

# 基于主机事件的攻击发现技术研究综述

冯 云<sup>1,2</sup>, 刘宝旭<sup>1,2</sup>, 张金莉<sup>1,2</sup>, 张 越<sup>1,2</sup>, 刘奇旭<sup>1,2</sup>

<sup>1</sup>中国科学院信息工程研究所 北京 中国 100093

<sup>2</sup>中国科学院大学网络空间安全学院 北京 中国 100049

**摘要** 在飞速发展的信息时代和数据时代,网络攻击对个人隐私、工作生活乃至生命财产安全带来了严重威胁。而主机作为人类进行日常工作交流、生活娱乐、数据存储的重要设备,成为了网络攻击的主要目标。因此,进行主机攻击发现技术的研究是紧迫且必要的,而主机事件作为记录主机中一切行为的载体,成为了当今网络攻防领域的重点研究对象。攻击者在主机中的各种恶意操作会不可避免地记录为主机事件,但恶意事件隐藏在规模庞大的正常事件中难以察觉和筛选,引发了如何获取主机事件、如何识别并提取恶意事件、如何还原攻击过程、如何进行安全防护等一系列问题的学术研究。本文对基于主机事件的攻击发现技术相关研究进行了广泛的调研和细致的汇总,对其研究发展历程进行了梳理,并将本文所研究的基于主机事件的攻击发现技术与入侵检测、数字取证两大研究方向从分析对象、分析方法、作用时间、分析目的4个方面进行了对比,阐明了本文所研究问题的独特之处,并对其下定义。随后,本文对基于主机事件的攻击发现技术涉及的关键概念进行了解释,提出了该领域面临的依赖关系爆炸和及时性两大问题,并将研究按照阶段划分为主机事件采集、主机事件处理、主机事件分析三个类别,分别介绍了三个类别围绕两大问题共计12个细分方向的研究成果和进展,最后结合研究现状提出了主机事件记录的完整性和可信性、攻击发现的时效性、跨设备的攻击发现、多步骤攻击的发现、算法的运用等5个未来可能的研究方向。

**关键词** 主机安全; 主机事件; 攻击发现; 攻击路径

中图法分类号 TP309.5 DOI号 10.19363/J.cnki.cn10-1380/tn.2023.07.03

## A Survey of Attack Discovery Technology Based on Host Events

Feng Yun<sup>1,2</sup>, Liu Baoxu<sup>1,2</sup>, Zhang Jinli<sup>1,2</sup>, Zhang Yue<sup>1,2</sup>, Liu Qixu<sup>1,2</sup>

<sup>1</sup>Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

<sup>2</sup>School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China

**Abstract** In the rapidly developing era of information and data, cyberattacks pose a serious threat to human privacy, work and even the safety of life and property. As an important equipment for human to communicate in daily work, and for entertainment and data storage, the host has become the main target of cyberattacks. Therefore, it is urgent and necessary to study the host attack discovery technology. And as a carrier to record all behaviors in the host, host events have become the focus of research. Attackers' various malicious operations in the host will inevitably be recorded as host events. However, malicious events are hidden in large-scale normal events, that are difficult to detect and filter out, leading to academic research on a series of issues such as how to obtain host events, how to identify and extract malicious events, how to reconstruct the attack process, and how to conduct security protection. This paper has carried out extensive research and detailed summary on the research related to host event based attack discovery technology, and combed its research and development history, then compares with intrusion detection and digital forensics from four aspects: analysis object, analysis method, time of action, and analysis purpose. After, the definition of host event based attack discovery technology was conducted. Subsequently, this paper explains the key concepts involved. Two major problems which are dependency explosion and timeliness are pointed out. Then the research is divided into three categories according to the stages: host event collection, host event processing and host event analysis. The research results and progress of the three categories around the two major problems, which are 12 subcategories in total, are introduced respectively. Finally, the possible research directions in the future are pointed out according to the current research situation, including integrity and credibility of host event records, timeliness of attack discovery, cross-device attack discovery, multi-step attack discovery, and algorithm application.

**Key words** host security; host event; attack discovery; attack path

通讯作者: 刘奇旭, 博士, 副研究员, Email:liuqixu@iie.ac.cn。

本论文获得了中国科学院青年创新促进会(No. 2019163), 国家自然科学基金项目(No. 61902396), 中国科学院战略性先导科技专项项目(No. XDC02040100)课题资助; 获得中国科学院网络测评技术重点实验室和网络安全防护技术北京市重点实验室资助。

收稿日期: 2020-03-19; 修改日期: 2020-05-31; 定稿日期: 2022-12-30

## 1 引言

近年来,网络攻击越来越频发,从个人到组织、从企业到国家都处在网络攻击的威胁之下,网络攻击不仅会给隐私和财富造成严重损失,也给社会的安全性和稳定性带来了严重威胁。而主机是网络攻击的重要目标,究其原因可以总结出以下几点:

1) 主机是当今信息时代个人和组织都普遍拥有和使用的设备,被广泛应用于处理日常工作、存储重要数据、开展信息交流,促进了社会各项职能的沟通运转。小到个人的隐私,大到国家的机密都可能在主机上进行存储,或由于在主机上处理过而留下痕迹,因此成为重要的攻击目标。

2) 对于企业等拥有较大规模的内网环境的组织,其内网由各种等级的主机、服务器及打印机、路由器等设备构成,设备之间存在着互联互通的可能性。很多 APT 攻击都是由内网中的一台主机作为突破口,通过内网横向移动逐渐入侵其他设备以实施更大的破坏。在这种场景中,对于主机的入侵和攻击是网络攻击的重要一环。

3) 主机通常情况下是由人进行保管和使用的,尤其是个人电脑和企业环境中的普通主机,使用者的安全意识和安全能力就成为了影响主机脆弱性的重要因素。根据卡巴斯基《2019 年网络攻击分析报告》<sup>[1]</sup>统计,一年中有 19.8% 的用户计算机遭受了至少一次攻击,钓鱼邮件、钓鱼网站、弱口令等问题依然是主机被攻击的主要入口。由此可以看出,主机使用者的安全意识和能力还需进一步提升,由此导致的主机脆弱性也让主机成为了攻击者主要的攻击起点。

针对面向主机的攻击发现的研究必要性呼之欲出,而准确及时地发现攻击从而避免损失是最重要的研究目标。相关研究可以大致划分为三类,即基于入侵检测的攻击发现、基于攻击取证的攻击发现和基于主机事件的攻击发现。这三类技术各有侧重,在分析对象、方法、目的和作用时间方面存在着一些差异,同时也存在着相通之处。本文对相关的研究做了充分的调研梳理,发现绝大多数的主机攻击发现研究是关于入侵检测或攻击取证的,但随着攻击的规模、未知性、隐匿性和复杂度的日益增长,这两种技术分别暴露出一些缺陷,而基于主机事件进行攻击发现的研究受到了越来越多的关注。

主机事件即主机中切实发生的事情,由实施主体、实施对象和具体的行为构成。主机事件可以由操作人员引发,也可以是系统中应用程序在运行过

程中的自主行动。一个事件是独立的一次行为,事件间存在着关联,攻击是由多个存在关联的事件串联而成的。基于主机事件的攻击发现就是指通过对事件的采集、处理、分析以及事件间关联的构建来还原攻击过程,发现攻击的发生或重构攻击的场景。基于主机事件的攻击发现是以事件为研究对象,以还原攻击过程作为攻击发现的方法,能够更好地应对目前越来越复杂的攻击发现问题。因此,本文对该领域的相关研究进行了总结梳理,帮助研究人员了解研究现状、把握研究方向、进一步开展研究工作。

文章结构如下,第 1 章是引言;第 2 章梳理了主机攻击发现的研究发展历程,对相关研究进行了对比分析,提出了对基于主机事件的攻击发现技术的定义,并对该领域的一些关键术语进行了释义;第 3 章对基于主机事件的攻击发现当前的研究问题和研究进展进行了分类整理,总结了该领域的重点研究问题,并整理了学者们针对各问题所提出的方案;第 4 章基于目前的研究现状,提出了一些未来发展方向;第 5 章对本文工作进行了总结。

## 2 背景介绍

### 2.1 发展历程

传统的攻击发现技术主要包括入侵检测和数字取证。下面将对这两种技术的发展历程进行阐述。

#### 2.1.1 入侵检测

入侵检测技术作为攻击发现的重要手段,在安全防护方面起着重要的作用。可以弥补防火墙技术、访问控制机制、数据加密、身份认证等静态安全防护策略的缺陷。

入侵检测是从计算机网络或系统中的若干关键点搜集信息并对其进行分析,从中发现网络或系统中是否有违反安全策略的行为或遭到攻击的迹象的一种机制<sup>[2]</sup>。早在 1980 年,Anderson 在一份技术报告<sup>[3]</sup>中首次提出了入侵检测的概念,并将入侵定义为:潜在的、有预谋的、未经授权的访问,企图致使系统不可靠或无法使用。入侵检测则是对入侵行为的识别,具有入侵检测功能的系统称为入侵检测系统(Intrusion Detection System, IDS)。

##### 1) 入侵检测的分析方法

入侵检测的核心在于检测,检测分析技术主要有两类:异常检测和误用检测。

异常检测技术也称为基于行为的检测技术,建立在一个系统的正常行为轮廓上,用户和系统的所有正常行为可以建立行为基线,若捕获到的行为与该行为基线的差异程度超过了阈值则发出入侵预

