拟态防御 DHR 模型若干问题探讨和性能评估

扈红超, 陈福才, 王禛鹏

国家数字交换系统工程技术研究中心 郑州 中国 450002

摘要 针对传统防御技术难以应对未知特征和未知缺陷的攻击, 近年来, 工业界和学术界尝试发展能够“改变游戏规则”的创新性防御技术。网络空间拟态防御(CMD: Cyberspace Mimic Defense)以动态异构冗余(DHR: Dynamical Heterogeneous Redundant)作为核心架构技术。针对信息系统保护的元功能, 采用非相似余度设计方法构造多个功能等价的异构执行体; 在系统运行期间, 动态调度元功能的不同异构执行体在线运行, 以阻断攻击者的攻击过程; 同时, 利用多模判决机制对多个异构执行体的输出结果进行判决, 以检测是否发生攻击。本文针对 DHR 模型的若干问题进行了探讨, 给出了一种理论分析方法, 并进行了实验仿真, 理论分析和仿真结果表明, DHR 能够大幅提升攻击者攻击难度, 增强信息系统的安全性。

关键词 动态异构余度; 动态调度; 异构性; 冗余性
中图法分类号 TN919.21 DOI 号 10.19363/j.cnki.cn10-1380/tn.2016.04.004

Performance Evaluations on DHR for Cyberspace Mimic Defense

HU Hongchao, CHEN Fucai, WANG Zhenpeng

National Digital Switching System Engineering & Technological R&D Center, Zhengzhou 450002, China

Abstract In recent years, both academia and industry have tried to develop innovative defense technologies, since existing defense technologies are difficult to deal with the attacks employing unknown security flaws or backdoors. Starting from analyzing the root causes of security problems in cyberspace, that is, 1) security flaws (holes and the back doors) in information systems are universal; 2) current cyberspace elements are static and homogeneous, as a result, the security flaws can be widely adopted; 3) existing techniques are difficult to check and remove security flaws. Due to this, professor Wu Jiangxing proposed a novel defense framework, namely cyberspace mimic defense (CMD), to defense network attacks employing unknown security flaws by introducing dynamical dissimilarity redundancy mechanism (DHR: dynamical heterogeneous redundant). DHR constructs several functionally equivalent variants for the meta function to be protected, dynamically schedules several variants to run in parallel to block the attacking process. At the same time, it uses multimode decision mechanism to decide which outputs of the running variants are correct and whether attacks have occurred. This paper mainly focuses on the evaluation issue of DHR, and analyzes its performance with a theoretical model. Simulations results show that DHR can significantly improve the security performance of information systems.

Key words Dynamical, heterogeneous and redundant; dynamical scheduling; heterogeneity; redundancy

1 引言

网络技术已广泛渗透到电力、金融、交通等关键基础设施[1]。在促进人类社会进步的同时, 网络及信息系统暴露的安全缺陷成为个人(如黑客)、不法团体利用攻击的对象, 给人类的生产生活带来极大的安全威胁, 网络安全已成为世界各国工业界和学术界共同关注的热点问题[2], 并开展广泛研究。然而相对于网络防护技术的进步, 网络攻击呈现愈演愈烈的态势, 究其原因, 主要有以下几点[2]: 一是网络空间组成要素的基因单一性, 如采用相同的计算架构、硬件、操作系统、软件、网络协议等, 攻击者极易利用挖掘的同一安全缺陷, 针对不同信息系统发起多次攻击; 二是网络组成要素的静态性, 如采用静态 IP 地址、静态端口、静态路由机制等, 攻击者很容易进行探测和持续入侵; 三是尚无找到有效检测安全

通讯作者: 扈红超, 博士, 副研究员, Email:13633833568@139.com。
本课题得到中国博士后基金项目(No.44603)、国家自然科学基金项目(No.61309020)、国家自然科学基金创新研究群体项目(No.61521003)和国家重点研发计划项目(Nos.2016YFB0800100, 2016YFB0800101)资助。
收稿日期: 2016-09-13; 修改日期: 2016-09-23; 定稿日期: 2016-10-20

缺陷的科学方法, 致使多年来网络防护一直延循“亡羊补牢”、“吃药打针”式的技术发展脉络, 从以杀毒软件、防火墙为代表的被动式防护, 发展到以入侵检测[3]、蜜罐[4]、沙箱[5]、入侵容忍[6]为代表的主动式防护。然而, 无论是被动式防护技术还是传统式主动防护技术, 皆以发现攻击者的攻击特征(如病毒特征、行为特征)或被攻击目标的安全缺陷为防护条件, 通过配置策略(如防火墙)、规则(如入侵检测)或打补丁的方式防护, 对于特征未知或缺陷未知的攻击行为束手无策。

针对这一问题, 近年来学术界试图通过创新防御思路, 寻求能够“改变游戏规则”的防护技术, 其中最具典型代表的是美国学术界提出的“移动移动目标防御(MTD: Moving Target Defense)[2]”。MTD 是针对目前攻防双方成本的不对称性, 以及被攻击目标缺乏弹性等问题提出的, 其主要思想有两点: 一是在信息系统配置属性上引入动态性和随机性, 如采用动态 IP[7-10]、动态端口[11]、动态路由[12]、随机化执行代码[13]、随机化地址空间[14]、随机化指令集合[15]和随机化数据[16]等; 二是提高信息系统要素组成的多样性, 如采用多样化编译技术生成同一软件的不同版本[17]。移动目标防御旨在通过引入动态性和多样性, 构建一种动态的、多样的、不确定的运行环境, 降低被攻击对象的可预测性, 从而阻断攻击者攻击过程, 提高攻击者的攻击难度和攻击成本。然而, 移动目标防御技术以主动变化提高攻击者攻击难度为目标, 当攻击者成功入侵系统后, 移动目标防御无法检测。

我们期望, 一个理想的防御系统不仅能够大幅提高攻击者的攻击难度, 还应能够实时检测出成功入侵的攻击行为, 即, 防护技术本身兼具内生的保护与检测双重功能。从网络空间安全问题的本质原因出发, 即, 1)安全缺陷(漏洞或后门)在信息系统中存在的普遍性; 2)网络空间组成要素的单一性和静态性导致安全缺陷易于利用; 3)现有技术手段难以彻底检测并消除安全缺陷, 邬江兴教授提出了网络空间拟态防御的创新思想。拟态防御的典型架构是动态非相似余度(DHR: Dynamical Heterogeneous Redundant)机制, DHR 主要思想有三点: 异构性(Heterogeneous), 即, 对信息系统保护的元功能 F 进行抽象, 并采用非相似余度设计方法构造 M 个功能等价的异构执行体 $\{f_1, f_2, \ldots, f_M\}$; 动态性(Dynamical), 在系统运行期间, 从 M 中动态选择 m(随时间变化)个不同的异构执行体运行, 隐藏信息系统内部的实现结构; 冗余判决(Redundant), 采用多模判决机制对 m 个在线的异构执行体的输出结果进行一致性判决, 检测是否发生了攻击行为。DHR 具有如下特点: 1)通过异构执行体的动态调度阻断攻击者的攻击过程, 能够大幅增加攻击者利用未知安全缺陷的难度; 2)攻击检测和防护不再依赖于攻击特征和安全缺陷的先验知识, 具有内在的威胁检测能力; 3)采用 DHR 能够基于不安全的构件构建相对安全的信息系统。

本文的后续安排如下: 第二部分对与本文思路相关的工作进行归纳和总结, 主要包括移动目标防御技术等; 第三部分从一个简单例子出发给出拟态防御的提出动机; 第四部分给出动态非相似余度的数学模型; 第五部分从理论层面分析其的安全性能; 第六部分对动态非相似余度模型进行理论评估; 第七部分以路由器为对象研究其实现案例; 第八部分讨论了动态非相似余度局限性; 第九部分为总结, 阐述了本文取得的进展及进一步开展的工作。

2 相关工作

本节对近年来出现的典型创新性防御技术“移动目标防御”的主要思想及研究进展, 以及与本文思想相关的机制如非相似余度、多模判决技术进行简要的回顾和总结。

针对移动目标防御的研究工作目前主要集中在建模与性能分析、技术应用等方面。在建模与性能分析方面, 文献[18]提出采用攻击表面模型刻画移动目标防御的有效性, 攻击表面是指攻击者攻击时的可利用系统资源集合, 而移动目标防御正是通过自动地改变系统的配置属性, 动态地改变暴露给攻击者的攻击表面, 使其无法预测, 从而达到提高系统的安全性的目的, 作者在该文中对移动攻击表面问题进行了形式化建模, 并提出一种量化移动性的方法, 并在安全性与可用性之间进行折中, 同时提出一种双方随机博弈模型来确定最优的防御策略。Zhuang R.在移动目标防御理论建模方面做了大量的工作, 文献[19]针对企业网络环境, 提出一种基于 Markov 转移概率的移动目标防御有效性分析模型。针对目前移动目标防御研究处于起步阶段, 包括缺乏标准的定义、攻击表面的具体含义、度量该类系统有效性的标准和理论模型等, 作者尝试为移动目标防御建立一套理论框架: 第一步, 建立 MTD 系统的理论, 该理论仅关注系统本身, 以及系统如何随时间的推移不断调整以达到其整体目标[20]; 第二步, 建立攻击者理论, 该理论描述攻击者的目标, 以及达到目标所采取的动作[21]; 最后, 综合前两步成果

