软件学报 ISSN 1000-9825, CODEN RUXUEW
[doi: 10.13328/j.cnki.jos.007385] [CSTR: 32375.14.jos.007385]

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

基于可编程交换机的网络安全研究进展*

邹志凯[1,2], 张梦豪[1,2,3], 李冠宇[4], 杨任宇[1,2], 沃天宇[1,2,3], 胡春明[1,2,3], 徐明伟[3,5]

[1](北京航空航天大学 软件学院, 北京 100191)

[2](复杂关键软件环境全国重点实验室 (北京航空航天大学), 北京 100191)

[3](中关村实验室, 北京 100081)

[4](华为技术有限公司, 北京 100094)

[5](清华大学 计算机系, 北京 100084)

通信作者: 张梦豪, E-mail: zhangmenghao@buaa.edu.cn

摘　要: 随着云计算、移动互联网和人工智能等网络应用的快速发展, 网络攻击和威胁日益增多和复杂化, 这要求网络安全防御技术能够有效地防御网络攻击, 保障关键基础设施网络的安全. 传统基于专有中间件的防御技术能够通过专有硬件实现高性能, 然而这些防御措施成本高昂, 部署新的防御通常需要升级设备. 基于软件的防御技术非常灵活, 但是基于软件的数据包处理会导致较高的性能开销. 可编程交换机的出现则为网络安全防御带来了新的契机, 由于其在灵活性和性能上的显著优势, 基于可编程交换机的网络安全研究已经成为近期的研究热点之一. 首先回顾可编程交换机的起源和架构, 然后深入探讨其相关特性在网络安全防御中的应用和优势, 包括易于管理、低成本、高灵活性和高性能. 接着, 从网络安全防御的基本三元组即预防、检测和响应的角度出发, 系统阐述了利用可编程交换机进行网络安全防御的技术, 包括访问控制、网络扫描、网络混淆、深度数据包检查、DDoS 检测与防御、智能数据平面等多个方面, 并且分析了这些技术的设计理念、实现机制和潜在局限性. 最后, 对基于可编程交换机的网络安全研究的未来发展方向进行了展望.

关键词: 网络安全; 可编程交换机; 预防; 检测; 响应

中图法分类号: TP393

中文引用格式: 邹志凯, 张梦豪, 李冠宇, 杨任宇, 沃天宇, 胡春明, 徐明伟. 基于可编程交换机的网络安全研究进展. 软件学报. http://www.jos.org.cn/1000-9825/7385.htm

英文引用格式: Zou ZK, Zhang MH, Li GY, Yang RY, Wo TY, Hu CM, Xu MW. Research Advances in Programmable Switches Driven Network Security. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7385.htm

Research Advances in Programmable Switches Driven Network Security

ZOU Zhi-Kai[1,2], ZHANG Meng-Hao[1,2,3], LI Guan-Yu[4], YANG Ren-Yu[1,2], WO Tian-Yu[1,2,3], HU Chun-Ming[1,2,3], XU Ming-Wei[3,5]

[1](School of Software, Beihang University, Beijing 100191, China)

[2](State Key Laboratory of Complex & Critical Software Environment (Beihang University), Beijing 100191, China)

[3](Zhongguancun Laboratory, Beijing 100081, China)

[4](Huawei Technologies Co. Ltd., Beijing 100094, China)

[5](Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

Abstract: With the rapid growth of network applications such as cloud computing, mobile internet, and artificial intelligence, network

* 基金项目: 国家重点研发计划 (2022YFB4502003); 国家自然科学基金 (62402025, 62402024); 中央高校基本科研业务费专项资金; 北京市自然科学基金 (L241050)
收稿时间: 2024-06-25; 修改时间: 2024-09-04; 采用时间: 2024-12-23; jos 在线出版时间: 2025-06-25

attacks and threats are becoming increasingly frequent and complex. This necessitates the development of network security defense technologies capable of effectively countering these threats and ensuring the security of critical infrastructure networks. Traditional defense technologies based on middleboxes can achieve high performance using specialized hardware; however, these solutions are costly, and deploying new defenses typically requires hardware upgrades. Software-based defense technologies offer high flexibility, but software-based packet processing leads to significant performance overhead. The emergence of programmable switches presents new opportunities for network security defense by offering notable advantages in both flexibility and performance, making this a prominent research focus. This study first reviews the origin and architecture of programmable switches and explores their relevant features and advantages in network security applications, including ease of management, low cost, high flexibility, and high performance. Subsequently, from the perspective of the basic triad of network security defense, namely prevention, detection, and response, this study systematically elaborates on various defense techniques utilizing programmable switches, such as access control, network scanning, network obfuscation, deep packet inspection, DDoS detection and mitigation, and intelligent data planes. The design principles, implementation mechanisms, and potential limitations of these technologies are analyzed. Finally, an outlook is provided on future research directions for network security based on programmable switches.

Key words: network security; programmable switches; prevention; detection; response

1 引 言

1.1 可编程交换机的来源

2008 年, 斯坦福大学教授 Nick McKeown 及其团队在 Ethane 项目[1]的基础上, 于 SIGCOMM 2008 会议上发表了题为“OpenFlow: Enabling Innovation in Campus Networks“的论文[2]. 这篇论文首次全面阐述了软件定义网络 (software-defined networking, SDN) 的核心理念, 即把传统网络设备中的数据转发功能 (数据平面) 与控制逻辑功能 (控制平面) 分离开来. 通过这种分离, 控制器可以通过统一且标准化的 OpenFlow 接口对数据平面的网络设备进行管理和配置, 从而实现了网络的集中管控, 提高了网络资源管理和使用的灵活性.

