并实验验证了该方法具有较高安全性能和服务性能.Luo 等人[72]提出一种基于端口和地址的随机跳变方法,该方法中,每个服务器都与虚拟地址空间范围相关联,每个运行的应用都与虚拟端口范围相关联.在固定的跳变间隔中,服务器会根据共享的密钥、源 ID、服务 ID 和时间,利用伪随机函数生成新的 IP 地址和通信端口,使得在不同时隙内,通信双方必定会使用不同的 IP、端口对,能够有效抵御攻击者的侦查和探测.

此外,随着 SDN 研究的不断推进,也涌现出了大量基于 SDN 架构的端信息跳变技术.Ma 等人[73]和雷程等人[74]针对当前跳变机制大多采用随机跳变、跳变空间有限的情况,提出一种自适应的端信息跳变方法,通过基于 Sibson 熵的感知分析网络安全状态和攻击者的扫描策略,制定基于视觉距离的跳变决策;同时,在 SDN 架构上基于 SMT 理论实现了低开销的启发式端信息跳变.Zhao 等人[75]提出一种基于 SDN 架构的双重跳变方法,通过同时跳变通信双方的端信息以及通信路径,使得通信数据将会由多条路径传送,同时,多个用户的数据流也会混合.该方法混淆了攻击者,使得攻击者无法准确区分单一用户的通信数据,从而增大了攻击者实行嗅探攻击的难度.

端信息跳变方法通过双方协同或一方的端信息跳变,能够使得攻击者在实施攻击前丢失其攻击目标,且难以截获双方的通信数据.但该方法对于同步和全局协调要求较高,若能充分利用 SDN 架构的灵活性、透明性的优势,则能更好地提升动态防御的有效性和可靠性.目前,在该方面开展的研究正在不断兴起.

2.4 基于平台攻击面的动态转移技术

平台,主要是指能够承载应用运行的软/硬件环境,其中包括处理器、操作系统、虚拟化平台及具体应用的开发环境等.基于平台攻击面的动态转移方法,主要是针对传统平台单一架构的缺陷,采用构建多样化运行平台的方式,动态改变应用的运行环境和系统配置,使系统呈现出随机性、不确定性和动态性,以此来提高攻击者对系统进行攻击的难度.

目前,基于平台攻击面的动态转移方法主要有平台动态迁移、虚拟化技术、Web 应用动态防御、移动平台动态防御等.

2.4.1 平台动态迁移

平台动态迁移主要是指通过构建多个平台,并使运行在其上的应用能够以可控的方式随机地在不同的平台上迁移,减少攻击者探测和攻击的成功几率,最终实现提升系统防御能力的目的.其中,又分为同平台的多配置迁移和多平台迁移.

对于同平台的多配置迁移,Lucas 等人[76]和 John 等人[77]提出了一种进化算法来创建多种平台配置,该方法基于原配置,采用进化算法对其进行混杂操作,确保其新生成的配置与原先不同,并根据配置信息评估当前的安全等级.该方法赋予了平台攻击面更多的变量,从而提升了抵御攻击的能力.但是,由于该方法需要根据收集到的不同安全事件重新生成新的配置,并对其进行安全评估,使得维护的开销很大.因此,该方法的应用范围目前也只局限在 Apache 服务器的 RedHat 平台上.

对于多平台迁移,Thompson 等人[78]构建了一种多操作系统轮换环境,对不同的主机分发不同的 Linux 版本,由管理员控制主机进行周期性的轮换,从而增大攻击者发现漏洞的难度.但是,该方法还只局限于操作系统多样性而进行的轮换.随后,Thompson 等人[79]又将上述思想扩展到 Web 服务平台之上,对于 Web 入流量,按照随机的时间间隔将其随机引入至 Apache 或 Nginx 服务器,构建出一种动态的 Web 应用轮换环境,通过减少平台暴露给潜在攻击者的时间,降低 Web 服务平台的被攻破可能,增强受保护 Web 应用的弹性.但是,该方法目前只适用于单一应用,且迁移的时间间隔的选取势必会存在防御开销与效果的权衡问题.Debroy 等人[80]在 SDN 架构下实现了多平台的动态迁移,该方法中,各个主机均与 SDN 控制器连接,并周期性地发送其状态信息.控制器根据遭受 DoS 攻击的可能性决定其迁移频率,并根据主机的状态、可用带宽以及受攻击的历史情况综合决定进行迁移的主机.该方法使得 DoS 攻击丢失其攻击目标,达到其动态防御的目的.

2.4.2 虚拟化技术

操作系统级虚拟化是在现有操作系统的基础上,以分区的方式提供多个独立的应用程序运行环境.各个分区都拥有完整的关键系统资源,与操作系统共用一个内核,其中的应用程序相互独立.而硬件级虚拟化是通过在硬件平台上生成多个可以独立运行操作系统的虚拟机实例,将硬盘分区、存储扇区、硬件和 CPU 等虚拟化,是从硬件角度对资源进行配置.由于本文介绍的虚拟化技术既包含操作系统级的虚拟机技术又包含虚拟机、虚拟服务器等硬件级虚拟化技术,故统称为虚拟化技术.虚拟化技术的应用在很大程度上解决了原先很多在实际网络环境中难以实现或难以部署的动态迁移方法,使得网络攻击面的动态转移方法有了更深入、更广泛的应用.

Okhravi 等人[81]提出一种可信动态逻辑异构系统(TALENT),运用操作系统级的虚拟化为应用提供迁移环境,并设立检查点来检测应用的运行状况,结合安全态势决定迁移的频率,通过应用迁移,使得攻击者无法锁定攻击目标.但该方法仍存在迁移开销大、迁移平台固定等问题.而 Bangalore 等人[82]提出的自清洗入侵容忍(self-cleansing intrusion tolerance,简称 SCIT)模型、Huang 等人[83]提出的移动攻击面(moving attack surfaces,简称 MAS)模型以及 Nguyen 等人[84]提出的移动目标防御自清洗入侵容忍(SCIT-MTD)方法中,通过创建多个虚拟服务器进行在线/离线状态的转换,提高 Web 服务的防御能力,使攻击者难以获得足够的攻击时间,但也存在资源冗余、管理开销大等问题.此外,转换的规律性、周期性,都有可能被攻击者利用.

在相关研究的推动下,基于虚拟化技术的迁移在云服务领域大放异彩.Peng 等人[10]提出了云服务的移动目标防御方法,通过在云上创建多个不同配置的虚拟机,使得服务可以在虚拟机之间迁移和部署,并兼顾迁移开销和资源冗余的问题,对虚拟机实施活跃/非活跃状态的切换.但是该方法适用于云服务部署密集、攻击者有较强攻击能力的情况;而对于云服务部署稀松、攻击者并没有集中攻击的情况,可能防御效果相比于静态防御方法并没有足够的优势.Jia 等人[85]设计了一种新的迁移机制,通过持续跟踪迁移后服务副本,快速识别和分离良性客户端会话和潜在恶意会话,并通过多轮迁移,逐步将持续攻击者同良性客户机隔离开来,以获得最大化的动态防御效果.Al-Salah 等人[86]构建了一种虚拟云系统模型,通过引入诱饵虚拟机(decoy virtual machines,简称 DVMs)来诱骗攻击者.其中,DVM 的部署和配置都由管理程序管控.通过在 DVM 上部署模拟的服务,迷惑攻击者,使得攻击者无法确定真实虚拟机的实际位置,增大攻击者的攻击难度,从而保护云服务的安全性.但是该方法中需要配置的 DVM 的数量并不明确,也没有体现对于开销和资源的考量,对于实际的部署可能还需要进一步的研究.Huang 等人[87]提出一种基于容器的云平台迁移技术,通过轻量级的操作系统虚拟化技术 Docker 对容器实行管理,并引入一种动态迁移策略,利用检测点机制、恢复模块、内存转储模块以及网络迁移模块对容器实施动态持续的迁移,实现了云平台上容器快速高效的迁移,保障了云平台的安全性,能够有效抵御边界信息泄露攻击.但是该方法仅采用仿真实现,对于大规模实际环境的部署以及迁移过程对容器内应用和服务的影响尚未考虑.