建立整体的 MTD 理论, 其目的是定义 MTD 系统理论和攻击者理论的组成要素之间如何相互交互, 该项工作目前还在进行之中。另外一类研究思路是采用博弈论来评估移动目标防御技术, 如文献[22]采用博弈论方法确定动态平台防御的最优策略, 将攻击者和防御者之间的交互建模为不完整的“领导-追随者”两者博弈。和现有的基于博弈论的研究不同, 文献[23]定义了通用的网络防御场景, 并利用经验博弈论方法, 采用系统性的仿真验证攻击者和防御策略之间的相互作用。[24]采用分级攻击表示模型(HARM: Hierarchical Attack Representation Model)评估移动目标防御技术的有效性, 并比较了部署移动目标防御技术时攻击图和HARM的扩展性。文献[25]从机密性、完整性和可用性三个方面对移动目标防御技术进行了研究, 并探索了在 MTD 系统中需要在机密性和可用性之间进行折衷。文献[26]将网络传播动力学(Cyber Epidemic Dynamics)模型理论引入到移动目标防御的安全性能评估中, 以期取得可量化安全性能指标。

移动目标防御技术的应用可以划分为三类: 基于终端的、基于网络的和基于云的。基于终端的移动目标防御技术又可分为数据级、指令级、代码级、内存级和应用级, 数据级常见技术为动态改变数据在内存中的存储方式, 如基于内存对象类别将数据和随机掩码执行“异或”运算; 指令级采用指令随机化(ISR: Instruction Set Randomization)技术改变每个进程的指令[27,28], 如对进程指令进行加密, 运行时进行解密, 以避免攻击者注入恶意指令; 代码级采用的是代码随机化(Code randomization)技术[31-38], 如利用编译技术、二进制重写技术等进行代码随机化, 防止代码重用攻击; 内存级使用的是地址空间布局随机化技术(ASLR : Address Space Layout Randomization)[39,40], 如在编译或运行时随机改变程序的内存布局, 对抗代码注入攻击; 典型的应用级技术是软件多样化[41], 即针对同一功能的构建 M 个软件版本; 另外一种技术是组合加密技术[42,43], 该技术将密钥切分为多片, 避免单一密钥被窃取或损坏。基于网络的移动目标防御技术的一种典型方法是将网络设备的静态配置属性动态化, 如文献[7]提出的MT6D(Moving Target IPv6 Defense)基于 IPv6 巨大的地址空间, 在不影响通信双方会话连接的条件下, 透明和动态地轮转网络和传输层的地址, 使攻击者难以锁定和入侵被攻击目标, 在提高 IP 地址跳变的不可预测性、自适应性的同时控制开销, Jafarian J.H. 提出的 RHM(Random Host Address Mutation)采用基于地址空间和 IP 地址的分层跳变方案[9], 在此基础上, 又在文献[10]中提出了主动自适应地址跳变技术, 该技术实时监控攻击者行为, 基于此动态地执行地址跳变; 文献[8]等研究了基于软件定义网络架构下的 IP 地址和路由的动态集中式控制问题; 另外, 文献[11]和[44]分别提出了基于端口跳变和虚拟网影射跳变的移动目标防御技术。基于云的移动目标防御技术的典型应用是利用虚拟机迁移, 解决云资源虚拟化后利用多用户共存发起的旁路攻击[45], 另外一种基于“快照、存储、恢复”的方法将任务在多个虚拟机之间动态切换执行, 如文献[46]提出了基于概率 MTD 的虚拟机部署策略。

3 数学模型

3.1 控制器威胁启迪

在软件定义网络中, 控制平面生成流表等配置信息, 并通过南向接口如 Openflow 下发到数据平面, 数据平面依据控制器下发的配置对数据流进行转发等处理; 另外, 控制平面还需要为用户提供应用开发和部署的北向接口。因此, 若恶意交换机接入到控制器或恶意应用部署到控制器上, 攻击者便可以对控制器进行劫持, 从而对 SDN 网络发起任意攻击。一种典型的攻击方式是攻击者劫持控制器后, 操纵流表将特定数据流转发到恶意转发设备或者是接收者进行窃听。以图 1 为例, 正常情况下发送方 *Sender* 的所有数据流经由交换机 $\langle SW1, SW2\rangle$ 传输给 *Receiver*, 若攻击者 *Attacker* 通过控制交换机 $SW3$ 劫持了控制器 *Controller*, 并在控制器上增加了一条表项 $\langle *, IP1, IP2, 25, Group\rangle$ (*Group* 表结构如图 1 所示); *Controller* 将该表项下发到交换机 $SW1$; 随后, $SW1$ 接收到 *Sender* 发送到 *Receiver* 的所有端口为 25(邮件系统知名端口)的数据分组后, 首先将分组发送到 $SW2$, 再将分组的目的地址修改为 *Attacker* 的目的 $IP3$, 并发送到 $SW3$。

针对这些问题, 我们尝试构建了**图 2** 所示的拟态控制器架构, 其主要特点是: 一是异构性, 和飞机的飞控系统类似[47,48], 采用了多个(可扩展到 N 个)功能等价、实现相异的 SDN 控制器实例, 如运行环境采用不同操作系统和处理器等; 二是动态性, 调度器动态地从控制平面中选择 m(1,2,3)个作为参考依据; 三是判决器, 判决器将调度器选择的 SDN 的下发流表作为输入进行判决, 并选择一个作为可信路由下发到数据平面。通过实验我们发现该架构有以下优点: 一是异构冗余机制增加了控制器的可靠性, 若一个控制器在系统运行期间发生故障, 其他控制器可以快速切换; 二是提高了攻击者攻击难度, 由于

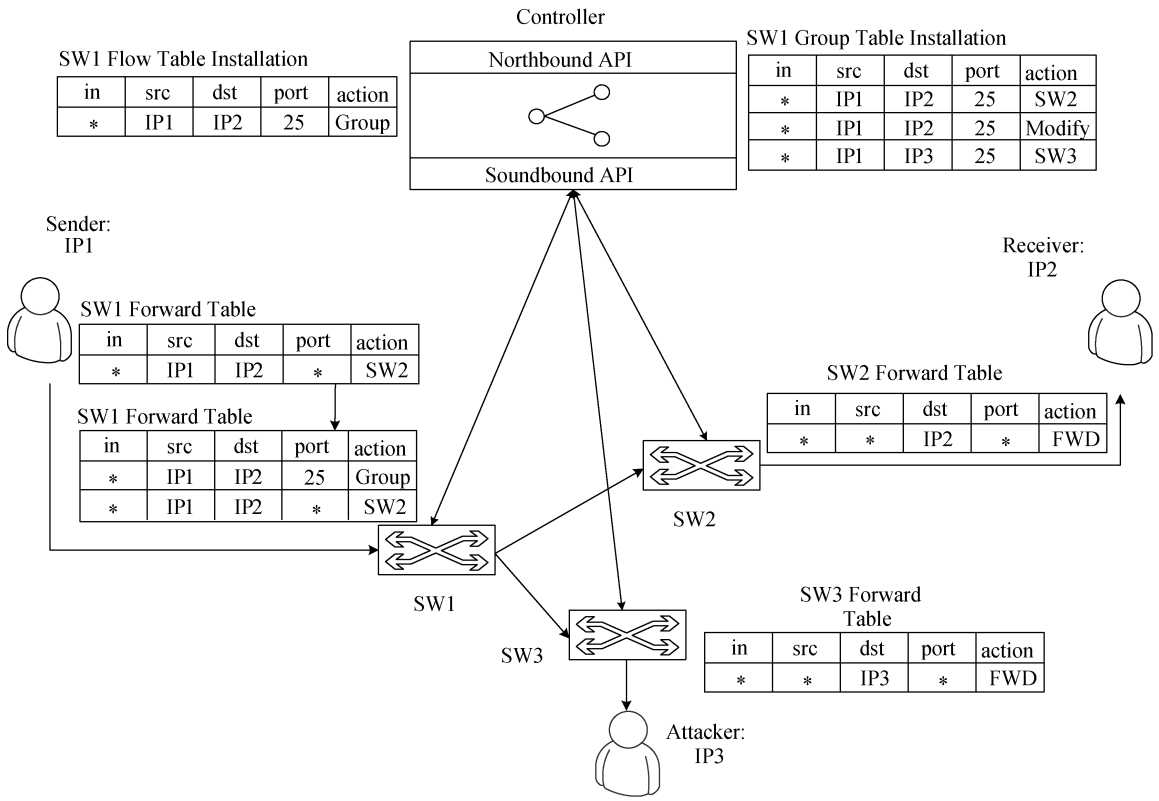


图 1　控制器劫持示例

动态调度器不断变换当前有效的控制器, 攻击者掌握的后门呈现为线上/线下不断切换, 难以持续利用; 三是即使攻击者攻击成功 Ctroller1, 由于判决器的存在, 被加入的流表通过 Ctroller1 下发时也可检测出来(除非攻击者在周期 T 内同时入侵了所有控制器, 无疑这对攻击者是极其困难的)。