从体系结构角度来看, SDN 主要分为 3 个平面: 应用平面、控制平面和数据平面. 应用平面执行网络应用, 控制平面根据网络应用生成的请求下发和更新整个网络的规则, 并且基于设定的规则来配置数据平面中的交换机. 数据平面中的网络设备 (路由器和交换机) 根据控制器给出的指令进行数据包的转发. 其中, 控制平面和数据平面之间通过 OpenFlow 协议进行通信, 从而实现控制与转发分离, 为网络管控提供统一的编程模型. OpenFlow 的核心功能在于其匹配动作表能够识别并匹配多种数据包头部信息, 例如 MAC 地址、IP 地址、协议号、TCP/UDP 端口号等, 并执行相应的动作. 但是这个匹配域是与协议相关的, 交换机想要向控制平面暴露更多的能力就需要支持更多的头部字段和匹配动作表, 这也使得规范变得越来越复杂, 从 1.0 版本的 12 个匹配域到 1.3 版本的 40 个匹配域再到 1.5 版本的 45 个匹配域, 而且这一扩展仍然没有停止的趋势[3]. 由于 OpenFlow 协议在定制匹配字段时的灵活性不足, 每当增加新的匹配字段时, 都需要对控制器程序、交换机协议及数据包处理逻辑进行重写, 这不仅增加了交换机软硬件设计的复杂度, 也严重限制了 OpenFlow 协议的可扩展性.

为了解决 OpenFlow 协议设计上的扩展性限制, 文献 [3] 提出了一种新型的、与协议无关的可编程语言 P4, 文献 [4, 5] 也给出了相应的转发模型, 推动了可编程交换机的出现和发展. P4 语言赋予了数据平面强大的编程能力, 普通开发人员可以灵活地对交换机中的数据包进行定制化处理. 2016 年, Barefoot 发布了 Tofino, 标志着商用可编程硬件交换机的出现, 后续几年 Broadcom、Cisco 相继发布了各自的可编程硬件交换机, 交换机可编程的趋势蔚然成型. 可编程交换机允许管理员通过网络编程语言如 P4 直接编程数据包处理逻辑, 不仅实现如路由器、防火墙等传统网络设备功能, 还能轻松支持各种新协议, 例如 VXLAN、NVGRE 等. 进一步地, 可编程交换机数据平面还支持实现大象流检测[6–8]、负载均衡[9,10]、键值对存储[11]、在网计算[12]等功能, 显著提高了网络的性能和灵活性. 可编程交换机的出现也给网络安全研究带来了新的机遇.

1.2 可编程交换机的架构

综述利用可编程交换机进行网络安全防御的策略和方法是本文的重点, 而非深入探讨可编程交换机的技术细

节, 所以本文仅简单介绍基于经典 RMT (reconfigurable match-action table) 架构[4]的可编程交换机. 如图 1 所示, 基于 RMT 架构的交换机流水线设计遵循多级流水线数据包处理流程. 网络数据包首先进入解析器进行数据包头部解析, 解析器可以提取自定义的头部格式并将解析到的头部信息和相关的元数据记录在数据包头向量 (packet header vectors, PHV) 中, 然后 PHV 被传递到入口流水线进行下一步处理. 可编程交换机中有多个入口和出口流水线, 每个流水线都有多个入口端口和出口端口. 当数据包到达其中一个入口端口时, 它首先由入口流水线处理, 然后转发到其中一个出口流水线进行处理, 最后发送到指定的出口端口. 在流水线内部, 数据包在每个阶段顺序进行处理, 每个阶段都有自己的专用资源, 如匹配动作表 (match-action table, MAT)、寄存器和用于计算的有状态算术逻辑单元 (arithmetic logic unit, ALU). 匹配动作表匹配数据包的某些头部字段或元数据, 并根据匹配结果执行相应的动作, 例如修改头部字段/元数据、读写寄存器或丢弃数据包. 寄存器用于存储必要的数据或中间状态, 以实现有状态的数据包处理. 程序员可以通过使用类似 P4 这样的语言来定义数据包头部的解析方式, 构建数据包处理逻辑, 并指定每个匹配动作表的匹配字段和动作. 然后, 通过交换机供应商提供的编译器将程序编译成二进制文件并加载到交换机芯片中, 实现自定义逻辑的线速数据包处理. 这样的设计使得交换机在保证高性能的同时可以执行特定的数据包处理任务, 满足各类网络功能的需求. 值得注意的是, 目前可编程交换机 ASIC 也存在着一些资源限制, 主要包括: 1) 流水线的数量以及每个流水线中阶段和端口的数量. 2) 每个阶段可以访问的三态内容寻址存储器 (ternary content addressable memory, TCAM) 和静态随机存取存储器 (static random access memory, SRAM) 的数量是有限的. 前者主要用于匹配动作表的通配符和前缀匹配. 后者用于前缀匹配和寄存器. 3) 读写寄存器也必须满足一些约束. 首先, 一个程序只能从同一阶段的表和动作中访问寄存器数组. 其次, 一个阶段中的所有寄存器必须并行访问. 最后, 每个寄存器数组在每个数据包中只能访问一次且 ALU 只能执行简单功能, 比如同时读/写、条件更新和基本算术运算.