警。异常检测技术的核心问题是建立行为基线, 主要方法有统计分析、模式预测。

误用检测技术也称为基于知识的检测技术或模式匹配检测技术。其前提是假设所有入侵行为都具有一定的模式或特征, 对所有已知入侵行为建立模式库, 将被检测的数据与模式库进行匹配来检测入侵攻击。误用检测技术主要有以下几种: 状态转移分析<sup>[4-5]</sup>、规则匹配、模型推理。

## 2) 入侵检测的技术演进

在体系结构方面, 早期的入侵检测体系结构都是集中式结构, 但难以应对网络结构的复杂化和大规模化, 因此向着分布式发展。随着云计算的发展, 基于云计算的分布式入侵检测系统研究成为了热点<sup>[6]</sup>。

在数据分析方面, 随着网络攻击和数据量的日益增长, 传统的技术在处理速度和处理质量上都难以满足需求, 大数据、机器学习、深度学习等技术成为入侵检测目前的研究热点。大数据的出现改进了入侵检测对海量数据的处理能力, 运用数据挖掘技术的自动化处理能力, 可以快速、高效地从海量数据中挖掘出入侵行为特征<sup>[7-9]</sup>。机器学习取代了大量的人力劳动, 通过大规模已知数据训练出系统的正常行为模式, 从而识别异常攻击行为<sup>[10-11]</sup>。随着攻击复杂程度的增加, 传统的机器学习遇到瓶颈, 深度学习的概念<sup>[12]</sup>在入侵检测领域受到关注。深度学习具有更高维的特征表达能力和对复杂问题的处理能力, 在模型构建和分类准确度上都有很好的表现<sup>[13-15]</sup>。异常检测常常因为缺乏标记而使用无监督学习方法, 但是建模成为异常检测的一个难点, 随着强化学习的理论和实践发展, 强化学习强调与环境的交互学习, 利用环境的反馈信号进一步优化选择策略, 解决了异常检测的模型不确定问题<sup>[16-18]</sup>。

### 2.1.2 数字取证

数字取证又叫计算机取证、电子取证, 是指从计算机及相关外设中获取可靠的数字证据并对其进行确认、保护、分析的过程。取证的目的是为了证明某个客观事实存在, 据此还原曾经发生的真实过程, 应用于司法鉴定、计算机攻击发现等领域。

根据取证时间可将计算机取证划分为事后取证和实时取证。事后取证也叫静态取证, 是指在已遭受入侵的情况下, 运用各种技术手段从计算机及外设中提取分析电子证据的过程。实时取证也叫动态取证, 与入侵检测系统、蜜罐、蜜网等紧密结合, 实时提取电子证据并动态智能分析攻击过程, 根据预先设定好的反应机制进行响应, 在确保取证安全的同

时获取攻击者的大量证据。

计算机取证主要技术包括: 证据获取技术、证据保全技术、证据分析技术等。

#### 1) 证据获取技术

证据获取技术的关键是在获取数据的同时不破坏原始介质, 应保证不改变原始记录、不在作为证据的计算机上执行无关的程序、详细记录所有的取证活动、妥善保存得到的证据<sup>[19]</sup>。现有的取证技术主要有对磁盘中删除数据进行恢复、关键字型证据搜索、文件内容显示等, 针对不同的数据存储介质和存储环境, 证据获取方式包括移动存储介质的数据获取、在线数据获取、关闭状态计算机硬盘数据获取<sup>[20]</sup>。

#### 2) 证据保全技术

证据保全技术是指为保证计算机证据的完整性、机密性、可用性、不可否认性和取证流程中使用电子证据的规范性和合法性, 使用电子证据保护机制对电子证据的传输、存储和使用等环节进行保护。

#### 3) 证据分析技术

证据分析技术就是利用相关工具分析原始数据、提取关键数据、挖掘有用信息, 试图对攻击者的身份、攻击时间、目标、意图、手段以及造成的后果给出明确说明, 还原攻击过程并给出分析报告。传统证据分析技术主要是在已经获取的数据流中寻找相匹配的关键词, 进行文件属性分析、数据解密、攻击者画像、证据间关联分析等<sup>[21]</sup>。

随着网络攻击内存化以及云计算、大数据、区块链等新技术的发展, 数字取证面临着更多的技术挑战, 国内外研究机构也针对这些问题做了相关研究, 包括内存取证技术、无文件攻击取证技术、大数据取证技术、云计算取证技术、区块链取证技术。

内存取证技术, 是指从计算机物理内存和页面交换文件中查找、提取、分析易失性证据, 是对传统基于文件系统取证的重要补充<sup>[22-23]</sup>。计算机中运行的程序都存储在内存中, 用于记录 CPU 运算的临时数据, 因此内存中存放了程序运行过程中最重要的状态信息。针对仅存于内存中、关机就消失的木马程序, 内存取证是唯一的方式。

无文件攻击取证技术用于应对无文件攻击<sup>[24]</sup>。无文件攻击也称为非恶意软件攻击, 由于其隐蔽、不在硬盘上保留传统恶意代码、功能依靠留驻内存代码等实现特点, 取证难度较传统恶意代码攻击要大, 需要重视易失性证据的保护提取。

随着存储技术和计算机网络通信技术的发展,

计算机系统和计算机网络中每天产生的数据庞大而且复杂, 用传统的数字取证工具无法在可接受的时间范围内对这些内容进行证据提取、分析和保全<sup>[21]</sup>, 大数据取证技术应运而生, 相关研究有分布式取证系统、内容抽样<sup>[25]</sup>、数据搜索、数据碰撞、数据挖掘、数据画像、犯罪网络分析、犯罪热点分析等。

云取证技术<sup>[26]</sup>是针对云计算的攻击进行的电子数据取证过程, 是电子数据取证技术在云计算场景下的应用。针对云取证技术的研究热点包括云取证模型的研究、云环境中电子数据的提取技术研究、云环境中电子数据的分析技术研究等。

近年来区块链逐渐兴起, 因其去中心化、不可篡改、可追溯、永久存储等技术特点天然适用于电子证据的存证固证, 可为保证电子证据的完整性、真实性和可审计性提供技术基础, 由此诞生了区块链取证技术, 其全程留痕、不可篡改的记录功能, 是解决电子证据存证固证的技术基础<sup>[27]</sup>。

### 2.1.3 基于主机事件的攻击发现

从入侵检测和数字取证两种技术的特点和发展历程可以看出, 两者虽然都适用于攻击发现场景且已经有了可观的研究进展, 但也存在着一定的局限性。从入侵检测技术的定义来看, 一切通过分析来自计算机网络或系统关键节点的线索来发现异常或攻击的技术都属于入侵检测技术的范畴。

本文所讨论的基于主机事件的攻击发现与入侵检测和数字取证的关系如图 1 所示, 三者存在着一些相通之处, 但也有各自的侧重点, 下面将对三种技术进行对比说明, 三种技术的对比总结于表 1。

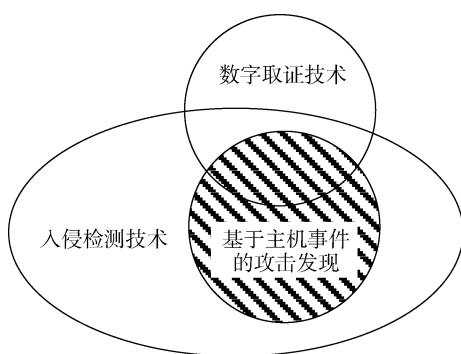


图 1 三种技术的关系图示

Figure 1 Relationship Between the Three Technologies

首先, 对于分析对象, 由于入侵检测技术定义的广泛性, 其分析对象既包括由入侵检测系统采集到的数据, 也包括系统、平台所记录的日志。由入侵检测系统采集的数据通常是依赖规则匹配的异常警

报, 而规则需要依靠人力进行维护和更新, 规则不全面或更新不及时就容易导致漏报和误报的发生。以日志为分析对象的技术是入侵检测的一种子技术, 称为日志异常检测技术(Log Anomaly)<sup>[28-32]</sup>, 其所分析的日志对象包括 Web 日志、系统日志 syslog、平台所产生的日志, 如 OpenStack、Hadoop 等。而主机事件是指主机内所有程序、对象产生的行为, 一切可以记录这种行为的日志、线索都可以作为主机事件的来源。其中最主要的数据来源是系统自身的审计日志框架, 如 Windows 系统的 ETW 框架<sup>[33]</sup>或 Linux 系统的 Auditd 工具<sup>[34]</sup>, 或在系统内核或上述框架上加以修改后开发出的工具。从捕获层面上看, 此类框架和工具在系统内核层面进行数据捕获, 而系统日志 syslog 位于应用层, 记录的是应用层的错误信息。从捕获来源上看, 主机事件来自系统中所有程序、对象的行为, 而 OpenStack、Hadoop 等平台日志或 MySQL 等程序日志只记录与本程序相关的行为, syslog 只记录系统中符合规则配置的错误信息。对于攻击取证技术, 由于其主要面向事后分析, 因此分析对象更加全面和充分, 包括了从物理介质中能够获取到的各类数据, 上述入侵检测技术的分析对象和主机事件都可以作为攻击取证的分析对象。