3.2　符号定义

为便于后续描述, 首先给出如表 1 中的符号定义。

表 1　本文用到的符号定义

符号	定义说明
i、j、k、l、m、n	本文使用的通用整数变量
F	网络操作系统功能函数
$f(\Lambda, \phi)$	元功能的一种实现实例, 由实现结构(类似于数据结构, 包括硬件架构如 CPU 架构、FPGA 架构、GPU 架构、虚拟机等)和实现算法 (如采用的操作系统、开发工具、语言, 以及实现算法)决定, 简称统一称为异构执行体。
$\mathbf{F}=\{f_j \mid 1\le j\le M\}$	元功能的实例集合, 类似于类的对象集合, 其中 f_j 为某一元功能第 j 个实例
$H \perp \{F_i \mid 1\le i\le N\}$	防护链抽象模型 H, 由元功能集合定义, F_i 为系统第 i 个元功能
$s=\{s_1, s_2, \ldots, s_i, \ldots, s_N\}$	防护链 H 的状态向量, $\boldsymbol{s}_i$ 为元功能 F_i 的执行体选择指示向量。
T	拟态变换周期
R	输入代理
X	执行体编排算法
A	输出判决器

3.3　基本模型

首先考察单个元功能的拟态防御理论模型, 如图 3 所示, 主要由四部分组成: 异构执行体集合 **F**、输入分发器 R、拟态变换器 X 和输出判决器 A 组成, 详细讨论如下。

3.3.1　异构执行体

$\mathbf{F}=\{f_j \mid 1\le j\le M\}$ 表示元功能 F 的异构执行体集合, 用“异构性”和“多样性”来刻画。其中, **“异构性”**是指元功能 $\mathcal{F}$ 存在功能相同、结构相异的多个实例, 即, 对于任意两个实例 f_i、$f_j \in \mathbf{F}$, 满足 $\Lambda_i \ne \Lambda_j$或$\phi_i \ne \phi_j$; 对于正交相异性, 满足 $\Lambda_i \ne \Lambda_j$且$\phi_i \ne \phi_j$。相异性设计的理论和方法是学术界和产业界长期以来关注的重要课题, 最早应用于波音和空客民用客机的飞控系统[47,48]。本文的重点在于如何利用异构性提高信息系统安全性, 对相异性设计的相关问题不再赘述, 后续工作会进一步探讨, 这里仅给出决定信息系统/网络异构性的几个关键属性: 如处理器(如 CPU、FPGA、GPU、DSP 等)、操作系统、编译器、语言、算法、数据结构、代码风格、数据库等。

“多样性”是指 **F** 中异构执行体的种类数 M, M 越大, 可选择的异构执行体组合越多(理论上为 $M!$), 攻击者需要探测的空间越大, 攻击的难度越高。

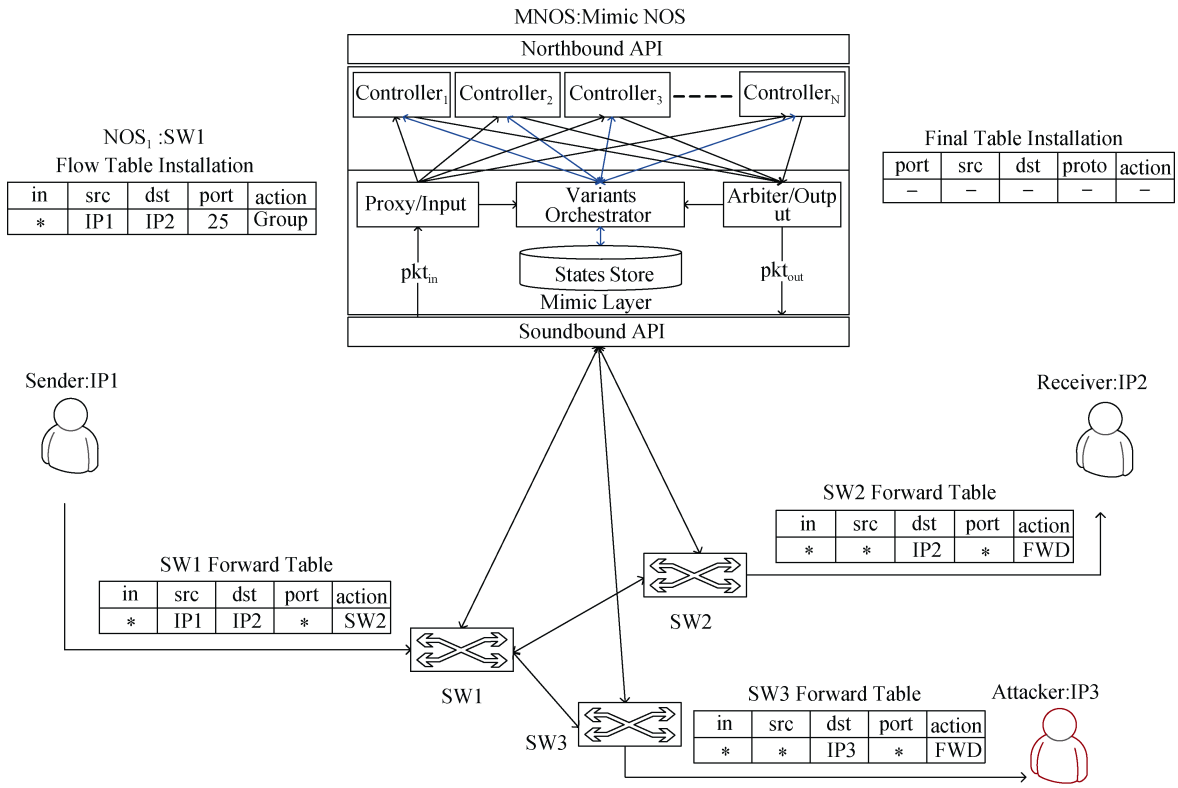


图 2　拟态控制器架构

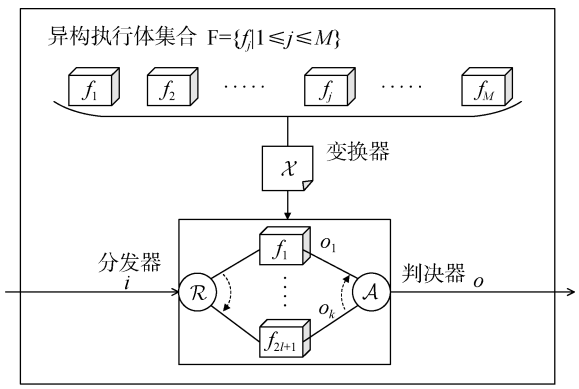


图 3　动态非相似余度模型

3.3.2　分发器

在任意时刻 t, 分发器 R 将输入分发到选择的 m 个执行体 $\boldsymbol{\Phi}_t$ 上, 每个执行体的输入完全一致, 独立执行计算, 对于功能等价的异构执行体, 相同输入必产生相同输出。

3.3.3　拟态变换器

拟态变换器 X (为便于描述, 拟态变换器和拟态变换算法均用 X 表示)实现异构执行体的调度, 同移动目标防御, 拟态变换具有“动态性”: 在系统变换时, 依据当前时间 t、状态 s 和选择的 m 个异构执行体集合 $\boldsymbol{\Phi}_t$, 从 $\mathbf{F}$ 中选取 m'个作为$(t, t+T)$时间区间运行的执行体, 即 $\boldsymbol{\Phi}_{t+T} = X\left(t, s, \boldsymbol{\Phi}_t\right)$。此外, 为确保判决器判决的准确性, 任意时刻 t 运行的异构执行体个数满足: $\left|\boldsymbol{\Phi}_t\right| \geq 2l+1 (l \geq 1)$。这样, 在观察者看来, 受保护元功能的物理实现结构、算法对外呈现出不确定性, 难以获得一致、确定性的视图, 不仅攻击者可利用的漏洞和后门也呈现不确定性, 而且攻击者的攻击链难以保持连续性和完整性。在具体实现上, 拟态变换器必须遵循两个基本原则: 一是执行体切换期间保持系统内外部状态的一致性; 二是最大化对外呈现不确定性。下面本文针对这两个问题详细讨论。