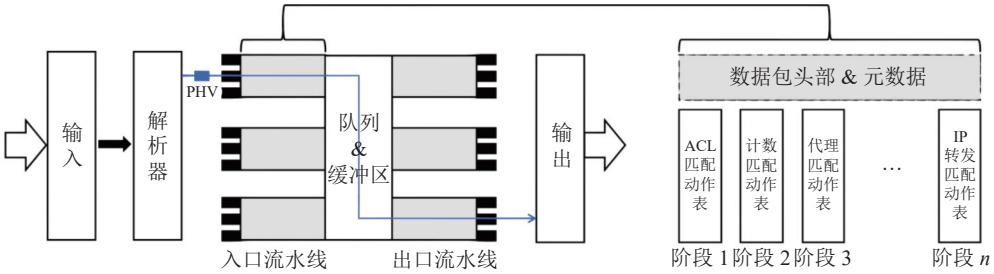


图 1 RMT 可编程交换机架构图

因此, 可编程交换机一方面以其优异的灵活性和高性能给网络安全防御技术带来了新的机会, 另一方面, 其计算模型和芯片资源方面也有着自己的限制. 研究人员必须在这些资源限制的基础上, 提出严谨周密的方案从而更好地利用可编程交换机实现网络安全防御功能.

1.3 可编程交换机的特性对网络安全的意义

传统基于 SDN 的网络安全技术虽然易于管理, 灵活性较高, 管理员可以根据需求修改其安全策略以处理动态攻击, 但是这些好处在数据包处理性能方面做出了很大妥协[13–15]. 这类软件化的安全防御技术在吞吐量和数据包处理延迟方面带来了显著的性能开销. 例如当 DDoS (distributed denial-of-service) 流量达到每秒几百 Gb 甚至每秒几 Tb 时, 管理员需要分配众多服务器和 CPU 核才能处理这种高容量攻击流量, 这样不可避免地增加了资金支出和运营费用. 同时, 基于软件的数据包处理也会使每个数据包处理带来几百微秒到几十毫秒不等的额外延迟[16,17], 这对于交易和在线支付这类对于延迟极为敏感的网络应用来说是不可接受的. 而近年来可编程交换机的出现则有效改善了这些缺点, 为实现易于配置和管理、低成本、高灵活性和高性能的网络安全技术带来了新的机遇.

与传统交换机不同, 可编程交换机可以直接通过网络编程语言如 P4 定制数据包的处理逻辑, 这种可编程性支持将安全功能从基于软件的通用服务器上卸载到基于硬件的交换芯片上, 为网络数据包的处理带来了极大的便

利. 具体来说, 可编程交换机相关特性及对于网络安全的意义如下.

(1) 易于管理. 可编程交换机可以通过类似 P4 的网络编程语言来定义和实现指定的安全功能, 而且这样的网络编程语言还提供了完整的编译工具链[18], 使这些自定义的安全策略可以高效地部署到交换机上. 开发人员可以像编写高级编程语言如 C 或 C++一样, 专注于实现网络安全的策略和功能, 而无需深入了解底层的硬件细节. 因此, 可编程交换机极大简化了网络安全功能的部署和管理.

(2) 成本低. 基于可编程交换机的网络安全防御的成本效益包括两部分: 设备成本 (以美元计) 和功耗 (以瓦特计). 据已有调查显示[19], 一个典型的 48 Gb/s 的 DDoS 防御中间件的成本约为 102 550 美元, 功耗为 600 W; 一个配备 40 Gb/s NIC 的常见服务器成本约为 4 400 美元, 满载时功耗为 600 W; 而一个 3.3 Tb/s 的 Barefoot Tofino 交换机的成本约为 10 500 美元, 功耗约为 450 W. 因此, 可编程交换机在每秒单 Gb 流量的处理成本上有数十到数百倍的优势, 这也展现了可编程交换机在安全防御成本方面的潜力.

(3) 可重配置, 灵活性高. 由于新的网络协议不断涌现, 传统交换机只能通过更换设备的方式来支持新协议, 而可编程交换机的数据包处理方式则可以被重新配置, 开发人员可以通过使用新程序重新配置交换机从而支持新的协议. 这使得开发人员可以动态更改在目标交换机上运行的安全功能, 或者编写新程序重新配置来添加新的网络安全策略, 使得网络安全防御具有较高的灵活性.

(4) 高性能. 与传统基于 SDN 的网络安全防御不同, 可编程交换机具有极高的处理性能, 可以保证每个安全功能都以高达数十 Tb/s 的线速处理数据包, 这种高性能极大改进了基于 SDN 的安全功能, 也超过了许多提供几百 Gbps 吞吐量的中间设备. 此外, 可编程交换机基于多级流水线的数据包处理架构可以有效降低延迟, 将数据包处理延迟限制在几微秒内.