2.4.3 Web 应用动态防御

针对 Web 攻击面的攻击手段主要包括跨站脚本攻击、HTML 代码注入攻击以及服务器端代码注入攻击等.在前面的章节中,已经有部分内容关于指令集随机化的 Web 动态防御方法以及 Web 服务的虚拟化迁移方法等,但考虑到 Web 应用的动态防御方法远不止于此,故在本节单独介绍近年来的其他几种方法.

Vadlamudi 与 Sengupta 等人[88,89]在 Web 应用的移动目标防御中引入了博弈理论,考虑到攻击者获取 Web 应用漏洞与防御者实行配置的动态转移之间的关联性,在攻防行为之间建立了博弈模型.同时,基于国家漏洞数据库(national vulnerability database,简称 NVD)的通用漏洞列表(common vulnerabilities and exposures,简称 CVEs)构建双方的回报函数,并兼顾 Web 攻击面转移的开销,生成转移策略,以指导防御者实施 Web 应用相关配置的动态转移.但新的可用配置和新的攻击手段都会改变原有博弈,博弈过程的持续更新将成为后续的一个研究方向.

Heydari 等人[90,91]提出了一种基于 Mobile IPv6 协议实施移动目标防御的反审查方案.该方法将内容提供者(Web 服务器)在逻辑上作为移动节点,依赖未修改的标准 Mobile IPv6 协议,利用多个转交地址来为每个用户分配访问组,并以某种参数化的时间间隔进行随机混排来更改转交地址.除了使用多个转交地址,还使用 DNS 过滤和反向、IPSec 技术实现反审查,从而保护服务器和终端用户的隐私和安全.

Niakanlahiji 等人[92]提出一种 Web 应用的移动目标防御方法(WebMTD),该方法向 Web 应用程序的 HTML 元素添加新属性,向服务端代码添加新的变量,并保证属性和变量值随机生成且周期变化,以此来驱动 Web 攻击面不断变更.除了引入随机变量,WebMTD 还对应用安插安全检查函数,以此检验代码块和元素的真实性,确保应用未受到攻击者恶意篡改,从而保障了 Web 应用的安全性

2.4.4 移动平台动态防御

近年来,随着智能手机的大规模普及,针对移动平台的攻击数量也呈现出上升之势.其中,攻击手段主要包

括手机木马、代码注入、远程连接等.通过对手机用户的隐私、重要财产信息进行窃取,或者对移动终端进行恶意控制和破坏,造成了用户的信息泄露、财产损失,对手机用户造成了不可忽视的安全威胁.而为了应对日益增多的移动平台攻击,防御手段也应运而生,而移动平台的动态防御手段更是作为其中的后起之秀,近年来得到了很大的发展.

Lee 等人[93]针对 Android 系统 Zygote 进程创建模型导致应用代码加载在同一内存地址的问题,提出了一种替代方案 Morula,在每个进程的创建过程中,依赖库文件、代码和数据文件都要求分配在不同的内存区域,并通过地址空间布局随机化的方式随机分配每个进程的内存布局,使得 Android 平台的攻击面呈现动态变化.

Liang 等人[94]提出了一种基于 Android 系统的新的堆栈随机化方法,通过轻量级的指令随机化策略,在函数的调用之间插入随机填充的函数,同时,随机变更存储 push 和 pop 指令的寄存器中的操作数,在不修改操作系统的前提下实现了堆栈布局的随机化,以此抵御基于堆栈布局的代码注入攻击以及返回导向编程攻击等.

Braden 等人[95]针对 JIT-ROP 攻击可以通过发现目标上的实际代码布局并随时重新定位攻击有效载荷来规避随机化的问题,提出了一种 Android 系统上的代码随机化技术:泄漏弹性布局随机化(LR2).尽管以前的解决方案依赖于虚拟化,而 LR2 仅需要底层处理器来辅助实施,可以使随机化后的代码布局免于泄漏,规避了之前的抗泄漏方法的诸多限制,取得了很好的效果.

Parikh 等人[96]在分析 Android 系统 Zygote 的不足与缺陷后,建立了保持更新的 Android 漏洞库与 ROP 攻击数据集,并将动态二进制插桩(dynamic binary instrumentation)技术引入 Android 系统,通过对系统中已运行应用动态地安插跟踪点,从而持续地监测 Android 应用的运行情况,在发现疑似 ROP 攻击或绕过 ASLR 攻击时,立即终止应用的运行,保障系统免受攻击.

移动平台的动态防御方法采取动态修改应用软件,使其产生动态特性,而使得攻击者难以实施具有针对性、明确性的攻击.但是这方面的调整往往会对移动平台造成一定的额外开销,如何能更好地在提升动态防御有效性的同时保持较低的开销,也成为了目前兴起的研究点.

2.5 攻击面动态转移技术的总结分析

对于上述 4 类攻击面的动态转移技术,并没有统一的归纳总结方法,其中的各项技术既有相互关联的共用特性,又在某些方面有所区分,因此有必要对上述攻击面的动态转移技术从不同的角度进行深入的分析和比较.

对于数据攻击面的动态转移技术,本文主要介绍了数据随机化与数据多样化两大类,其共同点在于它们都改变了数据在不同空间和维度的存在形式,使得攻击者的常规攻击手段无法有效实施,但也有其各自的特点和应用范围:数据随机化技术主要对内存数据中不安全的操作数、指针等进行了分类加密,确保由一种类型数据溢出到另一种类型的数据无法被有效识别,这一技术对解决很多缓冲区溢出攻击都有很显著的防御效果;而数据多样化技术并不强调对数据的加密,而是构建出针对特定类型数据的变体,在防御攻击的过程中,使得系统行为产生差异从而检测出攻击行为,但该技术不像数据随机化可以直接抵御攻击,而是作为一种检测攻击的动态多样手段.

对于软件攻击面动态转移技术,其涉及了密码技术、编译技术、动态执行技术、反汇编技术等诸多领域,能够在软件开发、编译、链接、部署、载入和运行的多个阶段引入.本文主要从地址空间布局随机化、指令集随机化、代码随机化以及软件多态化这 4 个方面归纳了主流的软件攻击面动态转移技术,这些技术之间也同样存在着密切联系,地址空间布局随机化、指令集随机化能够在软件多态化技术生成多种软件变体时提供技术支撑,而代码随机化则是针对当前很多绕过地址空间布局随机化、指令集随机化的代码复用攻击衍生出的防御手段.由于软件漏洞的不可避免,面对不断演进的攻击手段,还需不断研究新型的软件攻击面动态转移技术.