其次, 对于分析方法, 入侵检测技术的分析对象往往具有规范的格式或具有可预测的连续模式<sup>[32]</sup>, 所以十分适合进行模式或模型的建立, 也适合利用各种算法进行自动的特征提取和分析, 从而实现单条数据的异常发现, 或单一来源的异常检测。但是随着攻击规模的增大和复杂性的提高, 攻击的时间空间跨度都更加分散, 仅靠单点的异常检测已难以应对。如今, 以高级持续性威胁(Advanced Persistent Threat, APT)为代表的多步骤攻击已成为国家或组织间实施网络对抗的重要手段, 其数量、规模、复杂程度和影响范围都呈现着显著上升的趋势。APT 有着特定目标, 经过对目标的充分调查了解后制定针对性的方案实施攻击, 进行机密数据窃取<sup>[35-37]</sup>或关键基础设施破坏<sup>[38]</sup>, 因此防范难度高、危害程度大。APT 从开始到攻击完成会经历多个阶段, 每一个阶段都可能发生在不同载体上、使用不同的工具, 且各阶段之间可能存在着较长的时间间隔。传统的入侵检测系统侧重于检测出单独的异常, 对于 APT 这种长战线、跨设备的攻击, 每个阶段中发生的单个事件可能并不异常, 有些攻击者还刻意使用良性服务来降低被发现的风险。因此, 不仅需要判断单个事件的异常性, 还需要转换为事件分析思路, 将事件有效关联, 判断事件链的异常性。基于主机事件的攻击发

现的核心思路就是进行事件间依赖关系的推理，构建事件链，从而进行攻击发现，因此非常适用于以APT为代表的多步骤复杂攻击的发现和重构。

第三，对于作用时间，攻击取证技术通常是在已知有攻击或异常发生的情况下，进行事后的线索采集、数据分析和场景还原，这就导致了滞后性，在机制上就难以及时地进行攻击的发现和阻止；入侵检测技术和基于主机事件的攻击发现不强调作用时

间，但都以及及时性作为重要的研究方向。

最后，对于分析目的，虽然三者都是攻击发现技术，但“发现”的侧重点不同。入侵检测技术侧重攻击的检测，即是否有攻击发生；攻击取证技术侧重事后的攻击重建，即攻击过程和手段的发现和重构；而本文所讨论的基于主机事件的攻击发现技术既指发现攻击是否发生，也包含在发现有攻击发生的同时进行攻击过程和手段的还原。

表 1 三种技术的对比  
Table 1 Comparison of the Three Technologies

技术类别	分析对象	分析方法	作用时间	分析目的
入侵检测	入侵检测系统、系统、平台日志	单点异常检测，模型建立和匹配	兼有事中和事后	侧重及时攻击发现
数字取证	能够从物理介质中获取到的各类数据	信息挖掘，数据解密，证据间关联分析	侧重事后	侧重攻击过程重建
主机事件攻击发现	记录了主机内行为的各类线索，以内核审计框架为主要来源	建立事件间关联，识别异常事件链	兼有事中和事后	兼有及时攻击发现和攻击过程重建

最后，可以对基于主机事件的攻击发现技术进行定义：基于主机事件的攻击发现技术是以主机中所有程序、对象所产生的事件为分析对象，以推理事件间依赖关系、构建并分析事件链为核心思路，进行及时或事后的攻击事实发现、攻击过程与手段还原的技术。

2.2 关键术语释义

在主机攻击发现问题上有很多特定的术语，为了更好的对该领域的研究进行阐述，在这一节对一些关键术语进行释义。

1) **主体(Subject)**，通常指进程，由于进程可以产生自主行为并对其他的进程或文件产生影响，故表示为主体。

2) **对象(Object)**，包括文件、网络套接字等，会受到主体的影响而发生一定的变化，如文件内容的改变。

3) **依赖关系**，又称因果关系，当某主体对某对象施加了一定的影响，那它们中间就存在了一定的依赖关系。如，某进程对一个文件执行了写操作，文件属性的改变是由该进程导致的，那么该文件与该进程就产生了依赖关系。

4) **事件(Event)**，主体对对象施加的影响称为事件，如创建子进程、写入文件、读取套接字等。事件由一个三元组构成，即<主体，操作，对象>。

5) **信息流**，又称数据流，事件本质上是信息的流动，依赖关系本质上是由信息流维持的。例如，当

某进程读取了文件 A，又写入了文件 B，那么信息流就从文件 A 流向了进程，又从进程流向了文件 B。

6) **终止事件**，当某进程被杀死或某文件被删除，信息流就无法再流动，称此类事件为终止事件。

7) **事件链**，又称事件路径，从一个初始主体开始，沿着其引发的事件的顺序，即信息流的流向，到终止事件的对象为止，途径的所有主体和对象根据依赖关系相串联，构成事件链。

8) **节点(Node)**，事件链上的主体与对象统称为节点。

9) **边(Edge)**，事件链上节点之间的依赖关系以边的形式表示。

10) **依赖关系图**，又称依赖关系网，事件链上的节点不一定只存在于一条事件链上，多条事件链会根据节点交织在一起形成网络，称为依赖关系图。

11) **追踪(Trace)**，从一个节点开始，根据事件或依赖关系进行其他节点的发现和挖掘的过程称为追踪。根据信息流的方向，追踪分为前向追踪和后向追踪。前向追踪是指从某节点开始，向着它产生影响的方向，即信息流出的方向进行追踪。后向追踪是指从某节点开始，向着对它施加影响的方向，即信息流入的方向进行追踪。

12) **起源(Provenance)**，导致事件链上某一节点出现的之前所有节点都成为该节点的起源。起源追踪的本质是根据依赖关系进行后向追踪。

3 技术研究现状

基于主机事件的攻击发现以主机事件为研究对象, 通过对主机事件的分析、关联和推理发现异常, 识别攻击。在这个过程中会面临如下挑战。

1) 首先是依赖关系爆炸问题, 其原因在于主机端的事件数量繁多且事件间依赖关系错综复杂。每个应用程序在运行过程中会不断产生大量的事件, 如读写文件、修改注册表、套接字交互等, 即使是一个简单的程序启动都可能牵涉到很多文件的读写行为。一台普通的计算机每天会产生超过 100 万个事件<sup>[39]</sup>, 每天的日志数据量以 0.8~3.2GB 的速度增长<sup>[40]</sup>。这为攻击行为提供了良好的隐藏环境, 难以将攻击相关事件准确筛选出来。另外, 多条事件链通过对象节点交织在一起, 形成一张复杂的依赖关系网, 如果直接对所有事件进行依赖关系的构建, 将产生大量冗余的依赖关系, 严重影响攻击发现的时间和空间效率。尤其对以 APT 为代表的多步骤攻击, 其完整攻击过程往往分为多个步骤, 每个步骤之间可能存在着较长的时间间隔, 则攻击起源追溯和攻击路径还原将更加困难。

2) 攻击发现的及时性也是一个重要问题, 从主机端攻击发现的发展历程来看, 攻击取证技术用于在事后分析还原攻击, 入侵检测技术也存在着滞后性, 而滞后的攻击发现将难以避免攻击导致的危害和损失。因此, 如何有效处理数量繁多的主机事件、如何梳理事件间复杂的依赖关系、以及如何尽可能及时准确地发现攻击成为主要的研究问题。

通过主机事件进行攻击发现的过程可以分为三个阶段: 事件采集, 即从主机或相关设备中采集数据, 形成事件形式的记录; 事件处理, 即对采集到的主机事件进行一些预处理, 以便于进一步地分析; 事件分析, 即通过对主机事件本身及事件间关联进行分析来进行攻击发现的过程。

下面将以三个阶段为主线, 对研究者们围绕依赖爆炸和实时性问题所提出的技术方法的现状及发展进行阐述, 共计 12 个细分方向问题如表 2 所示。

3.1 主机事件采集

基于主机事件的攻击发现技术以主机事件为研究对象, 主机中的一切行为, 无论是正常行为还是异常的攻击行为都是由一个个事件串联而成, 事件的采集是进行研究的最基础工作。该领域的研究主要关注以下三种类型的主机事件: 进程事件, 例如进程创建、进程销毁; 文件事件, 例如文件读取、文件写入、目录遍历等; 网络事件, 例如套接字创建、

套接字销毁、数据传输等。

表 2 基于事件的主机攻击发现研究  
Table 2 Research on Event-based Host Attack Discovery

阶段划分	方法分类
事件采集	① 工具及框架
	② 事件约减
	③ 起源追踪
	④ 事件完整性与可信性
事件处理	① 执行单元划分
	② 事件合并
	③ 事件筛选
	④ 对象级起源追踪
事件分析	① 基于异常量化
	② 基于事件筛选
	③ 查询系统
	④ 攻击重放

3.1.1 相关工具及框架

主机事件的一个主要来源是系统的内核审计日志。审计日志可以跟踪用户层应用程序行为和内核层驱动创建的事件对象, 记录了进程的创建销毁、线程行为、文件操作、注册表操作、内存空间使用等事件, 代表性工具有 Windows 系统的 ETW 内核事件跟踪框架和 Linux 系统的 Auditd 内核审计框架。

ETW 框架由三部分组成, 分别是控制者 Controller、提供者 Provider 和消费者 Consumer。控制者负责管理一次事件记录会话的日志形式、存储位置, 并控制一次会话的启动和停止。提供者负责事件的生成, 调用内核函数进行事件编写, 并将生成的事件发送到控制器进行记录。消费者从会话记录中请求事件来进行进一步处理以满足对主机的审计需求, 它接收在缓冲区或日志文件中的事件, 并可设置一定的规则进行事件筛选。有很多研究者基于 ETW 框架开发了相关工具, 能够更灵活地进行功能扩展、事件筛选和异常查询, 如 pywintrace<sup>[41]</sup>和 Fibratus<sup>[42]</sup>。