2 基于可编程交换机的网络安全研究

Cordeiro 等人[20]发表了基于可编程交换机构建监控和安全技术的综述, 并讨论了可编程网络在未来网络管理运营中的研究机会和挑战, 但是该综述没有涵盖大多数利用可编程交换机进行网络安全防御的最新工作. 林耘森箫等人[21]、Michel 等人[22]、Hauser 等人[23]和 Kaur 等人[24]都调研了与可编程网络相关的架构、算法与应用等, 其中包括可编程换机的多种应用, 但是并未将重心完全放在网络安全防御方面, 只是将网络安全防御作为其中的一部分进行简单阐述. AlSabeh 等人[25]对由可编程交换机构建的网络安全技术进行了分类, 采用 STRID 分析[26]检查了一般 P4 应用 (如拥塞控制、负载均衡等) 相关的漏洞和解决方案. Chen 等人[27]从网络攻击的角度对基于可编程交换机的防御方法进行了分类, 讨论了相关技术的设计、实现和局限性等, 但是并未充分考虑目前热门的基于可编程交换机的网络智能平面技术, 没有涵盖这类技术的最新研究.

与上述已有的综述相比, 本文的新颖之处如下.

(1) 包含了基于可编程平面进行网络安全防御的最新研究成果及论文, 覆盖了当前该领域的最新进展. 本文重点针对基于可编程交换机的智能网络平面等具有前景的新兴技术进行了详细阐述, 同时进一步分析了基于可编程交换机的防御技术的未来发展及研究趋势, 展示了更多新颖的技术及思路.

(2) 与 Chen 等人[27]从网络攻击的角度分类不同, 本文侧重于从网络安全防御本身出发, 分析阐述了如何使用可编程交换机增强网络安全防御的基本三元组: 预防、检测和响应. 下面对这 3 个阶段进行简单的说明.

(a) 预防: 阻止攻击者和受保护目标接触. 通常通过设置一些安全策略比如访问控制列表 (access control list, ACL) 来实现, 这些策略指定了特定主体对具体目标的访问权限. 然而这一过程需要进行周密的规划与调查, 以降低误差发生的可能性. 否则, 安全策略可能错误地阻止合法用户访问或允许恶意用户进入.

(b) 检测: 是发现网络入侵的基本安全过程. 通常分为误用检测和异常检测. 误用检测基于已知模式或者特征签名来检测攻击, 收集非正常的网络行为特征建立相关的特征或者签名库, 当检测的网络流量与库中的记录匹配时, 就认为该网络流量是网络入侵流量. 异常检测则是通过查找网络流量中不符合正常预期行为的恶意模式来发现攻击和入侵.

(c) 响应: 对网络攻击的响应也是网络安全防御框架的一个重要组成部分. 网络攻击的响应包括直接丢弃检测到的攻击流量, 也可以对被攻击的主机或网络进行隔离, 防止攻击扩散到网络的其他部分. 响应还包括网络检疫, 即对疑似遭受网络攻击感染的主机实施进一步的检查以清除潜在的网络威胁.

实际网络安全防御技术通常通过组合基本三元组来实现具体的防御策略. 例如检测与响应相结合, 当检测到攻击后, 对攻击采取拒绝或丢弃的响应措施. 本文通过调研现有基于可编程交换机的安全防御技术, 发现这些技术主要采用预防及响应、检测及响应和预防、检测及响应 3 种组合. 因此, 本文将现有技术分为这 3 类进行系统阐述, 具体分类如图 2 所示.

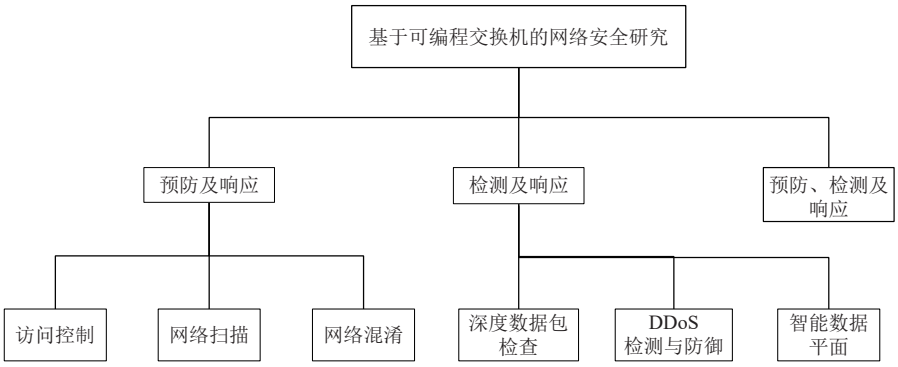


图 2 基于可编程交换机的网络安全研究分类图

2.1 预防及响应

访问控制、网络扫描和网络混淆是网络安全预防的 3 种核心策略, 各自具有不同的功能和定位, 同时在实际防御体系中相辅相成. 访问控制是预防阶段的关键技术, 通过实施精确的访问策略来限制和监控网络资源的访问, 确保只有授权的用户和系统能够访问敏感数据和资源. 网络扫描则是一种主动的防御手段, 它通过定期扫描网络中的设备和服务来识别安全漏洞和配置弱点, 从而允许网络管理员及时采取措施进行加固, 网络扫描的目的是提前发现并修复可能被攻击者利用的安全问题. 网络混淆作为预防阶段另一种主动防御手段, 通过修改网络特征来隐藏真实的网络结构和流量特性, 防止攻击者进行侦察和分析, 从而降低被攻击的可能性. 这三者之间存在着紧密的逻辑关系: 访问控制提供直接的访问限制, 网络扫描帮助完善和优化这些限制策略, 而网络混淆则是进一步提升了整体防御的隐蔽性和复杂性, 使得攻击者更难以获取有价值的网络信息. 通过将访问控制、网络扫描和网络混淆有机结合, 网络管理员可以构建一个更加稳固和全面的网络安全预防体系.