对于网络攻击面动态转移技术,本文主要从动态网络架构、网络地址随机化以及端信息随机跳变这 3 个方面归纳了主流的动态转移手段,这些动态化技术的共同点在于它们转变了网络结构、网络通信、网络服务在不同时间、空间的呈现形式,使得攻击者常规的攻击手段难以施展.同时,这些技术与传统网络安全防护手段并不冲突,而是相辅相成的关系,传统防御手段可以在认证授权、入侵检测、数据安全防护等方面发挥作用;而网络攻击面动态转移手段则主要通过架构、通信内容的动态变化,加大攻击者攻击难度,提升防御效能.在后续的研

究工作中,一方面需要能够在保障正常网络服务的前提下寻求效果与开销的平衡点;另一方面,SDN 技术的发展也使得网络攻击面动态转移技术能够在大规模网络中部署实施成为可能.

对于平台攻击面动态转移技术,本文主要从平台动态迁移、虚拟化技术、Web 应用动态防御、移动平台动态防御这 4 个方面归纳了主流的平台动态转移技术,这 4 类技术的出发点大致相同,都是通过构建多样化的运行平台,以防御者可控的方式变换应用运行的环境,使得平台呈现出多样性、不确定性、随机化和动态性,能够有效缩短应用在单一平台上的暴露时间,使得攻击者难以实施有效的探测.同时,虚拟化技术的发展对后续研究的推动起到了至关重要的作用,使得平台、应用的迁移更加轻量高效,使得该技术具有良好的研究前景.

此外,如表 1 所示,本文还具体分析了 4 个层面的各类攻击面动态转移技术,从其相关技术、相关工作以及其优势和不足进行了总结和分析,结果表明了各类攻击面动态转移技术均能在一定程度上抵御某些攻击行为,但总体而言,也都普遍存在着性能影响较大、资源开销较高、防御效果有待提高等问题.

Table 1 Summary of attack surface dynamic transfer technology in existing study

表 1 现有攻击面动态转移技术的总结

攻击面类型	技术类型	相关技术	相关工作	优势	不足
数据攻击面	数据随机化	操作数加密	Cadar 等人[15], 蔺羽佳等人[16]	通过代码插桩加解密内存访问,实现了数据的随机化保护	对特定溢出攻击,数据加解密技术可能失效
		数组、指针加密	Fen 等人[17]	即使发生溢出也不会指向恶意代码指向的地址	加密方式存在被破解的可能性
		全同态加密	Gentry[18] Brakerski 等人[19,20]	密钥的存储空间不断减小,逐步提高安全性能和计算效率	构造方法没有突破,加解密效率仍然低下
	数据多样化	多副本运行	Ammann 等人[24]	构架等价数据集进行多副本测试,可寻找到使系统正常运行的数据集	过分依赖投票机制,能否找到使原系统正常运行的等价数据集不确定
		多变体数据	Nguyen-tuong 等人[25]	构造与原语义一致的多变体,通过监测鉴别出是否存在攻击行为	构建相同语义的变体实施的难度较大,修改代码、维护的开销很大
		随机测试	文献[26,27]	生成测试数据集,发现被测系统潜在的故障,评估被测系统的有效性	并不直接抵御攻击,而是检测有无潜在脆弱性
软件攻击面	地址空间布局随机化	堆地址随机化,栈基地址随机化,PEB/TEB 随机化,DLL 地址随机化	文献[33–37]	随机化进程地址空间来实现计算机系统的多样化,可以有效抵御部分缓冲区溢出攻击	无法应对绕过 ASLR 的溢出攻击;攻击者仍有可能获取到跳转地址
	指令集随机化	内核修改,脚本语言随机化,Runtime 随机化,指令地址随机化	Thimbleby[38], Geneiatakis[39], Ping 等人[40], Wartell 等人[41], Venkat 等人[42], Sinha 等人[43]	能够抵御一部分代码注入型攻击,增大攻击者攻击受保护程序的难度	大多情况下需要修改硬件环境,对性能影响很大;程序运行和维护需要额外开销;无法应对针对内存信息的攻击
	代码随机化	就地代码随机化,Gadget 随机化	Pappas 等人[47], Koo 等人[48], Chen 等人[49]	随机化部分代码,除去能被复用的 Gadget,降低攻击者对内存的知悉程度,抵御了 ROP 攻击	目前,Snow 等人[50]和 Carlini 等人[51]提出的新的 ROP 攻击手段
	软件多态化	反向堆栈技术,寄存器随机化,指令修改技术,多变体运行技术	文献[54–59]	不同版本软件存在内部结构的差异,加大攻击者攻破不同版本的难度	加大了软件开发的成本和管理维护开销;多态软件的灵活性不高,重建变体也相对困难

Table 1 Summary of attack surface dynamic transfer technology in existing study (Continued)

表 1 现有攻击面动态转移技术的总结(续)

攻击面类型	技术类型	相关技术	相关工作	优势	不足
网络攻击面	动态网络架构	传统网络架构	文献[60–64]	随机改变 IP 地址、路由等网络配置,阻碍攻击者扫描和发现网络目标	大多应用在不同的场景下,且部署的代价和开销都普遍较高,应对不同的攻击手段,并没有出现通用的、部署容易、优化开销的架构方案
		基于 SDN 的动态网络架构	Kampanakis 等人[65], Wang 等人[66]	按照一定间隔随机改变主机地址、路由等网络变量,并对当前攻击者进行错误目标的诱导	
	网络地址随机化	IP 地址动态变换	Jafarian 等人[67,68]	无法知晓通信对方实际地址,增大了攻击者连接特定目标主机的难度	地址变换的间隔随机,转移范围过大,搜索时间可能较长都会导致较大系统开销
		POX 控制器管理	Wang 等人[69]	有效地突破了传统网络地址转移的范围限制,增大了攻击者获取真实主机地址的难度	
		进化算法	Makanju 等人[70]	对大范围的地址空间进行检索,并选择合适的网络地址进行转移	
	端信息随机跳变	端口、IP 地址跳变	林楷等人[72], Luo 等人[73]	通过双方协同或一方的端信息跳变,能够使得攻击者在实施攻击前丢失其攻击目标,且难以截获双方的通信数据,能够有效抵御攻击者的侦查和探测	对于同步和全局协调要求较高,端信息跳变的有效性还有待提高
		基于 SDN 架构的端信息跳变	Ma 等人[89], 雷程等人[90], Zhao 等人[75]		
平台攻击面	平台动态迁移	多配置迁移	Lucas 等人[76], John 等人[77]	使平台攻击面获得更多变量,攻击者将难以找到合适的攻击路径	运行维护的开销很大,应用范围较为局限
		多平台迁移	Thompson 等人[78,79], Debroy 等人[80]	增大攻击者发现漏洞的难度,使得攻击者丢失攻击目标	局限于操作系统多样性而进行的轮换,无法根除系统漏洞
	虚拟化技术	TALENT	Okhravi 等人[81]	根据安全态势进行迁移,使得攻击者无法锁定攻击目标	仅针对几个特定应用进行实现,对大量应用进行迁移会造成很大开销
		SCIT	Bangalore 等人[82], Huang 等人[83], Nguyen 等人[84]	服务器的在线/离线切换,使得攻击者难以获取足够攻击时间	转换间隔较小,虚拟服务器的数量庞大,会造成大量资源冗余和开销
		云服务部署	Peng 等人[10], Jia 等人[85], Al-Salah 等人[86], Huang 等人[87]	通过服务器迁移迷惑攻击者,使其无法确定真正的攻击目标	对于非集中攻击的情况,效果可能不及静态防御,且造成额外开销
	Web 应用动态防御	博弈理论	Vadlamudi 与 Sengupta 等人[88,89]	根据博弈结果生成转移策略,指导防御者实施 Web 应用相关配置的动态转移	虽然能够一定程度提高 Web 安全性,但仍存在着适用范围小、效率低以及开销大等诸类问题
		Mobile IPv6	Heydari 等人[90,91]	以一定时间间隔进行随机混排来更改转交地址,保护服务器和终端用户的隐私和安全	
		WebMTD	Niakanlahiji 等人[92]	向服务端代码添加新的变量,以此来驱动 Web 攻击面不断变更	
	移动平台动态防御	Morula	Lee 等人[93]	随机分配每个进程的内存布局,使 Android 平台的攻击面呈现动态变化	往往会对移动平台造成一定的额外开销,影响用户的使用体验
		Android 平台堆栈随机化	Liang 等人[94]	轻量级的指令随机化,有效抵御代码注入攻击和 ROP 攻击	
		泄漏弹性布局随机化	Braden 等人[95]	仅需要底层处理器来辅助实施,可以使随机化后的代码布局免于泄漏	
		Android 动态二进制插桩	Parikh 等人[96]	对系统中已运行应用跟踪点,持续地监测应用的运行情况	