Auditd 框架提供了一个安全日志记录框架, 用于捕获和记录与安全相关的事件, 由一个基于系统活动生成审计记录的内核组件、一个将这些记录存储到本地文件或远程服务器的用户空间守护进程以及一组用于审计日志检查和后处理的用户空间工具组成。用户可以通过命令行调用这些工具进行事件过滤规则设定、审计事件转发、审计报告生成等。

两种工具都用统一的格式记录系统中发生的种种事件流水及其属性, 包括事件的发起者、接受者、

发起者对接受者的操作、时间戳、进程号、对象标识、命令行命令等,对于事件间依赖关系的确定和事件流的追踪提供了条件。目前,绝大多数研究都基于两种工具采集得到的数据开展。

除了系统本身提供的工具和框架,美国国防部高级研究计划局(DARPA)还开展了一项透明计算计划,旨在开发技术和实验原型系统,以提供APT攻击的取证和实时检测,以及对理想策略的主动实施<sup>[43]</sup>。透明计算计划旨在通过在软件抽象的所有系统层的系统操作期间提供对组件交互的高保真可见性,使当前不透明的计算系统透明,同时施加最小的性能开销。该计划所开发的功能包括,记录和保存所有系统对象的起源,动态跟踪对象之间的依赖关系,将这些依赖关系组装成端到端的系统事件,并推理这些事件。目前有很多研究工作,尤其是致力于解决APT攻击发现的研究工作是基于透明计算计划产生的数据开展的<sup>[44-45]</sup>。

### 3.1.2 采集阶段的事件约减

依赖关系爆炸问题是目前研究的主要问题之一。数量繁多的主机事件和错综复杂的事件间依赖关系给处理分析带来了很大的理论难度,也给存储和运算带来了很大的开销。主机事件采集作为进行后续处理和分析的前提工作,是可以采取事件约减措施的第一阶段。

文献[46]旨在在事件采集阶段缓解依赖爆炸问题,把研究重点放在生成事件日志的系统内核上。其分析了日志运行时和存储开销基本上是由传输和处理包含大量冗余的原始日志造成的,因此提出了一个内核支持的日志缓存和约减系统。文章针对Linux系统,设计了一种分布在各种内核数据结构中的多层缓存方案,并使用缓存来检测和抑制冗余事件的生成。通过这种方法,可以在第一时间防止生成大量的原始日志,既减少了传输和存储日志导致的开销,也缓解了依赖关系爆炸问题,有助于事件间依赖关系的高效分析和后续的攻击发现。

### 3.1.3 采集阶段的起源追踪

事件起源追踪描述了数据是如何以现在的形式出现的,包括数据的来源和转换。事件起源信息为攻击取证、追溯和还原提供了基础,广泛应用于数据防泄漏、入侵检测等场景中。对于主机审计日志所记录的事件,事件之间的依赖关系通过日志本身所记录的信息来构建。通过一条记录中的父进程号,可以确定某进程是由哪个进程创建的,某文件由哪个进程进行了读写从而产生了信息的流动。事件依赖关系的本质是信息的流动,但主机审计日志只能记录

一个步骤的信息流动,例如,某进程从多个文件中读取了数据,随后又向多个文件中写入了数据,则难以确定信息的准确流向,这为攻击路径的还原提出了严峻的挑战。

很多事件起源追踪相关研究只针对单一的应用程序<sup>[47-49]</sup>或工作流<sup>[50-51]</sup>,这样导致了起源追踪的局限性,无法获得完整的全局事件记录,难以在攻击发现中发挥作用。在这一背景下,有研究人员提出通过改进系统事件捕获和记录机制,即在事件采集阶段增加属性记录、囊括更多更细致的数据来源,以提供更全面、安全、有效的事件起源和信息流转换记录。

文献[52]设计了PASS系统,在系统级别收集起源信息,可以记录对象创建时的系统环境、库版本等信息。文献[53]提出了LineageFS系统,将进程ID与进程读写的文件描述符相关联。在上述几个系统中,网络和进程间通信主要通过操作底层文件描述符反映在起源记录中,而文献[54]提出了Hi-Fi系统,捕捉所有的内核操作和应用程序操作,从而提供系统起源信息历史的全面视图,包括整个进程执行树、完整的文件系统结构以及可能包括网络通信的信息流。

Thomas等人设计了CamFlow系统<sup>[55]</sup>和CamQuery查询框架<sup>[56]</sup>,与以往的起源追踪系统会将捕获的事件记录为日志形式不同,本系统直接生成图结构,并将起源追踪粒度从进程级细化到了线程级,实现实时的事件采集和起源追踪。

文献[57]提出了LPROV系统,针对现有系统不能应对涉及共享库的依赖关系追踪的问题,将库函数追踪和系统调用追踪相结合,把库调用与引发它的堆栈标识在一起,以便准确地揭示库执行的起源。

### 3.1.4 事件记录完整性与可信性

侵入系统的攻击者很可能通过篡改、删除审计日志等数据,导致主机事件记录缺失、事件链无法完整真实地还原,从而隐藏攻击行为。因此,主机事件记录的完整性和可信性是进行事件间关联、事件链构建和异常事件链识别进而发现攻击的必要前提。

基于密码学的方法是这个领域的常用手段,利用数字签名、哈希、加密算法,结合特定标识、时间戳等具备唯一性的字符串来保证记录的完整性和可信性<sup>[58-60]</sup>。随着区块链技术的兴起,一些研究者借助其透明性和无法篡改的特点,将哈希值存储到区块链上,为完整性验证保存证据<sup>[61-62]</sup>。

在内核层面,文献[63]提出了第一个构建安全起源感知系统的通用框架,它基于LSM框架<sup>[64-65]</sup>和Netfilter框架<sup>[66]</sup>创建了一个起源感知平台,利用LSM



框架在特定内核数据结构中加入安全域, 在内核源代码关键节点处插入安全钩子, 并利用 Netfilter 框架对数据进行签名和验证, 从而防止记录被恶意篡改。

一些研究者在硬件层面进行研究。文献[67]利用 Intel SGX 硬件扩展重新设计一个日志记录系统, 提供一个带有密封和解封原语的安全区域, 以保护内

存和磁盘中的数据不被未经授权的方式修改。另外, 将日志记录与主机进行隔离, 使攻击者难以直接接触到独立的日志记录系统, 也是保证事件记录完整性和可信性的一种方法<sup>[68-70]</sup>。

关于主机事件采集阶段的相关工作及文献总结于表 3。

表 3 主机事件采集阶段相关工作  
Table 3 Related Work in Host Event Collection Stage

分类	核心思想	相关文献
相关工具及框架	利用系统自身提供工具或研究者开发的成熟框架进行标准的事件采集。	—
事件约减	在事件的采集阶段抑制冗余事件产生, 减少事件存储和后续处理分析带来的开销。	[46]
起源追踪	基于系统内核层面, 在采集阶段为事件提供附加信息, 以记录更准确全面的起源信息。	[52-57]
事件完整性与可信性	保证所采集事件的完整性和可信性, 为后续事件链构建、攻击发现提供必要前提。	[58-63, 67-70]

3.2 主机事件处理

主机事件处理阶段是为了更好地将采集到的事件数据用于主机攻击发现分析而采取一系列措施的阶段。根据攻击发现所面临的依赖关系爆炸和及时性两大问题, 研究人员开展了一系列的研究, 包括执行单元划分、事件合并、事件筛选以及对象级的起源追踪。

3.2.1 执行单元划分

该类技术以划分执行单元为核心思想。对于每一个应用程序来说, 主机审计日志并不记录其流入信息流和流出信息流的明确依赖关系。例如, 某用户在某一段时间通过浏览器访问了很多网页, 其中包括一个恶意页面, 并在该页面下载了一个恶意文件, 当研究人员从该恶意文件入手进行后向追溯其来源时, 只能确定文件来源是浏览器, 且在该文件下载时间之前被访问的页面都可能是该文件的来源, 难以定位到明确的恶意页面。有很多研究者都发现了这个问题, 提出了一些启发式方法来解决, 如通过时间戳筛选依赖关系、设置白名单进行过滤等<sup>[71-73]</sup>。然而, 当事件数量较多时, 时间戳方法难以奏效, 而白名单需要人为设置, 可能导致假阴性。为更好地解决这一问题, Kyu Hyung Lee 等人提出了执行单元划分思想, 并进行了持续的研究。

文献[74]通过逆向应用程序的二进制文件, 检测代码中的事件处理循环, 在代码中进行插桩来记录运行期间的单元边界及单元依赖性。一个单元本质上是一个事件处理循环的迭代。例如, 对于浏览器应

用程序, 其创建连接、加载页面等行为是通过循环迭代实现的。由此生成的依赖关系图中, 一个进程被分解成许多独立单元, 每个单元访问的文件、套接字等对象都有相应划分, 因此缓解了依赖关系爆炸和起源追溯难的问题。

随后, Kyu Hyung Lee 等人承接已有的进程单元划分工作, 提出了数据单元划分思想<sup>[75]</sup>。对于需要频繁读写文件的应用程序, 如数据库应用, 其文件读写事件与前面事件的依赖关系难以明确, 而数据单元划分思想将文件划分为多个数据单元, 以便根据较小的数据单元定义依赖关系。

文献[40]将执行单元划分思想系统化, 构建了从主机审计日志入手进行攻击发现的流程, 并提出用类似正则表达式的方式标识进程单元, 从而让依赖关系分析更加精确, 并在此基础上进行有效的线索化简。

文献[76]所提出的 Protracer 系统在执行单元划分的基础上, 向与进程起源相关的系统调用中插入追踪点, 并将审计日志与污点分析相结合, 从而限定了进行日志记录的条件, 仅当发生文件写入和数据包发送事件时进行记录, 避免了冗余日志记录和存储所导致的开销。