2.1.1 访问控制

利用可编程交换机进行访问控制的基本原理是通过在交换机上实时执行访问控制决策, 而不依赖于集中式控制器, 从而有效避免单点故障和性能瓶颈. 通过使用编程语言如 P4, 交换机能够灵活实现不同的访问控制策略, 以适应不同的网络需求. 通过优化硬件资源 (如 SRAM 和 TCAM) 的使用, 提高访问控制的可扩展性和效率. 此外, 在可编程交换机中使用寄存器或者近似数据结构 (如草图 (sketch) 或者布隆过滤器) 维护和管理状态信息, 支持复杂的有状态访问控制, 从而实现高性能、一体化的网络安全访问控制策略.

Kang 等人[28]针对 BYOD (bring your own device) 的安全问题[29–31]提出了一种新的范式 Poise, 实现了可编程网络内安全. 已有方法则通过 SDN 解决方案来解决 BYOD 的安全问题[32,33], 即收集设备上下文并在中央控制器处执行访问控制, 然而, 中央控制器可能成为瓶颈和攻击目标, 在远程控制器上处理上下文信号也过于缓慢, 无法进行实时决策更改. 利用可编程交换机, Poise 设计了一种新颖的安全原语, 可以在硬件中编程以支持各种上下文感知策略. Poise 的用户指定简明的策略, Poise 将它们编译成 P4 中原语的不同配置. 使用用户定义的协议对上下文信号进行编码, 使用可编程数据包处理计算访问控制决策, 并通过设计硬件数据结构支持有状态的、全网络范围的策略. 实验结果表明, 与传统的 SDN 防御相比, Poise 对控制平面饱和攻击具有抵御能力, 并且极大地提高了

防御的灵活性.

Bajaber 等人[34]则是针对跨主机攻击[35]设计了 P4Control 系统, 精确地限制网络中端到端信息流, 以线速实时防御和阻止跨主机攻击. 现有防御缺乏网络级可见性[36–38], 无法跟踪超出单个主机的攻击活动, 在面对跨主机攻击时存在着较大不足. P4Control 实现了基于可编程交换机和 eBPF (extended berkeley packet filter) 的网络内分布式信息流控制 (in-network decentralized information flow control, DIFC) 机制[39], 并且设计了 NETCL 语言, 用于指定 DIFC 防御策略. 具体而言, 网络管理员首先通过指定 NETCL 策略给主机和匹配的流量分配 DIFC 标签, 然后可编程交换机发送包含指定 DIFC 标签的控制包到相应的基于 eBPF 的主机代理, 主机代理使用接收到的 DIFC 标签初始化主机中现有的进程及文件. 随后当攻击者在主机上进行传播时, 主机代理会从网络流量中提取 DIFC 标签, 并将其接收进程的标签合并, 继续传播这个更新后的标签从而维持 DIFC 上下文的连续性. 当标记的网络流量到达交换机时, 交换机提取 DIFC 标签, 并将这些标签与配置好的 NETCL 策略进行匹配, 从而实现相应的网络访问控制, 允许良性流量通过, 丢弃恶意流量. 广泛的评估证明 P4Control 可以实时有效地阻止跨主机攻击, 在保持网络线速性能的同时开销较小.

与上面两种针对具体领域的访问控制策略不同, Jung 等人[40]从当前交换机上 ACL 的两个缺点, 即可扩展性和规则管理延迟的角度, 提出了一个可扩展和动态的网络内防御 ACL 系统 PortCatcher, 并在可编程交换机上进行了部署. PortCatcher 核心就是将第 4 层端口匹配从 TCAM 中分离出来, 即在 SRAM 中执行 ACL 端口范围匹配, 将 IP/协议匹配分离到 TCAM 中, 以提高其内存效率. 同时为了在 SRAM 中实现快速和可扩展的端口管理, PortCatcher 提出了一种称为线性范围映射 (linear range map, LRM) 的范围表示方法. LRM 方法在 SRAM 中的哈希表中实现端口范围匹配的紧凑表示. 它的核心理念是通过位图表示端口规则, 并将这些位图存储在哈希表中, 以实现快速、可扩展的端口匹配. PortCatcher 系统由 3 个组件组成: IP 模块、端口模块和 ACL 处理器. 源 IP、目的 IP 和协议在 IP 模块中通过 TCAM 匹配, 然后源端口和目的端口在端口模块中通过 SRAM 匹配. ACL 处理器驻留在交换机控制平面, 负责根据数据平面设计管理 ACL 规则. 实验结果表明 PortCatcher 在可编程交换机上显著提升了实时匹配性能, PortCatcher 比 ALPM 节省了 36%–61% 的 TCAM. 规则部署速度则是比最先进的方法快 168 倍, 同时比之前方法多阻止了 55.45% 的恶意流量, 有效提高了网络防御系统阻止恶意数据包的能力.