2.6 攻击面动态转移的策略

目前的攻击面动态转移方法大多都依赖于针对某些特定攻击,对相应的攻击面进行转移,从而实现抵御攻击、维护系统安全的目的.然而,对于如何进行转移、何时进行转移的问题,很多研究者在实行攻击面的动态转移时并没有充分考虑.于是,攻击面动态转移的策略的研究便应运而生.

关于博弈论方法的研究由来已久,而很大一部分研究者便将其融入到攻击面动态转移的策略研究中.通过对攻击者和防御方之间的动作、状态、攻击面转移作为特征,以双方的收益与付出作为评估标准,将攻击方和防御方之间的攻防过程建立博弈模型,以此求解防御方的最佳攻击面转移策略.Manadhata[97]从二人博弈的角度出发,对防御者和攻击者之间的相互关系进行建模,构造双方的回报函数,运用博弈论的方法来确定最佳的防御策略.但是该博弈方法是一种完全且完美的信息博弈,双方都清楚知晓对方的策略和回报,并且熟悉博弈之间已采取的各项操作,这种博弈方式不符合实际情况.Carter 等人[98]为了使得博弈过程更符合实际情况,假定攻击者拥有全部信息而防御者只拥有部分信息,构建攻防双方不完全信息的博弈,同时,对于静态攻击者和根据防御行为自适应改变攻击行为的这两类攻击者,分别建立模型,以此制定相应的防御决策.Wright 等人[99]针对前期研究中抵御 DDoS 攻击的一些防御手段,通过仿真的方式对其进行博弈论分析,以此评估攻击面转移策略在不同环境下的有效性和合理性.Feng 等人[100]向攻击面的动态转移策略方法中引入了信号传递博弈,通过防御者在执行转移行为后向攻击者泄露部分已部署的防御策略的方式,对攻击者后续的攻击行为造成干扰和影响,构建双方的贝叶斯斯塔克尔伯格博弈模型,以此寻求最佳的攻击面动态转移策略.

基于 Markov 模型的策略生成方法,也受到了一部分研究者的关注.Miehling 等人[101]对攻击者的可利用漏洞、攻击成功率、攻击路径等进行建模,并假设防御者只在特定时间才能观测到攻击行为并采取一定防御措施,同时对防御行为的开销进行定义,并将部分可观测马尔可夫决策过程模型(POMDP)引入其中,以此计算出可供选择的几种策略方案.Maleki 等人[102]将大量的攻击面转移技术抽象化,并为攻击面的动态转移过程建立 Markov 模型,以此评估攻击者攻击成功的概率与攻击者耗费时间与开销之间的关系;同时,根据该模型定义了安全强度的概念,以此来测量不同攻击面动态转移策略的有效性.Lei 等人[103]针对网络层攻击面的动态转移技术构建攻防双方的 Markov 博弈模型,其中兼顾双方、状态、攻防策略、转移概率、收益等.通过 Markov 决策过程描述网络状态在攻防进行博弈情况下的转移过程,并计算攻防双方的收益矩阵,以此选择最佳的防御策略.

此外,雷程和马多贺等人[73,74]提出的基于网络攻击面的自适应转换技术,通过对网络中攻击者扫描策略的感知和分析,制定出对应的转移策略,进行网络攻击面上端信息的跳变,也为后续攻击面动态转移策略的研究提供了一些思路.

3 未来研究方向展望

3.1 多层次攻击面动态转移技术的融合

尽管目前的攻击面动态转移方法能够通过改变特定系统资源属性或属性对外的呈现信息,使其攻击面发生变化,从而迷惑或误导攻击者,促使攻击者攻击错误目标或丢失攻击目标,改变网络防御的被动态势,提高系统的安全性,但是目前的攻击面转移技术大多只针对特定的某一类攻击或者某一攻击面而展开研究,这也导致目前的动态防御方法的适用范围较小.在不干扰现有安全防御手段的前提下,如何实现多层次攻击面动态转移技术的融合,形成体系化、系统化的动态防御体系,达到整体联动的动态防御效果,将需要进一步的研究.

3.2 攻击面动态转移的综合评估方法

进行攻击面的动态转移能否减小系统攻击面、提高系统安全等级、是否会对系统的可用性和性能造成影响,都是防御者最为关注的几个方面.目前的攻击面动态转移的评估方法中,大多集中在于攻击面的动态转移对于防御的有效性.同时,也有部分研究者提出基于 I/O 自动机的攻击面模型[8]、Markov 转移概率模型[97]、攻击图(attack graph)转移模型[104],通过这些模型,形式化地刻画攻击面的转移,并定量地对攻击面的转移以及防御的有效性、转移开销、性能耗费进行评估.但是,目前现有的这些评估手段大都由研究者针对特定攻击面的转移

提出,尚不成熟.而且目前的评估方法大多针对有效性,缺乏转移开销、性能耗费的评估,也缺少与不同类别转移技术的对比评估.因此,结合第 3.1 节中所提及的多层次攻击面转移的动态防御体系,实现定量描述系统攻击面的转移变化,综合评估系统的安全状态以及动态防御体系的有效性和开销,与不同类型转换技术的比较评估,将会是未来研究的一大方向.

3.3 基于威胁感知的攻击面动态转移方法

目前的攻击面动态转移方法大多缺乏对攻击以及威胁的动态感知,导致动态转移决策的选取具有盲目性.然而,盲目随机的转移方法一方面将极大地降低防御的有效性,另一方面,也会给系统带来较大的性能开销.虽然已经有部分研究者在网络攻击面上针对网络扫描阶段进行网络威胁的感知,分析攻击者的不同扫描策略行为,以指导后续的转移策略的生成[73,74],但是目前的研究仅处于起步阶段,具有一定的局限性.因此,如何实现系统的威胁感知,进行威胁信息的关联、融合,同时针对感知到的系统当前面临的威胁信息,有针对性地对相关的攻击面进行动态转移,以此制定出攻击面动态转移的最优转移策略,提升攻击面动态转移技术的有效性,将成为后续的一大研究方向.