问题 1: 异构执行体动态切换时状态的一致性

执行体的状态通常包含运行状态和数据两部分。本文设计了**图** 4 所示的框架保证执行体切换时状态的一致性。在该框架中, 在变换控制器的控制下, 变换器可以: 1)接收来自外部的切换指令, 实现按照系统预先设定的逻辑进行切换; 2)根据安全态势感知传感器(如入侵检测系统)的输入进行切换; 3)根据判决器的输入结果进行切换。变换器准备切换时, 首先计算出下一时间周期内选择的异构执行体集合 $\boldsymbol{\Phi}_{t+T} = X\left(t, s, \boldsymbol{\Phi}_t\right)$, 并调用执行体容器。执行体容器向资源控制器申请资源(如虚拟机、内存等)成功后, 按照执行体模板进行实例化, 并将执行体状态池的状态数据同步到实例化的执行体, 待准备就绪后, 变换器向切换管理器请求切换。切换管理器首先将

执行体 $\boldsymbol{\Phi}_{t+T}$ 加入到运行集合中, 即 $\boldsymbol{\Phi}_{t^+}=\boldsymbol{\Phi}_t\cup\boldsymbol{\Phi}_{t+T}$, 并作为判决器的输入; 其次, 将分发器的数据从 $\boldsymbol{\Phi}_t$ 转移到 $\boldsymbol{\Phi}_{t+T}$ 上; 最后将 $\boldsymbol{\Phi}_t$ 从运行集合 $\boldsymbol{\Phi}_{t^+}$ 中移出, 并更改判决器判决集合, 同时释放执行体资源。

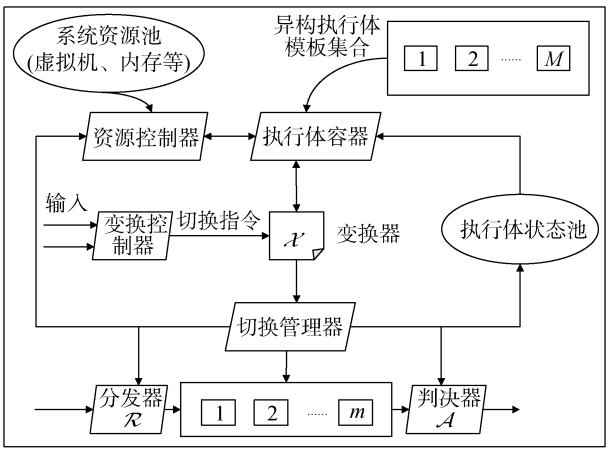


图 4　异构执行体切换示意图

问题 2: 如何最大化对外呈现不确定性

从信息论角度来看, 信源对外呈现的不确定大小或随机性可以用信息熵描述[50]。若将变换器看作离散信源、异构执行体选择状态看作事件的话, 则变换器信息熵可描述为: $H(X)=-\sum_{j=1}^{M}p_j\log(p_j)$, 其中, p_j 为统计意义上异构执行体 f_j 的出现概率。因此, 异构执行体对外呈现不确定性的大小转变为信息熵的大小, 求解最大化对外呈现不确定性问题变为求解最大化信息熵:

$$H_{\max}(X)=\max\left\{-\sum_{j=1}^{M}p_j\log(p_j)\right\}$$

由信息论相关知识可知, 由 M 个异构执行体构成的变换器, 当每个异构执行体出现的概率均相等且等于 $1/M$ 时, $H(X)$ 达到最大值, 即, $H_{\max}(X)=\log M$。

因而, 从异构执行体个数 M 越大, 变换器信息熵越大, 对外呈现的不确定性越大; 此外, 当且仅当每个控制器出现的概率相同时(前提是每个控制器必须保证完全异构), 系统的不确定性最大, 在设计实际的调度器应依据这一原则。

3.3.4　判决器

判决器对执行体集合 $\boldsymbol{\Phi}_t$ 的运行结果进行裁决, 并反馈到变换器 X, 变换器触发系统重构对异构体进行清洗。在入侵容忍系统中, 学者们对判决器的判决原则进行了大量的研究, 本文不再赘述, 重点给出一种基于执行体可信度的判决原则WTA(Weighted trustiness based Arbitrament), 具体如下: 在系统运行开始, 根据历史经验(执行体已暴露的漏洞和后门数量), 以及异构执行体硬件架构、运行环境和来源(自主设计、外部供给)等赋予可信度权值 $\{\omega_1,\omega_2,\cdots,\omega_M\}$; 若时刻 t 有 $2l+1$ 个执行体运行, 首先将输出结果一致的执行体划分为一个组 $\boldsymbol{G}_k$, 便构成了集合序列 $\{\boldsymbol{G}_1,\boldsymbol{G}_2,\cdots,\boldsymbol{G}_k,\cdots\}$, 即

$$\forall f_i、f_j\in\boldsymbol{G}_k,o_i=o_j,\boldsymbol{G}_k\subseteq\boldsymbol{\Phi}_t,且\sum|\boldsymbol{G}_k|=2l+1;$$

其次, 计算每个集合 $\boldsymbol{G}_k$ 的置信度 W_k:

$$W_k=\sum\omega_n,\ 其中,\ f_n\in\boldsymbol{G}_k。$$

最后, 选取置信度最大的集合作为输出结果, 如下式所示, 同时发现异常的执行体。此外, 在每次拟态变换时, 系统根据该周期内执行表现更新异构执行体的置信度。需要说明的是, 若存在两个集合 $\boldsymbol{G}_i$ 和 $\boldsymbol{G}_j$ 的置信度相同, 即, $W_i=W_j$, 则随机选择一个集合作为结果输出的参考集合。

$$o=o_m,\ \text{where}\ f_m\in\boldsymbol{G}_{k_{\max}}\ \text{and}\ k_{\max}=\max_k(W_k)。$$

可以看出, 攻击者若想攻击成功, 必须使得判决器选择错误的输出集合, 即, 错误结果对应的集合的置信度最大, 这也是拟态防御机制失效的条件, 本文后续会详细讨论。

3.4　通用模型

上面探讨了单个元功能的拟态防御模型, 实际的信息系统通常是分层分级的, 如典型的信息系统由硬件层、操作系统层、应用层等组成; 而网络结构同样也是分层分级的。为实现攻击目的, 攻击者需要经历多个中间节点。基于此特性, 下面探讨信息系统或网络的多级拟态防护链模型。假设攻击者到达元功能 F_N 之前需攻击 N-1 个元功能 $F_1,F_2,...,F_{N-1}$, 本文将攻击者达到攻击目标所经历的 N 个元功能所构成的序列称为防护链, 用 $H\perp\{F_1,F_2,...,F_N\}$ 表示。

分别用 $\mathbf{X}=\{X_1,X_2,\ldots,X_i,\ldots,X_N\}$ 和 $\boldsymbol{s}=\{\boldsymbol{s}_1,\boldsymbol{s}_2,\ldots,\boldsymbol{s}_i,\ldots,\boldsymbol{s}_N\}$ 表示 H 中元功能的拟态变换器序列和异构执行体的选择状态序列, 其中, $\boldsymbol{s}_i$ 为元功能 F_i 的执行体选择指示向量。若将整个防护链看作信源的话, 则根据前述讨论可知, 信源对外呈现的不确定性越大, 信源的信息熵最大, 攻击者的攻击难度也就越大, 由此可以得到如下定理:

定理 1: 由 N 个元功能构成的防御链 H, 当且仅当元功能变换器两两相互独立时, H 获得最大信息熵, 且 $H(\mathbf{X})=\sum_{i=1}^{N}\log M_i$, 其中, M_i 为第 i 个元功能的执行体数。