2.1.2 网络扫描

网络扫描也是网络安全预防的另一种典型技术, 用于发现网络中的活跃主机、端口和服务, 了解网络的安全状况, 可以帮助揭示安全漏洞[41–43]和监控服务部署[44–46]等, 这对于网络的安全预防至关重要. 然而传统的网络扫描器由于实现和部署位置的限制, 在速度和可扩展性方面无法匹配上日益增长的扫描空间. 首先, 传统网络扫描器都是在普通服务器上实现, 而服务器上的 CPU 并非专门用于高速数据包处理, 这导致网络扫描器的扫描速度受限. 其次, 从部署位置来看, 传统网络扫描器都位于网络边缘, 从边缘进行扫描通常受到终端主机上行带宽的限制, 这不可避免地限制了网络扫描任务的最大扫描速度. 此外, 端到端的长扫描路径意味着边缘网络的带宽浪费更大, 探测或响应数据包被丢弃的可能性也更高. 可编程交换机的出现为网络扫描技术带来了新的机遇, 可编程交换机强大的数据包处理能力和可编程性在保证灵活性的同时能保持每秒高达 Tb 的数据包处理性能. 此外交换机也为网络扫描提供了一个独特的有利位置, 不再受限于终端主机的上行带宽, 也不受端到端扫描路径带宽浪费的困扰.

Li 等人[47]就是从上述角度出发, 提出了 IMap, 使用可编程交换机实现快速可扩展的网络内扫描. IMap 包括一个探测数据包生成模块, 用于生成具有随机地址和自适应速率的高速探测数据包, 以及一个响应数据包处理模块, 用于正确高效地处理响应数据包. 具体而言, IMap 使用基于模板的数据包生成机制. 交换机 CPU 首先准备一组带有初始化头部的模板数据包, 并将它们注入到交换芯片中. 模板数据包通过交换芯片中的流水线, 经历加速器、复制器和编辑器这 3 个连续步骤, 最终形成具有所需探测信息的数据包. IMap 利用交换 ASIC 中的编辑器模块生成具有随机地址的探测数据包以覆盖完整的扫描地址空间, 使用 PIPR (probe IP range) 表和随机排列的探测地址范围, IMap 能够生成具有随机地址的数据包, 以覆盖完整的扫描地址空间. 同时, 为了适应不同网络条件, IMap 设计了一个节流阀, 并在交换机管道中动态调整. 节流阀基于控制平面的网络感知方法, 可以根据网络条件精确调整扫描速率, 动态调整探测数据包的生成速率, 以确保不影响网络的正常路由功能. 此外, IMap 通过无状态连接机制和

响应数据包聚合机制, 在保持交换机正常数据包转发功能的同时, 实现了高效的响应数据包处理, 将交换机转变为高速网络扫描器. 无状态连接机制通过将秘密状态编码到探测数据包的可变字段中 (如 TCP 扫描中的源端口和序列号), 在收到响应数据包时, 通过验证这些字段来区分普通数据包和响应数据包, 并定期更新密钥以确保安全性和一致性. 响应数据包聚合机制则是使用一个寄存器数组临时存储扫描结果, 每当数组满时, 最后一个响应数据包会打包所有结果并发送到存储服务器, 从而实现 N 比 1 的聚合, 减少存储服务器的带宽压力. 实验结果表明, IMap 在仅启用单一交换机端口的情况下也能在 8 min 内完成对校园网所有端口 (包括 6 个 B 类 IP 地址) 的扫描, 覆盖高达 250 亿的扫描空间, 实现了比现有最先进的网络扫描器快近 4 倍的扫描速度和高出 1.5 倍的扫描精度.

2.1.3 网络混淆

网络混淆旨在防止攻击者探查网络结构的真实信息, 防范主机探测、拓扑发现、指纹扫描等网络侦察技术, 以保护网络中的关键链路、关键节点和服务. 即使网络流量是端到端加密的, 但是数据包大小和时间信息等元数据仍然暴露了大量正在进行的活动信息, 攻击者可以通过对这些信息进行探测和窃听来进行流量分析攻击. 而网络混淆就是针对这个问题, 利用地址转换、路由突变和填充数据包等操作来修改网络流特性, 控制和修改攻击者收集到的网络信息, 最大限度混淆和降低这些信息的可用性.

Meier 等人[48]从破坏链路探测的角度出发, 在混淆网络拓扑以减轻链路洪泛攻击的同时保证路径追踪工具的可用性. NetHide 将网络混淆问题转换成了一个多目标优化问题, 由两个主要组件构成: 一个保留可用性和可扩展的混淆算法和一个直接在数据平面修改追踪流量的运行时系统. 网络混淆主要依赖于一个整数线性规划 (integer linear programming, ILP) 求解器和有效的启发式算法来计算符合要求的混淆拓扑, 然后控制器将拓扑转换为可编程网络设备的配置下发到数据平面即可编程交换机, 在数据平面直接拦截和修改路径追踪探针来混淆拓扑. NetHide 主要利用了交换机线速解析和修改网络数据包的能力, 主要是对 IP 头中的存活时间 (time-to-live, 简称 TTL) 字段以及 IP 源和目标地址进行修改. 大量真实拓扑实验也证明 NetHide 可以有效混淆网络拓扑, 并且在大于 90% 的概率下准确追踪到链路故障.