进程执行单元的划分需要对应用程序的二进制进行分析和插桩, 这依赖于一个训练阶段, 可能产生划分不精确的情况, 且不能应对由多个处理循环组成的高级任务。因此, 有研究人员提出了一种语义感知的程序注释和插桩技术<sup>[77]</sup>, 基于应用程序特定



的高级任务结构来划分执行单元。只需让程序开发人员在源码中进行一定的注释,而后通过静态分析的方式自动划分单元即可实现。不需要训练过程,且具有更丰富的语义信息,对于攻击的描述更加清晰。

### 3.2.2 事件合并

应用程序每时每刻都在产生大量的事件,但很多事件对依赖关系分析并没有意义,有些只是应用程序的例行过程,如程序启动时读写大量的固定文件;而有些事件对于后续事件的影响是等价的。对于以上两种情况,单一的事件不一定有分析价值,逐一分析会造成大量的资源消耗,因此,可以采用事件合并的方法进行处理。

文献[78]提出了基于模板的高效数据约减方法,其发现了程序在初始启动阶段会加载大量的库文件、且所加载的文件相对固定这一事实,因此将这些文件访问行为合并成模板,以此替换大量的文件读写事件,从而减少事件依赖关系图的体量。但这种方法的作用场景有限,而且需要通过训练生成模板。

文献[39]提出了等价事件的概念,若在前向和后向追踪的时候,无论移除两个事件中的哪一个,都能产生相同的依赖关系结论,则两个事件等价,等价的两个事件可以合并为一个事件参加后续的分析。文章提出了因果保留约减算法,通过时间戳顺序将相同两个实体间发生的等价事件进行合并,当进行后向分析时将时间戳较小的事件合并到较大的事件中,反之亦然,从而在减少事件量的同时保持正确的依赖关系。

### 3.2.3 事件筛选

将冗余事件进行过滤筛选是最直接的减少事件数量的思路。在一个攻击场景中,未参与攻击过程的应用程序所产生的事件都属于冗余事件,参与攻击过程的应用程序的正常工作事件也属于冗余事件。因此,如何有效的识别这些冗余事件而保留涉及攻击环节的关键事件是事件筛选方法的核心问题。对事件是否冗余的判断方法可分为三类:基于时间戳判断,基于事件定性方法的判断以及基于事件定量方法的判断。

基于时间戳的判断通过设定时间阈值或窗口筛选事件。文献[72]最早提出了利用主机日志发现攻击的方法,通过分析日志中的事件,还原事件间依赖关系,构建依赖关系图。该方法从一个检测点开始,进行后向的依赖追踪。对于冗余事件的筛选,其提出了分析时间阈值的方法。由于依赖关系的分析是后向进行的,因此,若某事件的时间戳大于已构建依赖图中所有事件最大的时间戳,则该事件被视为冗

余事件,不会被加入依赖关系图中,是一种适用于在事后对主机日志进行分析的方法。文献[79]同样利用了时间戳思想,文章认为一个攻击步骤通常在很短的时间内完成,因此无需对时间窗口之外的行为进行分析,通过限定时间窗口,大幅减少了事件数量。

基于事件定性的方法通过事件自身属性或事件间关系属性判断其是否是冗余事件。文献[75]借鉴了内存中的垃圾收集思想,设计了具有垃圾收集功能的主机审计日志系统。该文章设计了一个对象可达性判断算法,分析对象是否与已知的异常事件有依赖关系,即该对象产生的影响是否可达到异常事件,将影响小的对象(如一些生命周期很短的临时文件)进行垃圾回收,从而减少事件数量。文献[80]在文献[72]的基础上进行了改进,向审计日志中添加了关于执行读写操作的文件偏移量的详细信息,并在分析阶段检查记录的信息。对于两个符合时间戳关系的文件读写事件,若其对该文件的操作偏移量有重叠,则认为其具有依赖关系,否则视为冗余事件进行过滤。文献[44]提出了三种事件筛选方法,包括无关节点修剪,如临时文件读写事件;同名实体合并,合并具有相同名称的主体,而不考虑它们的进程 ID 和命令行参数等属性;重复事件过滤,例如将对统一文件的多次读取或写入合并成同一事件。

基于事件定量的方法通过给事件赋予分值判断其是否是冗余事件。文献[81]为每一个事件赋予三个属性,分别为稀有度、扇出度、数据流终止。稀有度指事件自身发生的次数是否稀有,扇出度指某事件的子事件数量,数据流终止指某事件是否导致了数据流的终止,如进程终止、文件删除等。根据三个属性的得分计算事件的优先级,筛选出优先级高的事件用于攻击路径的构建,此方法既可以实现线索的化简,又能够及时的构建攻击路径,更快的发现攻击。文献[45]提出了另一种量化分析方法,通过评估依赖性强弱来进行线索约减,提出用最小祖先覆盖和路径因子来评估依赖性强弱。最小祖先覆盖指的是攻击者要完全控制信息流需要控制的祖先事件的最小数量,路径因子指两个实体间信息流的最小祖先覆盖尺寸。通过匹配路径因子数值和预先设置的规则进行事件筛选,达到减轻依赖爆炸的作用。

### 3.2.4 对象级起源追踪

随着一系列研究的开展,事件起源追踪已经从针对单一应用程序扩展到主机全局,利用事件属性、时间戳等确定和筛选依赖关系、构建和还原事件路径,但跨主机的起源追踪依旧是一个问题,尤其是在当下跨主机攻击越来越多的情况下,如何将一次

攻击行为分散在不同主机上的事件串联起来成为研究热点。

文献[82]进行了对象级起源追踪的研究, 提出了一个数据流标记和跟踪系统 RTAG, 以可在主机间传输的文件数据作为研究对象, 利用主机 mac 地址、

文件标识符、字节偏移值构成 208 位的标签, 对文件的每一个字节进行全局独一性标记, 从而得到多台主机间事件的准确起源信息, 实现跨主机攻击调查。

关于主机事件处理阶段的相关工作及文献总结于表 4。

表 4 主机事件处理阶段相关工作  
Table 4 Related Work in Host Event Processing Stage

分类	核心思想	相关文献
执行单元划分	通过在应用程序源码中插桩进行执行单元划分, 则单元的流入信息流和流出信息流都可记录明确对应的依赖关系。	[40,74-77]
事件合并	通过将多个事件合并成同一事件或事件集合看待, 减少分析事件数量。	[39,78]
事件筛选	通过筛选可能涉及攻击的关键事件、过滤于攻击发现无益的冗余事件, 减少分析事件数量。	[44-45,72, 75,79-81]
对象级起源追踪	通过对事件链中的对象进行标签设计和分配, 实现信息流追踪, 获得准确的事件间依赖关系。	[82]

3.3 主机事件分析

主机事件分析是通过对事件的串联、事件链的构建和评估而发现主机攻击的过程, 其本质是发现攻击路径。但主机中每时每刻都在产生大量的事件, 程序的正常运行和人类的正常活动都会产生大量的正常事件链, 攻击事件链隐藏其中, 而且攻击事件链上也包含正常的节点和事件, 难以精确识别。研究人员提出了基于异常量化、基于事件筛选、查询系统、攻击重放等方法。

3.3.1 基于异常量化的分析

基于异常量化的方法是指将节点、事件或事件链的异常程度用数值进行表示和计算, 通过比较数值的大小或范围为其定性, 判断是否是攻击行为。

文献[79]设计了 GID 系统, 关注实体之间的交互异常, 设计了事件接收信息量和事件发送信息量两个分数, 通过在依赖关系图上进行随机游走和状态转移来计算得分, 最后按照事件链的得分高低进行异常性的判定。

文献[83]提出了 NODOZE 系统, 首先进行依赖关系图的生成, 随后根据之前长期收集的数据中事件的发生频率对事件进行评分。本文提出了一个思路, 即事件的异常程度应该受到相邻事件的影响, 比如一个异常进程所创建的子进程比一个正常进程创建的子进程可疑性更高。基于这一思路设计了分数传播算法, 最终为整条事件链计算得出一个分数, 用于判定是否有攻击发生。

根据事件异常性量化计算得出的事件链得分可能会受到事件链长度的影响, 即事件链越长, 异常

分数可能越高。针对这一问题, 上述两个研究分别提出了解决方案, 前者用 Box-Cox 变换方法标准化异常分数, 后者使用基于采样的方法来寻找衰减因子, 消除路径长度导致的分数偏差。

3.3.2 基于事件筛选的分析

基于事件筛选的方法的核心思想是, 只使用异常的事件进行攻击路径的构建, 避免了对正常事件和正常事件链的冗余分析, 大幅提升了分析效率, 因此非常适用于实时的攻击发现场景。

文献[81]设计了三种属性对事件的优先级进行刻画, 包括稀有度、扇出度和数据流终止, 按照优先级顺序选取事件进行攻击路径构建, 那么异常事件链将优先于正常事件链构建完成, 从而实现及时的攻击发现。

3.3.3 查询系统

在安全的各个领域都存在着特定语言表达的研究, 如密码系统<sup>[84-86]</sup>、网络入侵检测<sup>[87-90]</sup>等。这些语言被设计用来解决特定领域的问题, 通过特定的构造策略, 用于形式化地表示某种事件或逻辑关系, 有助于融合和综合分析多种场景、多种设备产生的数据。