证明: 首先证明 N=2 的情况, 假设元功能 F_1 和 F_2 的变换器 X_1 和 X_2 有关联, 异构执行体集合分别为 $\mathbf{F}=\{f_j\,|\,1\le j\le M_1\}$ 的 $\mathbf{G}=\{g_i\,|\,1\le i\le M_2\}$, 对应的概率分布函数分别为 $\mathbf{P}=\left(p_1,p_2,...,p_{M_1}\right)$ 和 $\mathbf{Q}=\left(q_1,q_2,...,q_{M_2}\right)$, 相关性用条件概率 $P\left(X_2=g_i\,|\,\mathrm{X}_1=f_j\right)=p_{ij}$ 描述, 则

$$
\begin{aligned}
&H_{M_1M_2}\left(\begin{array}{l}p_1p_{11},p_1p_{12},\cdots,p_1p_{1M_2},p_2p_{21},p_2p_{22},\cdots,\\ p_2p_{2M_2},\cdots,p_{M_1}p_{M_1}1,p_{M_1}p_{M_1}2,\cdots,p_{M_1}p_{M_1M_2}\end{array}\right)\\
&=-\sum_{i=1}^{M_1}\sum_{j=1}^{M_2}p_ip_{ij}\log p_ip_{ij}\\
&=-\sum_{i=1}^{M_1}\sum_{j=1}^{M_2}p_ip_{ij}\log p_i-\sum_{i=1}^{M_1}\sum_{j=1}^{M_2}p_ip_{ij}\log p_{ij}\\
&=-\sum_{i=1}^{M_1}\left(\sum_{j=1}^{M_2}p_{ij}\right)p_i\log p_i-\sum_{i=1}^{M_1}p_i\sum_{j=1}^{M_2}p_{ij}\log p_{ij}\\
&=-\sum_{i=1}^{M_1}p_i\log p_i+\sum_{i=1}^{M_1}p_i\left(-\sum_{j=1}^{M_2}p_{ij}\log p_{ij}\right)
\end{aligned}
$$

现考察 $\sum_{i=1}^{M_1}p_i\left(-\sum_{j=1}^{M_2}p_{ij}\log p_{ij}\right)$, 可知,

$$
\begin{aligned}
\sum_{i=1}^{M_1}p_i\left(-\sum_{j=1}^{M_2}p_{ij}\log p_{ij}\right)&\le\sum_{i=1}^{M_1}p_i\left(\log\sum_{j=1}^{M_2}p_{ij}\frac{1}{p_{ij}}\right)\\
&=\sum_{i=1}^{M_1}p_i\left(\log M_2\right)\\
&=\log M_2
\end{aligned}
$$

当且仅当 $p_{ij}=\dfrac{1}{M_2}$ 时取得最大值, 此时 $p_{ij}=p_j$, 因此 X_1 和 X_2 相互独立; 可以用归纳法证明 N 为任意整数时, 只有两两相互独立时, 取得最大值 $H_{\max}(\mathbf{X})=\sum_{i=1}^{N}\log M_i$。

4 理论分析

4.1 若干定义

为便于后续分析, 首先给出与理论分析相关的如下若干定义。

定义 1: 攻击能力 用攻击者攻击成功的概率密度函数来描述, 记为 $\upsilon(t)$, 则攻击者在 t 时间内攻击成功的概率记为 $p=\int_t\upsilon(t)dt$。若攻击者对同一目标依次发起了 n 次攻击, 每次攻击的持续时长均为 t, 攻击成功的概率分别为 $p_1p_2,\ldots,p_n$, 若 $\forall i、j,\Rightarrow p_i=p_j$, 且与时间无关, 则称攻击者的攻击能力是无记忆的, 否则, 为有记忆的; 若在任意的攻击周期内, $\upsilon(t)$ 和时间无关, 即 $\forall t_1\ne t_2,\Rightarrow\upsilon(t_1)=\upsilon(t_2)$, 则称攻击者攻击能力是非时变的, 否则为时变的。

从物理意义上讲, $\upsilon(t)$ 描述了攻击成功概率的变化速率, 即单位时间内攻击成功概率的分布, 本文选取“凸”型概率密度函数来刻画攻击者的攻击能力, 即,

$$\upsilon(t)=\lambda e^{-\lambda t},$$

其中, λ 均为常量。

在该概率密度函数下, 攻击者在开始的一段时间内攻击能力迅速积累, 但随着时间的增长, 攻击能力增长变缓, 但成功概率逼近于 1, 如图 5 所示。

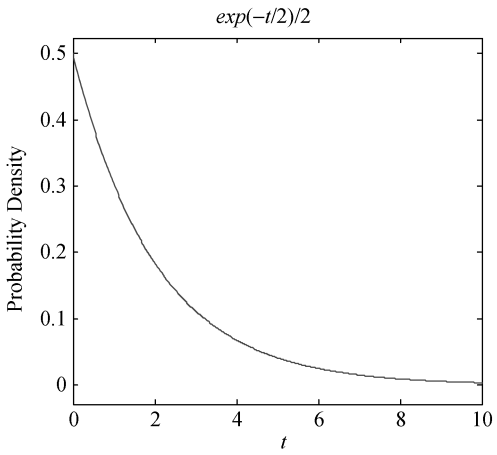


图 5 “凸”型概率密度函数

定义 2: 攻击成功 对于非相似性余度系统, 若不考虑非攻击因素导致的执行体失效问题, “攻击者攻击成功”等价于“判决器判决法则失效”。根据“3.3.4”中所述, “判决器判决法则失效”等价于“输出一致错误的执行体个数大于输出一致正确的执行体个数”, 即:

$$p_{attack\ success}=1-P\left(\forall\boldsymbol{G}_k\ne\boldsymbol{G}_{correct},W_k<W_{correct}\right)\quad(1)$$

其中, $\boldsymbol{G}_{correct}$ 为输出正确的执行体集合。

通常, 由于各异构执行体漏洞和后门的差异性(采用了不同硬件平台、不同开发工具、不同操作系统、不同数据结构和算法等因素), 即使攻击者成功入侵了不同的异构执行体, 尚若没有使各执行体产

生一致的错误输出, 从判决器判决的角度来看, 依旧攻击不成功, 即:

$$\begin{aligned} p_{attack\ success} &= P(attack\ success) \\ &\le P(intrusion\ success) \\ &= p_{intrusion\ success} \end{aligned} \tag{2}$$

为便于分析, 本文约定 $p_{attack\ success} = p_{intrusion\ success}$, 即攻击者入侵成功等同于攻击成功, 在后续的工作中再分析 $p_{attack\ success} \neq p_{intrusion\ success}$ 的情况。

定义 3: 安全增益 对于防御者而言, 攻击者攻击不成功的概率反映了其安全性能, 因此, 本文用在采用和不采用动态非相似性余度模型下, 攻击者攻击不成功的概率的比值来刻画安全增益, 若攻击者对应的攻击成功的概率分别记为 p' 和 p, 则安全增益 Δ 定义为:

$$\Delta = \frac{1-p'}{1-p} \tag{3}$$

定义 4: 收益投资比 由于采用动态非相似性余度模型时需要多个异构体并行执行元功能, 并随着时间的推移需要对异构执行体动态调度, 因此, 系统资源开销要大于不采用动态非相似性余度模型的系统, 分别用 R 和 R' 代表前后系统资源代价, 则收益投资 W 定义为:

$$W = \frac{\Delta}{R'/R} = \frac{R(1-p')}{R'(1-p)} \tag{4}$$

本节对从安全增益和投资收益比两个方面对拟态防御架构的安全性进行分析。为便于分析, 首先给出如下的定义和假设条件。

4.2 假设条件

假设 1: 攻击者预先不知道被攻击节点的拟态变换算法 $\mathcal{X}$, 同样不知道异构执行体的实现结构 Λ 和实现算法 ϕ, 必须在攻击过程中通过探测才能获取。

假设 2: 为便于分析, 假设攻击者的攻击是无记忆的, 即, 若攻击者在相同条件下先后针对 $\mathcal{F}$ 独立发起了 n 次攻击, 每次持续时长为 t, 攻击成功的概率密度为 $\upsilon(t)$, 第 i 次攻击成功的概率为 $P_i(t) = \int_0^t \upsilon(t)dt$, 那么 $\forall i \neq j,$, 满足:

$P_i(t) = P_j(t)$。

假设 3: 为便于分析, 假设攻击者对相同元功能的异构执行体攻击成功的概率密度函数均相同, 即对于 $\forall f_i、f_j \in \mathbf{F}$, $\upsilon_i(t) = \upsilon_j(t)$。

假设 4: 同样为便于分析, 假设判决器采用了大数判决原则, 即每个异构执行体的可信度权值满足: $\forall i \neq j, \omega_i = \omega_j$, 且任意时刻有 $2l+1$ 个异构执行体运行。

4.3 场景设定

本节尝试系统在采用动态非相似余度模型下, 量化其安全增益和收益投资比, 并尝试给出动态非相似余度理论模型的性能界。为便于对比, 假设除设定的防御机制外, 被保护系统没有采取其他的安全措施, 如入侵检测、防火墙等。本文分析单节点情形, 在该情形下, 假设攻击者攻击成功的概率密度函数是时变的, 被攻击执行体的攻击状态是不可累积的, 且执行体在拟态变换时执行清洗, 攻击者必须在周期 T 内攻击成功 l+1 个执行体才能攻击成功。