在上述基础上, Meier 等人[49]进一步分析了现有的互联网流量混淆技术在广域网 (wide area networks, WAN) 中的局限性. 通过填充混淆单个数据包和流大小的混淆方法[50–52]通常需要对终端主机的软件和协议进行修改, 但是对于运营 WAN 的许多组织和机构而言在所有终端主机上适应这些协议是很困难的. 而限制每个流量的传输时间和速率的方法[53–55]则会严重限制吞吐量, 不足以处理具有高吞吐量特性的 WAN 流量. Meier 等人针对 WAN 中网络混淆的这些局限性, 利用可编程交换机的高性能和灵活性设计了一种适用于 WAN 要求的流量混淆系统 ditto: 在不修改终端主机的情况下以线速实现高吞吐量的流量混淆. ditto 运行在可编程网络设备上, 无需更改端设备, 将传入的 WAN 流量根据预定义的模式将流量整形塑造成具有预定义大小和时间的重复数据包序列来实现流量混淆, 主要通过填充、缓冲/延迟传入的数据包来混淆数据包的大小和时序以及相对顺序. 此外, 当没有足够的真实流量传输时, ditto 通过插入伪造的数据包来填补空隙并确保数据包速率的一致性. 实验结果表明, ditto 在广域网环境中可以在 100 Gb/s 的线路速率下运行, 并且在每条 WAN 链路 70 Gb/s 的真实流量负载下开销极小, 能够实时本地响应流量变化, 快速适应不同的网络负载, 实现线速的高吞吐量流量混淆.

NetHide 和 ditto 都侧重于针对攻击防御的网络混淆, 而 Kon 等人[56]则将目光集中于互联网审查问题, 特别是现有代理服务容易被封锁的挑战, 通过对代理服务的混淆来提升代理服务的抗封锁能力, 促进互联网自由. 现有的审查规避技术主要分为终端用户代理和核心网络规避技术. 前者通过设置规避代理比如 Tor[57]和 Lantern[58]进行实现, 这类技术资源需求低且易于部署, 但是很容易被国家级审查识别和阻止. 后者则是使用诱饵路由 (decoy routing, DR)[59,60]和域名前置技术[61]等, 但是这些技术依赖关键基础设施, 参与门槛较高. Patrick 在这两类技术之外探索了基于边缘网络的新型技术, 提出了 NetShuffle 抗审查系统. NetShuffle 将代理服务与其固定标识符分离, 使用新兴的可编程交换机来实现混洗机制, 在网络中不断更换代理的网络位置 (如 IP 地址), 使得其难以被精确封锁. NetShuffle 在交换机内执行, 提供一个持续变化的代理地址视图, 同时保持内部网络不变. 被审查的客户端通过获取代理标识符 (如 abc.university.edu) 来访问 NetShuffle. DNS 解析器将该标识符解析为一个临时的客户端 IP 地址, 而边界路由器负责将此地址转换为实际的内部 IP 地址. NetShuffle 利用可编程交换机执行地址转换, 确保对内

部服务和客户端透明. 实验结果显示, NetShuffle 在有效规避审查的同时, 开销很小, 保持了低资源消耗.

2.2 检测及响应

本节将从深度数据包检查 (deep packet inspection, DPI)、DDoS 检测与防御以及智能数据平面 (intelligent data plane, IDP) 这 3 个分类方向阐述检测及相应的相关技术, 三者在网络安全防御中虽然相互正交且存在非空交集, 但它们各自的侧重点和应用场景有所不同. 具体而言, DDoS 检测与防御针对分布式拒绝服务攻击, 聚焦于网络流量中的报头特征, 通过分析数据包的报头信息识别异常流量模式和行为特征, 进而采取防御措施来提升网络安全防御能力. DPI 专注于对数据包的明文内容进行深入分析, 通过检查数据包的载荷部分识别出特定的特征或签名, 如攻击模式、恶意代码等, 从而过滤掉恶意明文数据包. DPI 难以对加密的网络流量进行有效分析, 而智能数据平面 IDP 则通过学习加密流量的特征识别恶意网络攻击, IDP 通过在数据平面上直接部署机器学习模型实现智能化的流量分析和决策, 使得流量分析不再依赖于静态规则, 而是基于数据驱动的动态决策. DDoS 检测与防御、DPI 以及 IDP 各自侧重于网络流量的不同层面, DPI 致力于明文内容的深入检查, DDoS 检测与防御关注报头特征分析, 而 IDP 则可以对加密流量进行智能化检测. 尽管三者之间存在一定的重叠, 但它们的核心目标和实现方式存在差异. 因此本文将对 DPI、DDoS 检测与防御以及 IDP 分别进行讨论.

2.2.1 深度数据包检查 (DPI)

深度数据包检查 (DPI) 是一种用于在数据包流经网络时实时检查和分析数据包内容的技术, 可以实现实时分析和决策. 与仅分析包头部分的传统数据包过滤技术不同, DPI 可以分析数据包标头和有效负载, 这也使得 DPI 可以提供更多有关数据包内容的信息从而用于查找、识别、分类和重新路由或阻止包含特定数据或代码有效负载的数据包. DPI 可用于恶意流量或恶意软件签名的检测等, 有效防御多种网络攻击. 传统的 DPI 技术主要基于软件技术[62]或者基于专有中间件[63–65]实现. 基于软件的技术虽然能通过算法优化实现吞吐量的提升, 但是因为服务器上 CPU 能力的限制, 性能还是无法与当今网络带宽和流量的快速增长相匹配. 专有中间件相比软件而言可以实现高吞吐量, 但是专有中间件价格较为昂贵, 成本较高. 总而言之, 无论是算法优化还是专有中间件硬件加速都无法在高性能和成本之间实现较好的平衡. 而可编程交换机的高性能和低成本则为这个问题带来了新的契机, 研究人员可以基于可编程交换机实现高吞吐量且成本优越的深度数据包检测.