3.4 基于三方博弈模型的攻击面转移决策

由于在攻击面动态转移的过程中,防御者为了减小系统的攻击面,往往需要修改或者禁用系统的某些特征;而系统在进行某项作业时,可能需要启用某些新特征或者修改某些原有特征,也存在着增大系统攻击面的可能.于是,防御者在实施动态转移攻击面的同时,必须在系统的安全性和可用性之间进行权衡,关于攻击面动态转移的决策方法的研究也随之应运而生.目前,主流的决策手段主要基于博弈论方法[105,106],通过对攻防双方的攻击行为、防御行为以及系统状态、行动回报等建立博弈模型,求解出使得防御方回报最大化的策略,据此来实行攻击面动态转移的决策.但是,目前基于博弈论的决策方法仅仅建立在攻击方和防御方两方上,并没有考虑到用户方参与博弈的情况.而且,该方法建立在攻防两方互相知晓对方的策略和回报的基础上,博弈模型还不够完善.于是,考虑到用户方的参与,如何建立预测博弈的后续行动和状态的三方博弈模型,兼顾安全成本和安全收益,可能将成为后续攻击面转移决策的一个研究方向.

4 结束语

攻击面的动态转移技术一直以来都是移动目标防御领域的重点问题.随着网络攻击技术和防御技术的不断演化与发展,该研究一直受到研究人员的广泛关注.针对这一问题,本文首先梳理了攻击面以及其动态转移的基本概念,然后从数据攻击面、软件攻击面、网络攻击面和平台攻击面这 4 个层次分别介绍了攻击面的动态转移技术,并对不同的转移技术进行分析和比较,分别指出它们的优点和缺陷,也得到了一些初步的结论.

我们认为,对攻击面动态转移技术研究的理解应体现在以下 5 个方面.

1) 攻击目标可变.当前,网络系统的确定性、静态性和同构性使得攻击者具有时间优势、信息不对称优势以及成本优势,也导致防御者自始至终处在被动的劣势地位.需要确定可变化的攻击面资源,通过动态转移,使得攻击面呈现出多样的变化,从而使得攻击面面对变化的攻击目标而无法实施针对性的攻击;
2) 资源实时掌握.知己知彼,百战不殆.防御者在作出攻击面动态转移的相关决策时,首先需要的便是对攻击面上存在资源的实时状态有着充分的了解,在面对攻击时,根据系统各层攻击面的实时情况,才能作出正确的防御决策,保证攻击面动态转移的有效性;
3) 响应转移迅速.在面对来势汹汹的进攻时,一方面需要保证攻击面动态转移的有效性,另一方面,防御方需要迅速作出响应,在攻击者进一步渗透系统其他资源之前,实现攻击面的动态转移,保证转移的时效性;
4) 攻击面转移可持续.在面对攻击者的攻击进行攻击面的动态转移时,不仅需要考虑规避攻击、提升系统安全,还要考虑动态转移的防御成本以及维护系统持续正常运行的性能,实现攻击面动态转移的

“可持续发展”;

5) 全方位攻击面转移.随着攻击手段的不断发展壮大,许多单一层次的攻击面转移往往存在着这样的问题:在进行某一攻击面动态转移的同时,反而暴露给攻击者其他层面的可利用的资源,仍然存在可乘之机.因此,要达到更好的防御效果,就需要多层次、全方位的攻击面动态转移,促使攻击者难以获取可利用资源,降低其攻击成功率.

目前,国内外有关攻击面动态转移技术的研究正处于快速发展的阶段,虽然已经有大量的具体实现的攻击面动态转移手段被提出,但其中也存在一些重要的研究方向才初步涉及,研究还没有深入开展,例如制定攻击面动态转移策略的相关研究、自适应的攻击面动态转移研究等.此外,还有许多其他的研究方面尚未涉及到,例如多层次融合的攻击面动态转移技术研究、攻击面动态转移的综合效能评估方法研究、基于威胁感知的攻击面动态转移方法研究、基于三方博弈的新型攻击面转移策略研究等,这些也在文中未来展望中有所提及,希望能够为后续的相关研究工作的开展提供建议与参考.

References:

[1] Jajodia S, Ghosh AK, Swarup V, *et al*. Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. New York: Springer Science & Business Media, 2011. 1–5.

[2] Zhang XY, Li ZH. Overview on moving target defense technology. Communications Technology, 2013,46(6):111−113 (in Chinese with English abstract).

[3] Cai GL, Wang BS, Wang TZ, *et al*. Research and development of moving target defense technology. Journal of Computer Research and Development, 2016,53(5):968−987 (in Chinese with English abstract).

[4] Jajodia S, Ghosh AK, Subrahmanian VS, *et al*. Moving Target Defense II: Application of Game Theory and Adversarial Modeling. New York: Springer Science & Business Media, 2013. 15−40.

[5] Okhravi H, Rabe MA, Mayberry TJ, *et al*. Survey of cyber moving target techniques. TR-1166. Lexington: Massachusetts Inst of Tech Lexington Lincoln Lab, 2013. 1–149.

[6] Howard M, Pincus J, Wing JM. Measuring relative attack surfaces. In: Lee DT, Shieh SP, Tygar JD, eds. Computer Security in the 21st Century. 2003. 109−137.

[7] Manadhata PK, Tan KM, Maxion RA, *et al*. An approach to measuring a system’s attack surface. No.0704-0188. Pittsburgh: Carnegie-Mellon Univ Pittsburgh Pa School of Computer Science, 2007. 1–29.

[8] Manadhata PK, Wing JM. An attack surface metric. IEEE Trans. on Software Engineering, 2011,37(3):371−386.

[9] Kurmus A, Tartler R, Dorneanu D, *et al*. Attack surface metrics and automated compile-time OS kernel tailoring. In: Proc. of the 20th Annual Network & Distributed System Security Symp. (NDSS). San Diego, 2013.

[10] Peng W, Li F, Huang CT, *et al*. A moving-target defense strategy for cloud-based services with heterogeneous and dynamic attack surfaces. In: Proc. of the 2014 IEEE Int’l Conf. on Communications (ICC). IEEE, 2014. 804–809.

[11] Foreman JC, Gurugubelli D. Identifying the cyber attack surface of the advanced metering infrastructure. The Electricity Journal, 2015,28(1):94−103.

[12] Sun K, Jajodia S. Protecting enterprise networks through attack surface expansion. In: Proc. of the 2014 Workshop on Cyber Security Analytics, Intelligence and Automation. ACM Press, 2014. 29–32.

[13] Cybenko G, Jajodia S, Wellman MP, *et al*. Adversarial and uncertain reasoning for adaptive cyber defense: Building the scientific foundation. In: Proc. of the Int’l Conf. on Information Systems Security. Cham: Springer-Verlag, 2014. 1–8.

[14] Bopche GS, Mehtre BM. Graph similarity metrics for assessing temporal changes in attack surface of dynamic networks. Computers & Security, 2017,64:16−43.

[15] Cadar C, Akritidis P, Costa M, *et al*. Data randomization. Technical Report, TR-2008-120, Cambridge: Microsoft Research, 2008.