4.4 评估模型

设 M 个执行体被攻击成功的概率密度函数分别为: $\upsilon_1(t), \upsilon_2(t), \cdots, \upsilon_N(t)$, 下面求解单位时间 t 内攻击者攻击成功的概率 P_1。

假设攻击者同一时刻只能攻击一个异构执行体, 在任意攻击周期 T 内, 随机从 $2l+1$ 个异构执行体中选取 $l+1$ 个作为攻击对象, 该 $l+1$ 执行体被攻击成功的密度函数分别为 $\upsilon_1'(t), \upsilon_2'(t), ..., \upsilon_{l+1}'(t)$, 被攻击成功所需时间分别为 $X_1', X_2', \cdots, X_{l+1}'$, 则在 t 时间内被攻击成功的概率为:

$$\begin{aligned} &P(X_1' + X_2' + \cdots + X_{l+1}' \le t) \\ &= \int_0^t \upsilon_1'(t_1) \cdot P(X_2' + X_3' + \cdots + X_{l+1}' \le t - t_1) dt_1 \\ &= \int_0^t \upsilon_1'(t_1) \cdot \int_0^{t-t_1} \upsilon_2'(t_2) \cdot P(X_3' + X_4' + \cdots + X_{l+1}' \le t - t_1 - t_2) dt_2 dt_1 \\ &= \cdots \\ &= \int_0^t \int_0^{t-t_1} \cdots \int_0^{t-t_1-t_2-\ldots-t_l} \upsilon_{l+1}'(t_{l+1}) \cdots \upsilon_2'(t_2) \cdot \upsilon_1'(t_1) \cdot dt_1 dt_2 \ldots dt_{l+1}. \end{aligned} \tag{5}$$

用 $F_1(t)$ 表示在一个拟态变换周期 T 内攻击成功的概率分布, 则需考虑从 M 个异构执行体集合中选取(*l*+1)个执行体的所有组合情况, 则,

$$F_1(t)=\sum P(X_1'+X_2'+\cdots+X_{l+1}'\le t)\cdot\frac{1}{C_M^{2l+1}\cdot C_{2l+1}^{l+1}}$$
$$=\frac{1}{C_M^{2l+1}\cdot C_{2l+1}^{l+1}}\cdot C_{M-(l+1)}^{l}\cdot\left(\begin{array}{l}\int_0^t\int_0^{t-t_1}\cdots\int_0^{t-t_1-t_2-\ldots-t_l}\upsilon_{l+1}(t_{l+1})\cdots\upsilon_2(t_2)\cdot\upsilon_1(t_1)\cdot dt_1dt_2\ldots dt_{l+1}+\\ \int_0^t\int_0^{t-t_2}\cdots\int_0^{t-t_2-t_3-\ldots-t_{l+1}}\upsilon_{l+2}(t_{l+2})\cdots\upsilon_3(t_3)\cdot\upsilon_2(t_2)\cdot dt_2dt_3\ldots dt_{l+2}+\\ \cdots\end{array}\right) \tag{6}$$
$$=\frac{1}{C_M^{l+1}}\cdot\left(\begin{array}{l}\int_0^t\int_0^{t-t_1}\cdots\int_0^{t-t_1-t_2-\ldots-t_l}\upsilon_{l+1}(t_{l+1})\cdots\upsilon_2(t_2)\cdot\upsilon_1(t_1)\cdot dt_1dt_2\ldots dt_{l+1}+\\ \int_0^t\int_0^{t-t_2}\cdots\int_0^{t-t_2-t_3-\ldots-t_{l+1}}\upsilon_{l+2}(t_{l+2})\cdots\upsilon_3(t_3)\cdot\upsilon_2(t_2)\cdot dt_2dt_3\ldots dt_{l+2}+\\ \cdots\end{array}\right).$$

若每一个异构执行体被攻击成功的概率密度函数均为 $\upsilon(t)$，则，

$F_1(t)=\int_0^t\int_0^{t-t_1}\cdots\int_0^{t-t_1-t_2-\ldots-t_l}\upsilon(t_{l+1})\cdots\upsilon(t_2)\cdot\upsilon(t_1)\cdot dt_1dt_2\ldots dt_{l+1}$。

分别用 P_T、$\overline{P}_T$ 表示在一个变换周期 T 内攻击成功与攻击不成功的概率，由式(6)可知，$P_T=F_1(T)$，$\overline{P}_T=1-F_1(T)$。则定义在时间 t 上的概率分布函数 $F(t)$ 求解如下:

① $0<t\le T$ 时

$F(t)=F_1(t)$

② $T<t\le 2T$ 时

$F(t)=P_T+F_1(t-T)\cdot\overline{P}_T$

③ $2T<t\le 3T$ 时

$$F(t)=P_T+F_1(2T-T)\cdot\overline{P}_T+F_1(t-2T)\cdot(\overline{P}_T)^2$$
$$=P_T\cdot(1+\overline{P}_T)+F_1(t-2T)\cdot(\overline{P}_T)^2$$

④ $3T<t\le 4T$ 时

$$F(t)=P_T(1+\overline{P}_T)+F_1(3T-2T)\cdot(\overline{P}_T)^2+F_1(t-3T)\cdot\overline{P}_T{}^3$$
$$=P_T\cdot(1+\overline{P}_T+\overline{P}_T{}^2)+F_1(t-3T)\cdot\overline{P}_T{}^3$$

类推可知:

当 $nT<t\le(n+1)T$ 时

$$F(t)=P_T\cdot(1+\overline{P}_T+\overline{P}_T{}^2+\cdots+\overline{P}_T{}^{n-1})+F_1(t-nT)\cdot\overline{P}_T{}^n$$。

由定义 3 和定义 4 可知，安全增益为:

$$\Delta=\frac{1-p'}{1-p}=\frac{1-F(t)}{1-\int_0^t\upsilon(t)dt}$$

收益投资比为:

$$W=\frac{R(1-p')}{R'(1-p)}=\frac{1-F(t)}{N\cdot\left(1-\int_0^t\upsilon(t)dt\right)}$$

5 仿真实验

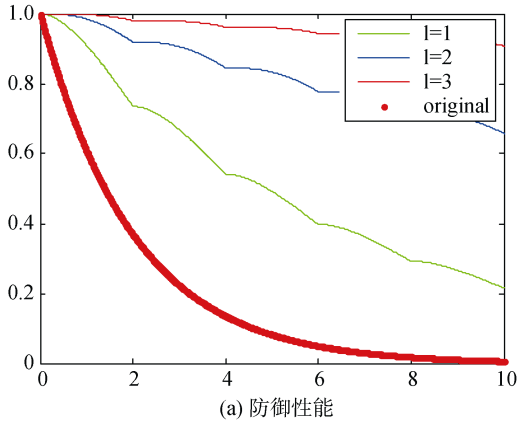
本节基于第 4 节的分析结果，给出在“I)单节点模型和 II)防护链模型”两场景下拟态防御的安全性能的数值解。试验环境采用的是 Lenovo ThinkCentre 工作站，处理器为 i7-4700@3.60GHz，内存为 8.0GB，操作系统为 Windows7 64 位版本，编程环境为 Matlab2013b。从 4.4 节可以看出，攻击者在时间 t 内攻击成功的概率和异构执行体的个数 M、在线运行的执行体个数 $2l$+1、拟态变换周期 T 和攻击者的概率密度函数有关。试验过程如下：首先计算在 T 取固定值，l 取不同值时系统的防御性能；其次，计算在 l 取固定值，T 取不同值时系统的防御性能。

图 7.(a)、7.(b)和 7.(c)分别给出了攻击成功概率密度函数为指数函数 $\upsilon(t)=\lambda e^{-\lambda t}$ (取 λ=0.5)、拟态变换周期 T=2、l=1、2、3 时，系统的防御性能、安全增益和收益投资比随 l 的变化曲线。可以看出，防御性能以及安全增益随异构执行体的个数增加而增加，这与预期结果一致；然而收益投资比则不一定，如 l=3 时的收益投资比高于 l=1,却低于 l=2 时。