传统的信息流追踪实现需要大量的工程工作<sup>[91-92]</sup>, 而这无法应对攻击的及时发现。文献[69,93]设计实现了基于流的实时查询系统, 用户通过构造符合策略的查询语句, 可以实时检测数据流中的异常行为。通过这样的查询系统, 借助异常规则等专家知识, 仅通过几十行代码就可以实现信息流追踪、入侵检测、数据防泄漏等功能。但该技术适用于人工的异常调

查, 而非自动的异常发现。

3.3.4 攻击重放

攻击重放技术是指将主机事件等信息投放到虚拟环境中, 利用事件信息重构攻击场景和重现攻击过程, 这类方法适用于攻击发生后的具体分析, 研究重点在于更精确的事件间依赖关系还原, 以实现攻击手段或攻击特征的捕捉与分析。

文献[44]提出了基于标签的事件表示方法, 为实体分配两个标签, 包括信任度标签 **t-tag** 和机密度标签 **c-tag**, 其中机密度标签只用于描述数据, 而信任度标签有两个, 一个用于描述代码, 一个用于描述数据。标签的使用能够帮助确定分析的优先顺序和重点, 基于标签传播可以更有效的发现事件间的依

赖关系, 从而重构攻击场景和发现攻击。

文献[94]提出 **MCI** 系统, 基于双重执行技术 **LDX**<sup>[95]</sup>来推理系统调用之间的依赖关系, 生成依赖关系模型。**LDX** 的运行机制为, 给定一个原始执行作为主执行, 生成一个改变源的从执行, 比较两次执行的输出内容, 以此来判断源与输出之间的依赖关系。生成的依赖关系模型能够表示细粒度的依赖关系, 包括在系统调用级别不可见的依赖关系, 例如由内存操作引起的依赖关系。随后, **MCI** 用模型解析日志, 来识别收集的日志事件之间的依赖关系, 从而重构攻击路径。

关于主机事件分析阶段的相关工作及文献总结于表 5。

表 5 主机事件分析阶段相关工作  
Table 5 Related Work in Host Event Analysis Phase

分类	核心思想	相关文献
基于异常量化	对事件或事件链的异常程度用数值进行表示和计算, 通过比较数值的大小或范围为其定性, 判断是否是攻击行为。	[79,83]
基于事件筛选	筛选出异常事件进行攻击路径的构建, 避免冗余分析, 提高攻击发现效率。	[81]
查询系统	设计特定的查询语法, 为用户提供异常行为检测和攻击发现的查询接口。	[69,93]
攻击重放	利用主机事件信息, 在虚拟环境中重构攻击场景和重现攻击过程, 实现攻击的发现或攻击特征的捕获与分析。	[44,94]

4 未来研究方向

根据主机攻击发现的发展历程和研究现状, 我们可以提出五个未来研究方向。

1) 主机事件记录的完整性和可信性

主机事件记录的完整性和可信性是进行准确的事件间关联、事件链构建和异常事件链识别进而发现攻击的先决条件。目前大多数的相关研究利用密码学方法或借助区块链技术实现, 但由此带来的时间和空间消耗给及时的攻击发现需求带来了一定的挑战。因此, 在未来的研究中, 尤其是涉及实际应用的研究上, 主机事件记录的完整性和可信性保障、以及在记录不完整情况下攻击发现方法的鲁棒性仍然是一个研究重点。

2) 攻击发现的时效性

对于主机攻击发现来说, 最重要的是及时地发现, 在攻击实施过程中或还未造成严重损失之前发现并告警, 以采取相应的防范措施, 若不能及时发现攻击, 那么攻击发现的意义就损失了大半。目前很多研究团队提出的方案适用于攻击发生后对事件进

行分析、对攻击场景进行重构。也有很多研究致力于突破攻击发现的时效性问题, 例如对线索进行化简以减少处理时间和分析复杂度, 或通过实时事件分析筛选的方式构建事件链等。未来如何更高效的分析事件、如何实现实时而精确的攻击发现还需要进一步的研究。

3) 跨设备的攻击发现

攻击的行为轨迹往往经过多台设备。以一个针对企业内网的窃密型攻击为例, 攻击者通常会先控制内网中一台主机, 然后以该主机为跳板, 在内网中横向移动, 直到入侵到重要机器上, 实施窃密行为, 随后将机密数据传输到内网中的另一台主机上, 再从该主机回传到自己的服务器上。因此仅对单一主机进行日志事件分析将无法还原攻击全貌, 需要将多台主机上的事件加以融合分析, 才能够构建出完整的攻击路径。如何对多台设备的日志数据和事件进行有效的整合, 同时又不影响攻击发现的实时性, 是未来的一个研究重点。

4) 多步骤攻击的攻击发现

对于以 **APT** 为代表的多步骤攻击, 其各个步骤

之间可能独立性较强,存在着较长的时间间隔,或不发生在同一主机上,因此其关联性难以轻易挖掘,单一步骤的异常性也难以精确判定。进行多步骤之间的关联,尤其是跨时间、跨设备的多步骤关联以发现攻击需要受到研究人员的关注。目前,有学者提出了借助 APT 模型将事件链映射到攻击步骤的方法<sup>[45]</sup>,基于 ATT&CK 框架<sup>[96]</sup>构建一个中间映射层,ATT&CK 框架描述了近 200 种行为模式,每种模式定义为一个“战术,技术,过程”(Tactics, Techniques, and Procedures, TTP),每个 TTP 都可以与杀伤链中的一个阶段的行为相对应,从而将日志数据最终映射到 APT 攻击的某一阶段。此外,长时间间隔攻击行为的存在也对日志数据的长期存储能力提出了要求。

#### 5) 算法的运用

越来越多的算法被应用于信息安全领域。借助数学算法,将问题转化为数学模型,可以有效地对问题进行量化求解,用数字表征关系、状态、可能性、程度等属性。对于基于主机事件的攻击发现技术,其所研究的事件链是一个链式结构,事件链由各个节点及其之间的依赖关系构成,其本质是信息的流动,可以将节点视为信息的某一个状态,而信息的流动则为状态的转移。因此,事件链天然地可以与隐马尔可夫模型相匹配。另外,事件链由多个节点连接构成,而一个节点可能不只属于一条事件链,而是与多个其他节点具有依赖关系,就导致了多条事件链相互交织,构成了依赖关系网络。有很多数学模型都可以用来解决图问题或网络问题,如矩阵模型、动态贝叶斯网络等。文献[79]和文献[83]建立了状态转移矩阵,用于记录节点间的状态转移概率,来实现节点间的分数传播。在未来的研究中,更多的将数学算法与模型结合到基于主机事件的攻击发现领域是一个重要的方向。

近年来机器学习和深度学习算法逐渐兴起,被广泛应用于信息安全的各个领域,在恶意性检测、恶意家族分类等领域取得了很多优秀成果,但在基于主机事件的攻击发现领域还未被广泛使用。算法依赖大量的训练数据作为支撑,且目前效果越好的算法,其需要的数据数量、数据质量和设备运算能力越高。对于基于主机事件的攻击发现来说,正常数据量远大于异常数据量,攻击路径中既包含正常事件也包含异常事件,攻击行为复杂多变、模式难以抽象,且算法运行需要大量时间,这为及时而准确的主机攻击发现提出了严峻挑战,也是未来的一个研究重点。

## 5 总结

主机是网络攻击中的主要目标,主机攻击发现对于保护我们的工作生活具有重要意义,而主机事件记录了主机中发生的所有行为,因此基于主机事件的攻击发现技术研究受到学者的广泛关注。本文从相关研究的发展历程和关键概念入手,整理汇总了当前的研究现状,并基于研究现状和依赖关系爆炸、及时性两大关键问题,从主机事件采集、主机事件处理、主机事件分析三个阶段,分别介绍了研究人员在这三个阶段提出的各类研究成果和工作。最后,本文对未来的研究进行了展望,提出了主机事件记录的完整性和可信性、攻击发现时效性、跨设备攻击的发现、多步骤攻击的发现以及算法的运用五个未来研究方向。

## 参考文献

- [1] Kaspersky Security Bulletin 2019. Statistics. Kaspersky. [https://go.kaspersky.com/rs/802-IJN-240/images/KSB\\_2019\\_Statistics\\_EN.pdf](https://go.kaspersky.com/rs/802-IJN-240/images/KSB_2019_Statistics_EN.pdf). Jan. 2020.
- [2] Hu D Y, Min J H. *Networksecurity*[M]. Beijing: Tsinghua University Press, 2004.  
(胡道元, 闵京华. 网络安全[M]. 北京: 清华大学出版社, 2004.)
- [3] Anderson J P. Computer security threat monitoring and surveillance. Technical Report. James P. Anderson Company. 1980.
- [4] Ilgun K, Kemmerer R A, Porras P A. State Transition Analysis: A Rule-Based Intrusion Detection Approach[J]. *IEEE Transactions on Software Engineering*, 1995, 21(3): 181-199.
- [5] Ilgun K, Processing C A. USTAT: a real-time intrusion detection system for UNIX[C]. *Proceedings 1993 IEEE Computer Society Symposium on Research in Security and Privacy*, 2002: 16-28.
- [6] Modi C, Patel D, Borisaniya B, et al. A Survey of Intrusion Detection Techniques in Cloud[J]. *Journal of Network and Computer Applications*, 2013, 36(1): 42-57.
- [7] Singh K, Guntuku S C, Thakur A, et al. Big Data Analytics Framework for Peer-to-Peer Botnet Detection Using Random Forests[J]. *Information Sciences*, 2014, 278: 488-497.
- [8] Huang M L, Lu L F, Zhang X Y. Using Arced Axes in Parallel Coordinates Geometry for High Dimensional BigData Visual Analytics in Cloud Computing[J]. *Computing*, 2015, 97(4): 425-437.
- [9] Doroudian M, Arastouie N, Talebi M, et al. Multilayered database intrusion detection system for detecting malicious behaviors in big data transaction[C]. *2015 Second International Conference on Information Security and Cyber Forensics*, 2016: 105-110.
- [10] Barapatre P, Tarapore N Z, Pukale S G, et al. Training MLP neural network to reduce false alerts in IDS[C]. *2008 International Conference on Computing, Communication and Networking*, 2009: 1-7.
- [11] Hu J H, Processing C A. Network intrusion detection algorithm based on improved support vector machine[C]. *2015 International Conference on Intelligent Transportation, Big Data and Smart City*, 2016: 523-526.