[16] Man YJ, Yin Q, Zhu XD. Fine-Grained data randomization technique based on field-sensitive pointer analysis. Journal of Computer Applications, 2016,36(6):1567−1572 (in Chinese English abstract).

[17] Fen Y, Fuchao Y, Xiaobing S, *et al*. A new data randomization method to defend buffer overflow attacks. Physics Procedia, 2012, 24:1757−1764.

[18] Gentry C. Fully homomorphic encryption using ideal lattices. In: Proc. of the 41st Annual ACM Symp. on Theory of Computing (STOC). 2009,9(4):169−178.

[19] Brakerski Z, Gentry C, Vaikuntanathan V. (Leveled) fully homomorphic encryption without bootstrapping. ACM Trans. on Computation Theory (TOCT), 2014,6(3):13.

[20] Brakerski Z. Fully homomorphic encryption without modulus switching from classical GapSVP. In: Safavi-Naini R, Canetti R, eds. Proc. of the Advances in Cryptology (CRYPTO 2012). LNCS, Berlin: Springer-Verlag, 2012. 868−886.

[21] Berkoff A, Liu FH. Leakage resilient fully homomorphic encryption. In: Proc. of the Theory of Cryptography Conf. Berlin, Heidelberg: Springer-Verlag, 2014. 515−539.

[22] Ducas L, Micciancio D. FHEW: Bootstrapping homomorphic encryption in less than a second. In: Proc. of the Annual Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer-Verlag, 2015. 617−640.

[23] Lai J, Deng RH, Ma C, *et al*. CCA-Secure keyed-fully homomorphic encryption. In: Cheng CM, Chung KM, Persiano G, Yang BY, eds. Proc. of the Public-Key Cryptography (PKC 2016). LNCS, Berlin: Springer-Verlag, 2016. 70−98.

[24] Ammann PE, Knight JC. Data diversity: An approach to software fault tolerance. IEEE Trans. on Computers, 1988,37(4):418−425.

[25] Nguyen-Tuong A, Evans D, Knight JC, *et al*. Security through redundant data diversity. In: Proc. of the IEEE Int'l Conf. on Dependable Systems and Networks with FTCS and DCC (DSN 2008). IEEE, 2008. 187−196.

[26] Barus AC, Chen TY, Kuo FC, *et al*. A cost-effective random testing method for programs with non-numeric inputs. IEEE Trans. on Computers, 2016,65(12):3509−3523.

[27] Liu H, Chen TY. Randomized quasi-random testing. IEEE Trans. on Computers, 2016,65(6):1896−1909.

[28] Mitropoulos D, Spinellis D. Fatal injection: A survey of modern code injection attack countermeasures. PeerJ Computer Science, 2017,3:e136.

[29] Nashimoto S, Homma N, Hayashi Y, *et al*. Buffer overflow attack with multiple fault injection and a proven countermeasure. Journal of Cryptographic Engineering, 2017,7(1):35−46.

[30] Prandini M, Ramilli M. Return-Oriented programming. IEEE Security & Privacy, 2012,10(6):84−87.

[31] Alneyadi S, Sithirasenan E, Muthukkumarasamy V. A survey on data leakage prevention systems. Journal of Network and Computer Applications, 2016,62:137−152.

[32] Lin J, Mi C, Shi Y. Approach of tamper detection for sensitive data based on negotiable hash algorithm. Int'l Journal of Performability Engineering, 2017,13(5):711.

[33] Forrest S, Somayaji A, Ackley DH. Building diverse computer systems. In: Proc. of the 6th Workshop on Hot Topics in Operating Systems. IEEE, 1997. 67−72.

[34] Seo J, Lee B, Kim S, *et al*. SGX-Shield: Enabling address space layout randomization for SGX programs. In: Proc. of the 2017 Annual Network and Distributed System Security Symp. (NDSS). San Diego, 2017.

[35] Chen Y, Wang Z, Whalley D, *et al*. Remix: On-demand live randomization. In: Proc. of the 6th ACM Conf. on Data and Application Security and Privacy. ACM Press, 2016. 50−61.

[36] Werner J, Baltas G, Dallara R, *et al*. No-Execute-After-Read: Preventing code disclosure in commodity software. In: Proc. of the 11th ACM on Asia Conf. on Computer and Communications Security. ACM Press, 2016. 35−46.

[37] Gras B, Razavi K, Bosman E, *et al*. ASLR on the line: Practical cache attacks on the MMU. In: Proc. of the 2017 Annual Network and Distributed System Security Symp. (NDSS). San Diego, 2017.

[38] Thimbleby H. Can viruses ever be useful? Computers & Security, 1991,10(2):111−114.

[39] Geneiatakis D. Minimizing databases attack surface against SQL injection attacks. In: Proc. of the Int'l Conf. on Information and Communications Security. Springer Int'l Publishing, 2015. 1−9.

[40] Ping C, Jinshuang W, Lin P, *et al*. Research and implementation of SQL injection prevention method based on ISR. In: Proc. of the 2016 2nd IEEE Int'l Conf. on Computer and Communications (ICCC). IEEE, 2016. 1153−1156.

[41] Wartell R, Mohan V, Hamlen KW, *et al*. Binary stirring: Self-randomizing instruction addresses of legacy x86 binary code. In: Proc. of the 2012 ACM Conf. on Computer and Communications Security. ACM Press, 2012. 157−168.

[42] Venkat A, Shamasunder S, Shacham H, *et al*. Hipstr: Heterogeneous-isa program state relocation. ACM SIGARCH Computer Architecture News, 2016,44(2):727−741.

[43] Sinha K, Kemerlis VP, Sethumadhavan S. Reviving instruction set randomization. In: Proc. of the 2017 IEEE Int'l Symp. on Hardware Oriented Security and Trust (HOST). IEEE, 2017. 21−28.

[44] Lee J, Jang J, Jang Y, *et al*. Hacking in darkness: Return-oriented programming against secure enclaves. In: Proc. of the USENIX Security. 2017. 523−539.

[45] Tran M, Etheridge M, Bletsch T, *et al*. On the expressiveness of return-into-libc attacks. In: Proc. of the Recent Advances in Intrusion Detection. Berlin, Heidelberg: Springer-Verlag, 2011. 121−141.

[46] Ruan Y, Kalyanasundaram S, Zou X. Survey of return-oriented programming defense mechanisms. Security and Communication Networks, 2016,9(10):1247−1265.

[47] Pappas V, Polychronakis M, Keromytis AD. Smashing the gadgets: Hindering return-oriented programming using in-place code randomization. In: Proc. of the 2012 IEEE Symp. on Security and Privacy (SP). IEEE, 2012. 601−615.

[48] Koo H, Polychronakis M. Juggling the gadgets: Binary-level code randomization using instruction displacement. In: Proc. of the 11th ACM on Asia Conf. on Computer and Communications Security. ACM Press, 2016. 23−34.

[49] Chen X, Bos H, Giuffrida C. CodeArmor: Virtualizing the code space to counter disclosure attacks. In: Proc. of the 2017 IEEE European Symp. on Security and Privacy (EuroS&P). IEEE, 2017. 514−529.

[50] Snow KZ, Monrose F, Davi L, *et al*. Just-in-Time code reuse: On the effectiveness of fine-grained address space layout randomization. In: Proc. of the 2013 IEEE Symp. on Security and Privacy (SP). IEEE, 2013. 574−588.