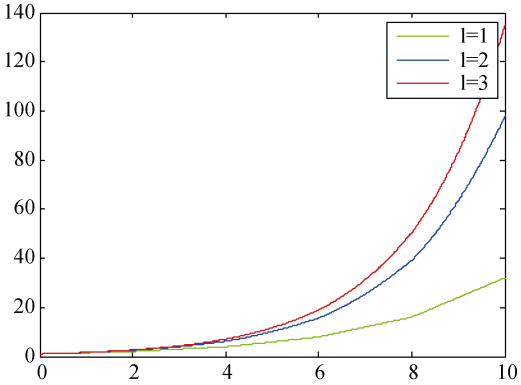
图 8.(a)、8.(b)和 8.(c)分别给出了攻击成功概率密度函数为指数函数 $\upsilon(t)=\lambda e^{-\lambda t}$ (取 λ=0.5)，l 分别为 1、2、3，T 的取值为 1, 2, 3 对系统的防御性能的影响。从图中可以看出，T 取值越小，则系统防御性能越好；从该图也可以看出，系统防御性能随 l 增加而增加。

6 局限性讨论

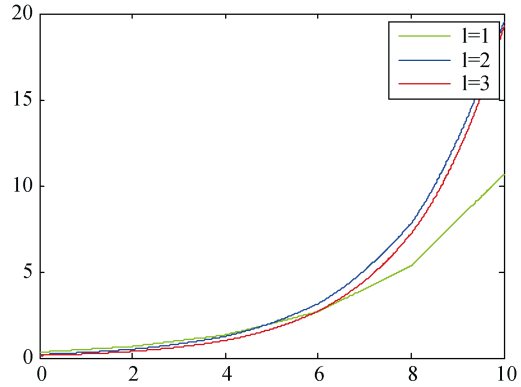
可以看出，拟态防御的实施前提是所保护的元功能必须具有输出，即对于相同的输入，必须产生一致的可预测的输出结果，这样多个异构执行体的执行结果才能进行比较，以判决被保护的元功能是非被攻击，对于没有一致的输出结果的元功能，无法应用拟态防御思想；另外，拟态防御无法应对针对资源有限性发起的攻击，如 DDoS(Distributed Denial of Service)等，这类攻击本质上关注的不是致



(a) 防御性能

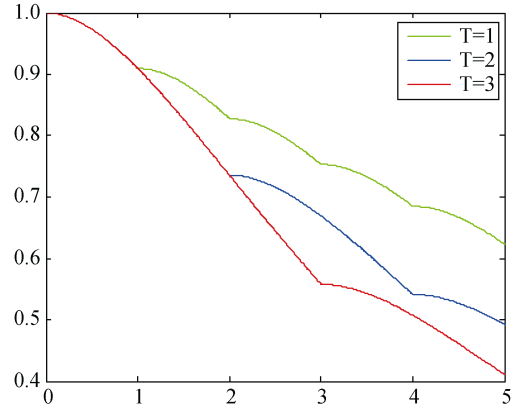


(b) 安全增益

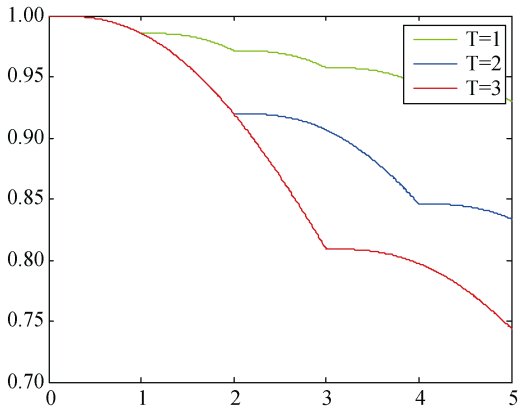


(c) 收益投资比

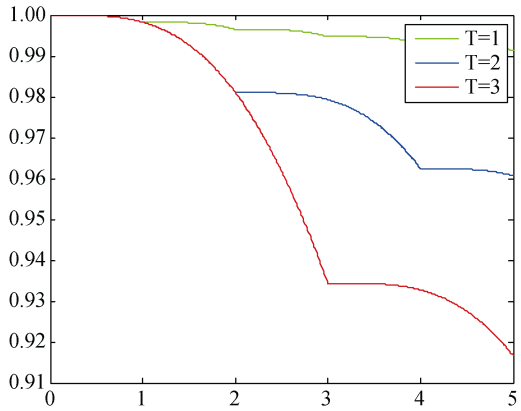
图 6　防御性能、安全增益和收益投资比随 l 变化曲线



(a) l=1，T取不同值的性能



(b) l=2，T取不同值的性能



(c) l=3，T取不同值的性能

图 7　防御性能随 l 和变换周期 T 的变化曲线

使被攻击系统产生错误的输出，而是使系统资源耗尽，无法对外提供服务；此外，拟态防御也无法保护协议规程本身存在的缺陷，如针对 BGP 路由协议的中间人劫持攻击，因为该类攻击利用了 BGP 协议本身的设计缺陷，所有基于该协议实现的异构执行体均存在相同脆弱性。

7　结论

本文针对现有防御技术无法应对未知特征和未知缺陷的攻击，提出了一种理想的防御机制，称为拟态防御，或动态非相似余度(DHR: Dynamical Heterogeneous Redundant)。DHR 标准框架主要由异构执行体集合、分发器、拟态变换器和判决器组成，主要思路是针对保护的元功能构造多个功能等价的异构执行体，利用拟态变换器动态调度在线的异构执行体，使异构执行体对外呈现线上/线下的动态切换，从而使异构执行体的后门和漏洞难以利用，达到阻断攻击者的攻击过程的目的；同时依托判决器，实现在攻击发生时，能够发现未知攻击。

本文首先介绍了网络空间拟态防御 DHR 的基本模型，并在基本模型的基础上推广到一般的信息系

统或网络, 构建了具有N个节点的防护链模型, 并分为四种场景对基本模型和防护链模型的效果进行了理论推导, 针对攻防过程的复杂性, 难以得出解析解, 因此, 在理论推导的基础上给出了特定参数下的数值解。理论和实践结果均表明, 在一定程度上增大系统资源开销的条件下, 拟态防御能够显著提升被保护系统的安全性, 具备两个显著的优点: 1)无需漏洞和后门特征就可以防护; 2)系统具备内置的发现攻击的能力; 3)可以基于不安全的异构执行体构建安全的信息系统。

当然, 本文着重介绍拟态防御的基本思想, 无论在理论研究还是工程实践上都有大量工作需要进一步研究, 包括: 如何评估异构执行体之间的相异性、如何在获得安全性的同时不降低系统的性能, 以及如何在现有系统中导入拟态防御等。

致　谢

本文工作受到中国博士后基金项目(No.44603)、国家自然科学基金项目(No.61309020)、国家自然科学基金创新研究群体项目(No.61521003)和国家重点研发计划项目(Nos. 2016YFB0800100, 2016YFB0800101)。网络空间拟态防御由信息工程大学邬江兴教授于 2013 年提出, 本文的相关工作是在邬江兴教授相关工作的基础上完成的。

参考文献

[1] National Science and Technology Council, Federal Cybersecurity Research and Development Strategic Plan, Feb., 2016.

[2] National Science and Technology Council, Trustworthy Cyberspace: Strategic Plan for The Federal Cybersecurity Research and Development Program, Dec., 2011.

[3] National Institute of Standards and Technology, Guide to Intrusion Detection and Prevention Systems (IDPS), Feb., 2007.

[4] L. Spitzner. Honeypots: catching the insider threat, Computer Security Applications Conference, IEEE, 2003, pp.170 – 179.

[5] Suman Jana; Donald E. Porter; Vitaly Shmatikov, TxBox: Building Secure, Efficient Sandboxes with System Transactions, 2011 IEEE Symposium on Security and Privacy, Date 22-25 May 2011, pp. 329-344.

[6] Feiyi Wang; F. Jou; Fengmin Gong; C. Sargor; K. Goseva-Popstojanova; K. Trivedi, SITAR: A Scalable Intrusion-tolerant Architecture for Distributed Services, Foundations of Intrusion Tolerant Systems, 2003, pp. 359-367.

[7] Dunlop M, Groat S, Urbanski W, et al. Mt6d: A moving target ipv6 defense[C]//Military Communications Conference, 2011-Milcom 2011. IEEE, 2011: 1321-1326.

[8] Jafarian J H, Al-Shaer E, Duan Q. Openflow random host mutation: transparent moving target defense using software defined networking[C]//Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012: 127-132.

[9] Jafarian J H, Al-Shaer E, Duan Q. An Effective Address Mutation Approach for Disrupting Reconnaissance Attacks[J]. Information Forensics and Security, IEEE Transactions on, 2015, 10(12): 2562-2577.

[10] Jafarian J H, Al-Shaer E, Duan Q. Adversary-aware IP address randomization for proactive agility against sophisticated attackers[C]//Computer Communications (INFOCOM), 2015 IEEE Conference on. IEEE, 2015: 738-746.

[11] Luo Y B, Wang B S, Cai G L. Effectiveness of Port Hopping as a Moving Target Defense[C]//Security Technology (SecTech), 2014 7th International Conference on. IEEE, 2014: 7-10.