- [12] Hinton G E, Osindero S, Teh Y W. A Fast Learning Algorithm for Deep Belief Nets[J]. *Neural Computation*, 2006, 18(7): 1527-1554.
- [13] Tang T A, Mhamdi L, McLernon D, et al. Deep learning approach for Network Intrusion Detection in Software Defined Networking[C]. *2016 International Conference on Wireless Networks and Mobile Communications*, 2016: 258-263.
- [14] Javaid A, Niyaz Q, Sun W Q, et al. A Deep Learning Approach for Network Intrusion Detection System[C]. *The 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, 2016: 21-26.
- [15] Yin C L, Zhu Y F, Fei J L, et al. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks[J]. *IEEE Access*, 5: 21954-21961.
- [16] Servin A, Kudenko D. Multi-Agent Reinforcement Learning for Intrusion Detection[M]. *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008: 211-223.
- [17] Otoum S, Kantarci B, Mouftah H, et al. Empowering reinforcement learning on big sensed data for intrusion detection[C]. *ICC 2019 - 2019 IEEE International Conference on Communications*, 2019: 1-7.
- [18] Yin X. *Network intrusion detection based on improved dictionary learning*[D]. Nanjing: Nanjing University of Posts and Telecommunications, 2018.  
(尹秀. 基于改进的字典学习的网络入侵检测方法研究[D]. 南京: 南京邮电大学, 2018.)
- [19] Sun B. *Research on Key Problems of Computer Forensic Methods*[D]. Beijing: Institute of Software, Chinese Academy of Sciences, 2004.  
(孙波. 计算机取证方法关键问题研究[D]. 北京: 中国科学院研究生院(软件研究所), 2004.)
- [20] Chen L, Wang G Y. Survey of Computer Forensics[J]. *Journal of Chongqing University of Posts and Telecommunications*, 2005, 17(6): 736-741.  
(陈龙, 王国胤. 计算机取证技术综述[J]. *重庆邮电学院学报(自然科学版)*, 2005, 17(6): 736-741.)
- [21] Xu R S, Wu H Y, Liu B X. Overview of Computer Forensics. *Journal of Computer Engineering and Applications*. 2001. 37(21): 7-8.  
(许榕生, 吴海燕, 刘宝旭. 计算机取证概述. *计算机工程与应用*. 2001. 37(21): 7-8.)
- [22] Zhang Y, Liu Q Z, Li T, et al. Research and development of memory forensics. *Journal of Software*. 2015, 26(5): 1151-1172.  
(张瑜, 刘庆中, 李涛, 等. 内存取证研究与进展. *软件学报*. 2015. 26(5): 1151-1172.)
- [23] Pu H Q, Guo Y F, Wei B G. Survey on Electronic Forensics Research[J]. *Computer Systems & Applications*. 2019, 28(1): 10-16.  
(蒲泓全, 郭艳芬, 卫邦国. 电子取证应用研究综述. *计算机系统应用*. 2019. 28(1): 10-16.)
- [24] Cheng X K, Wu C B. Forensic Analysis of File-FreeAttack[J]. *Telecom World*, 2019, 26(8): 19-20.  
(成星恺, 吴崇斌. 无文件型攻击取证分析[J]. *通讯世界*, 2019, 26(8): 19-20.)
- [25] Pringle N, Burgess M. Information Assurance in a Distributed Forensic Cluster[J]. *Digital Investigation*, 2014, 11: S36-S44.
- [26] Chen G X, Du Y H, Du J, et al. Research of Digital Forensics under Cloud Computing Environment[J]. *Netinfo Security*, 2013(8): 87-90.  
(陈光宣, 杜彦辉, 杜锦, 等. 云环境下电子取证研究[J]. *信息安全*, 2013(8): 87-90.)
- [27] Shang L. Survey on Forensics Techniques of Electronic Evidence in Blockchain[J]. *Office Informatization*, 2019, 24(10): 33-34.  
(尚蕾. 基于区块链技术的电子数据取证研究综述[J]. *办公自动化*, 2019, 24(10): 33-34.)
- [28] Lu J Z, Lv F M, Zhuo Z L, et al. Integrating Traffics with Network Device Logs for Anomaly Detection[J]. *Security and Communication Networks*, 2019, 2019: 1-10.
- [29] Wibisono S R, Kistijantoro A I, Bioengineering, et al. Log anomaly detection using adaptive universal transformer[C]. *2019 International Conference of Advanced Informatics: Concepts, Theory and Applications*, 2020: 1-6.
- [30] Brown A, Tuor A, Hutchinson B, et al. Recurrent Neural Network Attention Mechanisms for Interpretable System Log Anomaly Detection[C]. *The First Workshop on Machine Learning for Computing Systems*, 2018: 1-8.
- [31] Zhang X, Xu Y, Lin Q W, et al. Robust Log-Based Anomaly Detection on Unstable Log Data[C]. *The 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019: 807-817.
- [32] Meng W B, Liu Y, Zhu Y C, et al. Log anomaly: Unsupervised Detection of Sequential and Quantitative Anomalies in Unstructured Logs[C]. *The 28th International Joint Conference on Artificial Intelligence*, 2019: 4739-4745.
- [33] Event tracing for windows (ETW). [http://msdn.microsoft.com/en-us/library/windows/desktop/aa363668\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/aa363668(v=vs.85).aspx).
- [34] Linux audit. <https://linux.die.net/man/8/auditd>.
- [35] Anthem cyberattack. <http://abcnews.go.com/Business/anthem-cyber-attack-things-happen-personal-information/story?id=28747729>. Feb. 2015.
- [36] Case study: The Home Depot data breach. <https://www.sans.org/reading-room/whitepapers/casestudies/case-study-home-depot-data-breach-36367>. Jan. 2015.
- [37] OPM government data breach impacted 21.5 million. <http://www.cnn.com/2015/07/09/politics/office-of-personnel-management-data-breach-20-million>. Jul. 2015.
- [38] Stuxnet. <https://en.wikipedia.org/wiki/Stuxnet>.
- [39] Xu Z, Wu Z Y, Li Z C, et al. High Fidelity Data Reduction for Big Data Security Dependency Analyses[C]. *The 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016: 504-516.
- [40] Ma S Q, Lee K H, Kim C H, et al. Accurate, low cost and instrumentation-free security audit logging for windows[C]. *The 31st Annual Computer Security Applications Conference*, 2015: 401-410.
- [41] Pywintrace. <https://github.com/fireeye/pywintrace>.
- [42] Fibratus. <https://github.com/rabbitstack/fibratus>.
- [43] Transparent computing engagement 3 Data Release. <https://github.com/darpa-i2o/Transparent-Computing>.
- [44] Hossain M N, Milajerdi S M, Wang J N, et al. SLEUTH: Real-Time Attack Scenario Reconstruction from COTS Audit Data[C]. *The 26th USENIX Conference on Security Symposium*, 2017: 487-504.