[51] Carlini N, Wagner D. ROP is still dangerous: Breaking modern defenses. In: Proc. of the USENIX Security Symp. 2014. 385−399.

[52] Maisuradze G, Backes M, Rossow C. What cannot be read, cannot be leveraged? Revisiting assumptions of JIT-ROP defenses. In: Proc. of the USENIX Security Symp. 2016. 139−156.

[53] Temizkan O, Park S, Saydam C. Software diversity for improved network security: Optimal distribution of software-based shared vulnerabilities. Information Systems Research, 2017,28(4):828−849.

[54] Cui W, Peinado M, Cha SK, *et al*. Retracer: Triaging crashes by reverse execution from partial memory dumps. In: Proc. of the 38th Int'l Conf. on Software Engineering. ACM Press, 2016. 820−831.

[55] Crane S, Liebchen C, Homescu A, *et al*. Readactor: Practical code randomization resilient to memory disclosure. In: Proc. of the 2015 IEEE Symp. on Security and Privacy (SP). IEEE, 2015. 763−780.

[56] Tagatac DM, Polychronakis M, Stolfo SJ. Using diversity to harden multithreaded programs against exploitation. 2016 IEEE 2nd Int'l Conf. on Big Data Security on Cloud (BigDataSecurity), IEEE Int'l Conf. on High Performance and Smart Computing (HPSC), and IEEE Int'l Conf. on Intelligent Data and Security (IDS). IEEE, 2016. 208−213.

[57] Shterenberg SI, Krasov AV, Ushakov IA. Analysis of using equivalent instructions at the hidden embedding of information into the executable files. Journal of Theoretical and Applied Information Technology, 2015,80(1):28.

[58] Volckaert S, Coppens B, De Sutter B. Cloning your gadgets: Complete ROP attack immunity with multi-variant execution. IEEE Trans. on Dependable and Secure Computing, 2016,13(4):437−450.

[59] Volckaert S, Coppens B, De Sutter B, *et al*. Taming parallelism in a multi-variant execution environment. In: Proc. of the 12th European Conf. on Computer Systems. ACM Press, 2017. 270−285.

[60] Al-Shaer E. Toward network configuration randomization for moving target defense. In: Jajodia S, Ghosh AK, Swarup V, *et al*, eds. Proc. of the Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. New York: Springer Science & Business Media, 2011. 153−159.

[61] Yackoski J, Bullen H, Yu X, *et al*. Applying self-shielding dynamics to the network architecture. In: Jajodia S, Ghosh AK, Swarup V, *et al*, eds. Proc. of the Moving Target Defense II: Application of Game Theory and Adversarial Modeling. New York: Springer Science & Business Media, 2013. 97−115.

[62] Yackoski J, Li J, DeLoach SA, *et al*. Mission-Oriented moving target defense based on cryptographically strong network dynamics. In: Proc. of the 8th Annual Cyber Security and Information Intelligence Research Workshop. ACM Press, 2013. 57.

[63] Jia Q, Sun K, Stavrou A. Motag: Moving target defense against internet denial of service attacks. In: Proc. of the 2013 22nd Int'l Conf. on Computer Communications and Networks (ICCCN). IEEE, 2013. 1−9.

[64] Albanese M, De Benedictis A, Jajodia S, *et al*. A moving target defense mechanism for Manets based on identity virtualization. In: Proc. of the 2013 IEEE Conf. on Communications and Network Security (CNS). IEEE, 2013. 278−286.

[65] Kampanakis P, Perros H, Beyene T. SDN-Based solutions for moving target defense network protection. In: Proc. of the 2014 IEEE 15th Int'l Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM). IEEE, 2014. 1−6.

[66] Wang L, Wu D. Moving target defense against network reconnaissance with software defined networking. In: Proc. of the Int'l Conf. on Information Security. Cham: Springer-Verlag, 2016. 203−217.

[67] Jafarian JH, Al-Shaer E, Duan Q. Openflow random host mutation: Transparent moving target defense using software defined networking. In: Proc. of the 1st Workshop on Hot Topics in Software Defined Networks. ACM Press, 2012. 127−132.

[68] Jafarian JHH, Al-Shaer E, Duan Q. Spatio-Temporal address mutation for proactive cyber agility against sophisticated attackers. In: Proc. of the 1st ACM Workshop on Moving Target Defense. ACM Press, 2014. 69−78.

[69] Wang S, Zhang L, Tang C. A new dynamic address solution for moving target defense. In: Proc. of the Information Technology, Networking, Electronic and Automation Control Conf., IEEE. IEEE, 2016. 1149−1152.

[70] Makanju A, Zincir-Heywood AN, Kiyomoto S. On evolutionary computation for moving target defense in software defined networks. In: Proc. of the Genetic and Evolutionary Computation Conf. on Companion. ACM Press, 2017. 287−288.

[71] Lin K, Jia CF. End hopping based on message tampering. Journal on Communications, 2013,34(12):142−148 (in Chinese with English abstract)

[72] Luo YB, Wang BS, Wang XF, *et al*. RPAH: Random port and address hopping for thwarting internal and external adversaries. In: Proc. of the 2015 IEEE Trustcom/BigDataSE/ISPA. IEEE, 2015. 263−270.

[73] Ma D, Lei C, Wang L, *et al*. A self-adaptive hopping approach of moving target defense to thwart scanning attacks. In: Proc. of the Int'l Conf. on Information and Communications Security. Cham: Springer-Verlag, 2016. 39−53.

[74] Lei C, Ma DH, Zhang HQ, Yang YJ, Wang LM. Moving target defense technique based on network attack surface self-adaptive mutation. Chinese Journal of Computers, 2018,41(5):1109−1131 (in Chinese with English abstract). http://kns.cnki.net/kcms/detail/11.1826.TP.20170819.0034.010.html

[75] Zhao Z, Gong D, Lu B, *et al*. SDN-Based double hopping communication against sniffer attack. In: Proc. of the Mathematical Problems in Engineering, 2016. 2016.

[76] Lucas B, Fulp EW, John DJ, *et al*. An initial framework for evolving computer configurations as a moving target defense. In: Proc. of the 9th Annual Cyber and Information Security Research Conf. ACM Press, 2014. 69−72.

[77] John DJ, Smith RW, Turkett WH, *et al*. Evolutionary based moving target cyber defense. In: Proc. of the Companion Publication of the 2014 Annual Conf. on Genetic and Evolutionary Computation. ACM Press, 2014. 1261−1268.

[78] Thompson M, Evans N, Kisekka V. Multiple OS rotational environment an implemented moving target defense. In: Proc. of the 2014 7th Int'l Symp. on Resilient Control Systems (ISRCS). IEEE, 2014. 1−6.

[79] Thompson M, Mendolla M, Muggler M, *et al*. Dynamic application rotation environment for moving target defense. In: Proc. of the 2016 Resilience Week (RWS). IEEE, 2016. 17−26.

[80] Debroy S, Calyam P, Nguyen M, *et al*. Frequency-Minimal moving target defense using software-defined networking. In: Proc. of the 2016 Int'l Conf. on Computing, Networking and Communications (ICNC). IEEE, 2016. 1−6.

[81] Okhravi H, Comella A, Robinson E, *et al*. Creating a cyber moving target for critical infrastructure applications using platform diversity. Int'l Journal of Critical Infrastructure Protection, 2012,5(1):30−39.