[12] E. Al-Shaer, W. Marrero, A. El-Atway and K. AlBadani, Network Configuration in a Box: Towards End-to-End Verification of Network Reachability and Security, In Proceedings of 17th International Conference on Network Communications and Protocol (ICNP'09), pp. 123-132, Princeton, 2009.

[13] Portner J, Kerr J, Chu B. Moving Target Defense Against Cross-Site Scripting Attacks (Position Paper)[M]//Foundations and Practice of Security. Springer International Publishing, 2014: 85-91.

[14] Hovav Shacham, Matthew Page, Ben Pfaff, Eu-Jin Goh, Nagendra Modadugu, and Dan Boneh. On the effectiveness of address-space randomization. In ACM Conference on Computer and Communications Security (CCS), CCS '04, pages 298–307, New York, NY, USA, 2004. ACM.

[15] Stephen W. Boyd, Gaurav S. Kc, Michael E. Locasto, Angelos D. Keromytis, and Vassilis Prevelakis. On The General Applicability of Instruction-Set Randomization. IEEE Transactions on Dependable and Secure Computing, 7(3), 2010.

[16] Anh Nguyen-Tuong, David Evans, John C. Knight, Benjamin Cox, and Jack W. Davidson. Security through Redundant Data Diversity. In IEEE/IFPF International Conference on Dependable Systems and Networks, June 2008.

[17] Azab M, Eltoweissy M. ChameleonSoft: Software behavior encryption for moving target defense[J]. Mobile Networks and Applications, 2013, 18(2): 271-292.

[18] Pratyusa K. Manadhata and Jeannette M. Wing. An attack surface metric. IEEE Transactions on Software Engineering, 99(PrePrints), 2011, Volume: 37, Issue: 3, Pages: 371 – 386.

[19] Zhuang R, DeLoach S A, Ou X. A model for analyzing the effect of moving target defenses on enterprise networks[C]//Proceedings of the 9th Annual Cyber and Information Security Research Conference. ACM, 2014: 73-76.

[20] Zhuang R, DeLoach S A, Ou X. Towards a Theory of Moving Target Defense[C]//Proceedings of the First ACM Workshop on Moving Target Defense. ACM, 2014: 31-40.

[21] Rui Zhuang, Alexandru G. Bardas, Scott A. DeLoach, Xinming Ou. A Theory of Cyber Attacks[C]//Proceedings of the Second ACM Workshop on Moving Target Defense. ACM, 2015.

[22] Carter K M, Riordan J F, Okhravi H. A Game Theoretic Approach to Strategy Determination for Dynamic Platform Defenses[C]//Proceedings of the First ACM Workshop on Moving Target Defense. ACM, 2014: 21-30.

[23] Prakash A, Wellman M P. Empirical Game-Theoretic Analysis for Moving Target Defense[C]//Proceedings of the Second ACM Workshop on Moving Target Defense. ACM, 2015: 57-65.

[24] Jin B. Hong, Dong Seong Kim. Assessing the effectiveness of moving target defenses using security models[J]//Dependable and Secure Computing, IEEE Transactions on, 2015, 10(11): 1545-5971.

[25] Cyber Cybenko G, Hughes J. No free lunch in cyber security[C]//Proceedings of the First ACM Workshop on Moving Target Defense. ACM, 2014: 1-12.

[26] Yujuan Han, Wenlian Lu, Shouhuai Xu. Characterizing the Power of Moving Target Defense via Cyber Epidemic Dynamics, HotSoS'14, Proceedings of the 2014 Symposium and Bootcamp on the Science of Security, April 08 - 09 2014, Raleigh, NC, USA.

[27] G. Kc, A. Keromytis, and V. Prevelakis. Countering code-injection attacks with instruction-set randomization. In Proc. ACM CCS'03

[28] E. Barrantes, D. Ackley, T. Palmer, D. Stefanovic, and D. Zovi. Randomized instruction set emulation to disrupt binary code injection attacks. In Proc. ACM CCS'03, pp 281–289.

[29] F. Cohen. Operating system protection through program evolution. Comput. Secur., 12(6):565–584, October 1993

[30] S. Forrest, A. Somayaji, and D. Ackley. Building diverse computer systems. In Proc. HotOS-VI.

[31] C. Giuffrida, A. Kuijsten, and A. Tanenbaum. Enhanced operating system security through efficient and fine-grained address space randomization. In Proc. USENIX Security'12

[32] A. Homescu, S. Brunthaler, P. Larsen, and M. Franz. Librando: transparent code randomization for just-in-time compilers. In Proc. ACM CCS'13

[33] T. Jackson, B. Salamat, A. Homescu, K. Manivannan, G. Wagner, A. Gal, S. Brunthaler, C. Wimmer, and M. Franz. Diversifying the Software Stack Using Randomized NOP Insertion In S. Jajodia, A. Ghosh, V. Swarup, C. Wang, and X. Wang, editors, Moving Target Defense, pages 77–98.

[34] V. Kiriansky, D. Bruening, and S. Amarasinghe. Secure execution via program shepherding. In Proc. USENIX Security'02.

[35] C. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. Reddi, and K. Hazelwood. Pin: Building customized program analysis tools with dynamic instrumentation. In Proc. PLDI'05.

[36] N. Nethercote and J. Seward. Valgrind: A framework for heavy-weight dynamic binary instrumentation. In PLDI'07.

[37] V. Pappas, M. Polychronakis, and A. Keromytis. Smashing the gadgets: Hindering return-oriented programming using in-place code randomization. In IEEE Symposium on Security and Privacy'12.

[38] R. Wartell, V. Mohan, K. Hamlen, and Z. Lin. Binary stirring: Self-randomizing instruction addresses of legacy x86 binary code. In ACM CCS'12.

[39] The PaX Team. http://pax.grsecurity.net/docs/aslr.txt.

[40] S. Bhatkar, D. DuVarney, and R. Sekar. Address obfuscation: An efficient approach to combat a board range of memory error exploits. In USENIX Security Symposium, 2003.

[41] A. Avizienis. The n-version approach to fault-tolerant software. IEEE TSE, (12): 1491–1501, 1985.

[42] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In CRYPTO'89, pages 307–315.

[43] A. Shamir. How to share a secret? CACM, 22:612–613, 1979.

[44] Fida Gillani, Ehab Al-Shaer, Samantha Lo, etc., Agile Virtualized Infrastructure to Proactively Defend Against Cyber Attacks[C]//INFOCOM 2015,ACM,2015.

[45] Vikram S, Yang C, Gu G. Nomad: Towards non-intrusive moving-target defense against web bots[C]//Communications and Network Security (CNS), 2013 IEEE Conference on. IEEE, 2013: 55-63.

[46] Peng W, Li F, Huang C T, et al. A moving-target defense strategy for Cloud-based services with heterogeneous and dynamic attack surfaces[C]//Communications (ICC), 2014 IEEE International Conference on. IEEE, 2014: 804-809.

[47] Design Considerations in Boeing 777 Fly-By-Wire Computers, Y. C. (Bob) Yeh, Boeing Commercial Airplane Group.

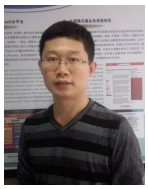
[48] Triple-Triple Redundant 777 Primary Flight Computer, Y. C. (Bob) Yeh.

[49] Stefano Vissicchio, Luca Cittadini, Olivier Bonaventure, Geoffrey G. Xie, Laurent Vanbever. On the Co-Existence of Distributed and Centralized Routing Control-Planes.

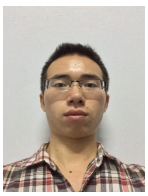
[50] Robert M. Gray. Entropy and Information Theory, Springer, 2011.

[51] J. Moy. OSPF Version 2, RFC2328, https://tools.ietf.org/html/rfc2328.

[52] ZTE. ZXR10 M6000-S Carrier-class Router. http://wwwen.zte.com.cn/en/products/ bearer/ data_communication/router_bmsg.



扈红超 于 2010 年在信息工程大学信息与通信工程专业获得博士学位。现任国家数字交换系统工程技术研究中心副研究员。研究领域为网络安全防御、云安全、新型网络体系结构。Email: 13633833568@139.com



王禛鹏 于 2015 年在武汉大学通信工程专业获得学士学位。现在国家数字交换系统工程技术研究中心信息与通信工程专业攻读硕士学位。研究领域为网络安全防御。Email: whuwzp@foxmail.com

陈福才 于 2002 年在信息工程大学计算机科学与技术专业获得硕士学位。现任国家数字交换系统工程技术研究中心研究员。研究领域为网络安全防御、电信网安全、云安全等。Email: 13503827650@139.com