- [45] Milajerdi S M, Gjomemo R, Eshete B, et al. HOLMES: real-time APT detection through correlation of suspicious information flows[C]. *2019 IEEE Symposium on Security and Privacy*, 2019: 1137-1152.
- [46] Ma S Q, Zhai J, Kwon Y, et al. Kernel-Supported Cost-Effective Audit Logging for Causality Tracking[C]. *The 2018 USENIX Conference on Usenix Annual Technical Conference*, 2018: 241-253.
- [47] Ragib H, Sion R, Winslett M. The case of the fake picasso: Preventing history forgery with secure provenance[C]. *The 7th USENIX Conference on File and Storage Technologies*. 2009.
- [48] Macko P, Seltzer M. A General-Purpose Provenance Library[C]. *The 4th USENIX conference on Theory and Practice of Provenance*, 2012: 6.
- [49] Zhou W C, Fei Q, Narayan A, et al. Secure Network Provenance[C]. *The Twenty-Third ACM Symposium on Operating Systems Principles*, 2011: 295-310.
- [50] Foster I, Vockler J, Wilde M, et al. Chimera: a virtual data system for representing, querying, and automating data derivation[C]. *Proceedings 14th International Conference on Scientific and Statistical Database Management*, 2002: 37-46.
- [51] Widom J. Trio: A system for integrated management of data, accuracy, and lineage[R]. Stanford InfoLab, 2004.
- [52] Muniswamy-Reddy K K, Holland D A, Braun U, et al. Provenance-Aware Storage Systems[C]. *The annual conference on USENIX'06 Annual Technical Conference*, 2006: 4.
- [53] Sar C, Cao P. Lineage File System. <http://crypto.stanford.edu/~cao/lineage.html>.
- [54] Pohly D J, McLaughlin S, McDaniel P, et al. Hi-Fi: Collecting High-Fidelity Whole-System Provenance[C]. *The 28th Annual Computer Security Applications Conference*, 2012: 259-268.
- [55] Pasquier T, Han X Y, Goldstein M, et al. Practical Whole-System Provenance Capture[C]. *The 2017 Symposium on Cloud Computing*, 2017: 405-418.
- [56] Pasquier T, Han X Y, Moyer T, et al. Runtime Analysis of Whole-System Provenance[C]. *The 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018: 1601-1616.
- [57] Wang F, Kwon Y, Ma S Q, et al. Lprov: practical library-aware provenance tracing[C]. *The 34th Annual Computer Security Applications Conference*, 2018: 605-617.
- [58] Snodgrass R T, Yao S S, Collberg C. Tamper Detection in Audit Logs[C]. *The Thirtieth international conference on Very large data bases - Volume 30*, 2004: 504-515.
- [59] Ray I, Belyaev K, Strizhov M, et al. Secure Logging as a Service—Delegating Log Management to the Cloud[J]. *IEEE Systems Journal*, 2013, 7(2): 323-334.
- [60] Ma D, Tsudik G. A New Approach to Secure Logging[J]. *ACM Transactions on Storage*, 2009, 5(1): 2.
- [61] Sutton A, Samavi R. Blockchain Enabled Privacy Audit Logs[M]. *Lecture Notes in Computer Science*. Cham: Springer International Publishing, 2017: 645-660.
- [62] Cucurull J, Puiggali J. Distributed Immutabilization of Secure Logs[M]. *Security and Trust Management*. Cham: Springer International Publishing, 2016: 122-137.
- [63] Bates A, Tian D, Butler K R B, et al. Trustworthy Whole-System Provenance for the Linux Kernel[C]. *The 24th USENIX Conference on Security Symposium*, 2015: 319-334.
- [64] Wright C, Cowan C, Morris J, et al. Linux security module framework[C]. *Ottawa Linux Symposium*. 2002, 8032: 6-16.
- [65] Wright C, Cowan C, Morris J, et al. Linux security modules: General security support for the linux kernel[C]. *Foundations of Intrusion Tolerant Systems*, 2003, 2004: 213-226.
- [66] The Netfilter Project: Packet Mangling for Linux 2.4. The Netfilter Core Team. <http://www.netfilter.org/>. 1999.
- [67] Karande V, Bauman E, Lin Z Q, et al. SGX-Log: Securing System Logs with SGX[C]. *The 2017 ACM on Asia Conference on Computer and Communications Security*, 2017: 19-30.
- [68] Garera S, Rubin A D. An Independent Audit Framework for Software Dependent Voting Systems[C]. *The 14th ACM conference on Computer and communications security*, 2007: 256-265.
- [69] Huh J H, Martin A, Communication N A B T, et al. Trusted logging for grid computing[C]. *2008 Third Asia-Pacific Trusted Infrastructure Technologies Conference*, 2008: 30-42.
- [70] Quynh N A, Takefuji Y. A Central and Secured Logging Data Solution for Xen Virtual Machine[C]. *The 24th IASTED international conference on Parallel and distributed computing and networks*, 2006: 218-224.
- [71] Goel A, Po K, Farhadi K, et al. The Taser Intrusion Recovery System[C]. *The twentieth ACM symposium on Operating systems principles*, 2005: 163-176.
- [72] King S T, Chen P M. Backtracking Intrusions[C]. *The nineteenth ACM symposium on Operating systems principles*, 2003: 223-236.
- [73] King S T, Mao Z M, Lucchetti D G, et al. Enriching intrusion alerts through multi-host causality[C]. *The Network and Distributed System Security Symposium*. 2005.
- [74] Lee K H, Zhang X, Xu D. High Accuracy Attack Provenance via Binary-based Execution Partition[C]. *The Network and Distributed System Security Symposium*. 2013.
- [75] Lee K H, Zhang X Y, Xu D Y. LogGC: Garbage Collecting Audit Log[C]. *The 2013 ACM SIGSAC conference on Computer & communications security*, 2013: 1005-1016.
- [76] Ma S Q, Zhang X Y, Xu D Y. ProTracer: towards practical provenance tracing by alternating between logging and tainting[C]. *Proceedings 2016 Network and Distributed System Security Symposium*, 2016: 2-4.
- [77] Ma S Q, Zhai J, Wang F, et al. MPI: Multiple Perspective Attack Investigation with Semantics Aware Execution Partitioning[C]. *The 26th USENIX Conference on Security Symposium*, 2017: 1111-1128.
- [78] Tang Y T, Li D, Li Z C, et al. NodeMerge: Template Based Efficient Data Reduction for Big-Data Causality Analysis[C]. *The 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018: 1324-1337.
- [79] Dong B X, Chen Z Z, Wang H, et al. GID: Graph-Based Intrusion Detection on Massive Process Traces for Enterprise Security Systems[EB/OL]. 2016: arXiv: 1608.02639. <https://arxiv.org/abs/1608.02639>
- [80] Sitaraman S, Venkatesan S, Communication N A B T, et al. Forensic analysis of file system intrusions using improved backtracking[C]. *Third IEEE International Workshop on Information Assurance*, 2005: 154-163.
- [81] Liu Y, Zhang M, Li D, et al. Towards a Timely Causality Analysis for Enterprise Security. *The Network and Distributed System Security Symposium*, 2018.

- [82] Ji Y, Lee S, Fazzini M, et al. Enabling Refinable Cross-Host Attack Investigation with Efficient Data Flow Tagging and Tracking[C]. *The 27th USENIX Conference on Security Symposium*, 2018: 1705-1722.
- [83] Hassan W U, Guo S J, Li D, et al. NoDoze: combatting threat alert fatigue with automated provenance triage[C]. *Proceedings 2019 Network and Distributed System Security Symposium*, 2019.
- [84] Bhargavan K, Corin R, Deniélou P M, et al. Cryptographic protocol synthesis and verification for multiparty Sessions[C]. *2009 22nd IEEE Computer Security Foundations Symposium*, 2009: 124-140.
- [85] Bhargavan K, Fournet C, Corin R, et al. Cryptographically Verified Implementations for TLS[C]. *The 15th ACM conference on Computer and communications security*, 2008: 459-468.
- [86] Liu C, Wang X S, Nayak K, et al. OblivM: A programming framework for secure computation[C]. *2015 IEEE Symposium on Security and Privacy*, 2015: 359-376.
- [87] Borders K, Springer J, Burnside M. Chimera: A Declarative Language for Streaming Network Traffic Analysis[C]. *The 21st USENIX conference on Security symposium*, 2012: 19.
- [88] Cuppens F, Ortalo R. LAMBDA: A Language to Model a Database for Detection of Attacks[M]. *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000: 197-216.
- [89] Sommer R, Vallentin M, De Carli L, et al. HILTI: An Abstract Execution Environment for Deep, Stateful Network Traffic Analysis[C]. *The 2014 Conference on Internet Measurement Conference*, 2014: 461-474.
- [90] Vallentin M, Paxson V, Sommer R. VAST: A Unified Platform for Interactive Network Forensics[C]. *The 13th Usenix Conference on Networked Systems Design and Implementation*, 2016: 345-362.
- [91] Krohn M, Yip A, Brodsky M, et al. Information Flow Control for Standard OS Abstractions[C]. *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, 2007: 321-334.
- [92] Roy I, Porter D E, Bond M D, et al. Laminar: Practical Fine-Grained Decentralized Information Flow Control[C]. *The 30th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2009: 63-74.
- [93] Gao P, Xiao X S, Li D, et al. SAQL: A Stream-Based Query System for Real-Time Abnormal System Behavior Detection[C]. *The 27th USENIX Conference on Security Symposium*, 2018: 639-656.
- [94] Kwon Y, Wang F, Wang W H, et al. MCI: modeling-based causality inference in audit logging for attack investigation[C]. *Proceedings 2018 Network and Distributed System Security Symposium*, 2018: 2: 4.
- [95] Kwon Y, Kim D, Sumner W N, et al. LDX: Causality Inference by Lightweight Dual Execution[C]. *The Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, 2016: 503-515.
- [96] ATT&CK. MITRE. [https://attack.mitre.org/wiki/Main\\_Page](https://attack.mitre.org/wiki/Main_Page).



冯云 于 2016 年在湖南大学信息安全专业获得学士学位。现在中国科学院大学网络空间安全学院攻读博士学位。主要研究领域为网络攻防技术、网络攻击追踪溯源。Email: fengyun@iie.ac.cn



刘宝旭 于 2002 年在中国科学院研究生院获得博士学位。现任中国科学院信息工程研究所研究员、中国科学院大学网络空间安全学院教授。主要研究领域为网络安全攻防对抗、态势感知、威胁情报、溯源取证等。Email: liubaoxu@iie.ac.cn



张金莉 于 2016 年在北京邮电大学信息安全专业获得硕士学位。现任中国科学院信息工程研究所助理研究员。主要研究领域为网络攻防技术、网络攻击追踪溯源。Email: zhangjinli@iie.ac.cn



张越 于 2017 年在山西大学计算机技术专业获得硕士学位。现任中国科学院信息工程研究所助理研究员。主要研究领域为网络攻防技术、网络攻击追踪溯源。Email: zhangyue@iie.ac.cn



刘奇旭 于 2011 年在中国科学院研究生院信息安全专业获得博士学位。现任中国科学院信息工程研究所副研究员、中国科学院大学网络空间安全学院副教授。主要研究领域为网络攻防技术、网络安全评测。Email: liuqixu@iie.ac.cn