[82] Bangalore AK, Sood AK. Securing Web servers using self cleansing intrusion tolerance (scit). In: Proc. of the 2nd Int'l Conf. on Dependability 2009 (DEPEND 2009). IEEE, 2009. 60−65.

[83] Huang Y, Ghosh AK. Introducing diversity and uncertainty to create moving attack surfaces for web services. In: Jajodia S, Ghosh AK, Swarup V, *et al*, eds. Proc. of the Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats. New York: Springer Science & Business Media, 2011. 131−151.

[84] Nguyen QL, Sood A. Scalability of cloud based SCIT-MTD. In: Proc. of the 2017 IEEE Int'l Conf. on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 2017. 581−582.

[85] Jia Q, Wang H, Fleck D, *et al*. Catch me if you can: A cloud-enabled ddos defense. In: Proc. of the 2014 44th Annual IEEE/IFIP Int'l Conf. on Dependable Systems and Networks (DSN). IEEE, 2014. 264−275.

[86] Al-Salah T, Hong L, Shetty S. Attack surface expansion using decoys to protect virtualized infrastructure. In: Proc. of the 2017 IEEE Int'l Conf. on Edge Computing (EDGE). IEEE, 2017. 216−219.

[87] Huang R, Zhang H, Liu Y, *et al*. RELOCATE: A container based moving target defense approach. In: Proc. of the 2017 7th Int'l Conf. on Computer Engineering and Networks (CENet 2017). Shanghai, 2017. href="https://pos.sissa.it/cgi-bin/reader/conf.cgi?confid=299">https://pos.sissa.it/cgi-bin/reader/conf.cgi?confid=299,id.8

[88] Vadlamudi SG, Sengupta S, Taguinod M, *et al*. Moving target defense for Web applications using bayesian stackelberg games. In: Proc. of the 2016 Int'l Conf. on Autonomous Agents & Multiagent Systems. Int'l Foundation for Autonomous Agents and Multiagent Systems, 2016. 1377−1378.

[89] Sengupta S, Vadlamudi SG, Kambhampati S, *et al*. A game theoretic approach to strategy generation for moving target defense in Web applications. In: Proc. of the 16th Conf. on Autonomous Agents and MultiAgent Systems. Int'l Foundation for Autonomous Agents and Multiagent Systems, 2017. 178−186.

[90] Heydari V, Kim S, Yoo SM. Anti-Censorship framework using mobile ipv6 based moving target defense. In: Proc. of the 11th Annual Cyber and Information Security Research Conf. ACM Press, 2016. 7.

[91] Heydari V, Kim S, Yoo SM. Scalable anti-censorship framework using moving target defense for Web servers. IEEE Trans. on Information Forensics and Security, 2017,12(5):1113−1124.

[92] Niakanlahiji A, Jafarian JH. WebMTD: Defeating Web code injection attacks using Web element attribute mutation. In: Proc. of the 2017 Workshop on Moving Target Defense. ACM Press, 2017. 17−26.

[93] Lee B, Lu L, Wang T, *et al*. From zygote to morula: Fortifying weakened aslr on android. In: Proc. of the 2014 IEEE Symp. on Security and Privacy (SP). IEEE, 2014. 424−439.

[94] Liang Y, Ma X, Wu D, *et al*. Stack layout randomization with minimal rewriting of Android binaries. In: Proc. of the Int'l Conf. on Information Security and Cryptology. Springer Int'l Publishing, 2015. 229−245.

[95] Braden K, Davi L, Liebchen C, *et al*. Leakage-Resilient layout randomization for mobile devices. In: Proc. of the 20th Annual Network & Distributed System Security Symp. (NDSS). San Diego, 2016.

[96] Parikh V, Mateti P. ASLR and ROP attack mitigations for ARM-based android devices. In: Proc. of the Int'l Symp. on Security in Computing and Communication. Singapore: Springer-Verlag, 2017. 350−363.

[97] Manadhata PK. Game theoretic approaches to attack surface shifting. In: Jajodia S, Ghosh AK, Swarup V, *et al*, eds. Proc. of the Moving Target Defense II: Application of Game Theory and Adversarial Modeling. New York: Springer Science & Business Media, 2013. 1−13.

[98] Carter KM, Riordan JF, Okhravi H. A game theoretic approach to strategy determination for dynamic platform defenses. In: Proc. of the 1st ACM Workshop on Moving Target Defense. ACM Press, 2014. 21−30.

[99] Wright M, Venkatesan S, Albanese M, *et al*. Moving target defense against DDoS attacks: An empirical game-theoretic analysis. In: Proc. of the 3rd ACM Workshop on Moving Target Defense. ACM Press, 2016. 93−104.

[100] Feng X, Zheng Z, Cansever D, *et al*. A signaling game model for moving target defense. In: Proc. of the INFOCOM 2017—IEEE Conf. on Computer Communications. IEEE, 2017. 1−9.

[101] Miehling E, Rasouli M, Teneketzis D. Optimal defense policies for partially observable spreading processes on bayesian attack graphs. In: Proc. of the 2nd ACM Workshop on Moving Target Defense. ACM Press, 2015. 67−76.

[102] Maleki H, Valizadeh S, Koch W, *et al*. Markov modeling of moving target defense games. In: Proc. of the 3rd ACM Workshop on Moving Target Defense. ACM Press, 2016. 81−92.

[103] Lei C, Ma DH, Zhang HQ. Optimal strategy selection for moving target defense based on Markov game. IEEE Access, 2017,5: 156−169.

[104] Zhuang R, DeLoach SA, Ou X. A model for analyzing the effect of moving target defenses on enterprise networks. In: Proc. of the 9th Annual Cyber and Information Security Research Conf. ACM Press, 2014. 73−76.

[105] Do CT, Tran NH, Hong C, *et al*. Game theory for cyber security and privacy. ACM Computing Surveys (CSUR), 2017,50(2):30.

[106] Nguyen TH, Wright M, Wellman MP, *et al*. Multi-Stage attack graph security games: Heuristic strategies, with empirical game-theoretic analysis. In: Proc. of the 2017 Workshop on Moving Target Defense. ACM Press, 2017. 17−26.

附中文参考文献:

[2] 张晓玉,李振邦.移动目标防御技术综述.通信技术,2013,46(6):111−113.

[3] 蔡桂林,王宝生,王天佐,等.移动目标防御技术研究进展.计算机研究与发展,2016,53(5):968−987.

[16] 蔺羽佳,尹青,朱晓东.基于域敏感指针分析的细粒度数据随机化技术.计算机应用,2016,36(6):1567−1572.

[71] 林楷,贾春福.基于消息篡改的端信息跳变技术.通信学报,2013,(12):142−148.

[74] 雷程,马多贺,张红旗,杨英杰,王利明.基于网络攻击面自适应转换的移动目标防御技术.计算机学报,2018,41(5):1109−1131. http://kns.cnki. net/kcms/detail/11.1826.TP.20170819.0034.010.html

周余阳(1994－),男,江苏泰州人,博士生,主要研究领域为网络安全,移动目标防御.

郭春生(1994－),男,硕士生,主要研究领域为网络安全.

程光(1973－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为网络空间安全监测和防护,网络大数据分析.

戴冕(1988－),男,博士生,主要研究领域为软件定义网络,数据中心网络,网络测量技术.