Wang 等人[66]针对 DPI 中多字符串匹配的效率问题, 提出了一个利用可编程交换机进行高效多字符串模式匹配的系统 BOLT, BOLT 开发了一种利用三元位编码状态的快速高效的状态编码方案, 以适应可编程交换机中有限的内存容量来容纳大量规则. 其次, BOLT 提出了一种可变 k 步长转换机制, 以在可接受的条目数量增加下显著扩大吞吐量. 在 BOLT 系统中, 管理员首先需要在签名数据库中定义一系列匹配规则. 然后, 控制器从匹配规则中提取字符串模式, 使用 Aho-Corasick (AC) 算法[67]为这些模式构建非确定有限状态自动机 (non-deterministic finite automaton, NFA), 并将 NFA 转换为底层匹配动作表项. 通过这些利用智能状态编码和可变 k 步长转换机制实现的高效匹配动作表, 数据平面在流水线中逐字节地对每个数据包进行匹配. 如果匹配到任何预定义模式, 数据平面将执行相应的动作, 如丢弃、放行或警告. 评估结果表明, BOLT 在吞吐量方面提供了数量级的提升, 并且在模式集大小/类型和工作负载流量方面具有良好的扩展性.

与 BOLT 针对于字符串匹配效率不同, Gupta 等人[68]希望在可编程交换机上实现更为通用的 DPI 解决方案, 其利用可编程交换机支持克隆和再循环数据包的特性, 在数据平面中使用 P4 进行深度数据包检查. 并且使用该方法在一台通用可编程交换机数据平面中构建了应用层防火墙 (URL 过滤器), 在过滤数千个 URL 时实现基本的线速性能. 具体来说, DeeP4R 系统在交换机中实现了一个确定有限状态自动机 (deterministic finite automaton, DFA), 用于在数据包中匹配目标字符串如关键字、URL 等. 随着一次提取一个字符地遍历数据包, DFA 会根据字符进行状态转换 (通过再循环和截断的方法). DFA 的状态能够确定是否已经看到目标字符串, 因此可以匹配在数据包中的任何位置找到的 URL (以及可能的其他字符串或关键字). 但该方法是破坏性的, 因为它消耗了正在匹配的数据包. 为了在防火墙中使用再循环和截断, DeeP4R 系统克隆了数据包. 一份副本可以用于匹配, 另一份根据关键字

(如 URL) 是否找到而决定接受或丢弃. 再循环和截断具体是指从出口部分的组装器返回到入口部分的解析器, 使数据包再次通过流水线, 有效地形成一个循环. 每次流经交换机管道, 数据包被编辑, 移除第 1 个字节, 并检查其值以根据 DFA 进行状态转换. 除了性能以外, DeeP4R 也可以非常简单地移植到其他平台, 具有较好的扩展性. 系统实现和实验结果表明, DeeP4R 可以高效地进行深度数据包处理, 性能显著优于传统防火墙服务, 而且可以较为简单地移植到其他平台.

2.2.2 DDoS 检测与防御

分布式拒绝服务攻击 (DDoS)[69]是网络安全领域长期存在的重大威胁, 攻击者利用一台或多台不同位置的计算机对一个或多个目标同时发动攻击, 消耗目标服务器计算资源或者网络带宽, 使服务器运行缓慢或者宕机, 从而造成服务器无法正常地提供服务. 随着越来越多易受攻击的物联网 (Internet of Things, IoT) 设备接入互联网, 这一威胁也变得更加严重. 此外, DDoS 攻击的方式也愈加多样化, 如 IP 伪造攻击[70]、洪泛攻击[71]、脉冲波攻击[72]等, 这给 DDoS 检测和防御带来了很大挑战. 与上述安全防御方法类似, 传统防御方法大多基于专有中间件或者基于软件进行防御, 这些方法在防御成本和性能延迟方面很难取得平衡, 所以许多研究者也基于可编程交换机来构建更为高效的 DDoS 检测防御系统, 相关方法的对比如表 1 所示. 其中软硬件协同是指某项技术是否依赖控制平面和数据平面协同进行防御; 动态调整是指某项技术能否动态改变防御策略应对动态攻击; 通用性是指某项技术支持 DDoS 防御策略的通用程度; 配置复杂性是指某项技术在实际部署过程中配置参数等方面的复杂程度; 部署方式是指某项技术在防御部署时是只需要单点即可实现防御还是需要多点或全局进行协同防御.

表 1 DDoS 检测与防御方法比较

方法分类	方法名称	核心技术	软硬件协同	是否支持动态调整	通用性	配置复杂性	部署方式
通用防御	Poseidon Jaqen	通用DDoS攻击检测与缓解	是	是	高	高	单点
IP伪造攻击防御	NetHCF	跳数过滤	是	是	低	低	单点
LFA防御	Ripple	去中心化全景防御	否	否	低	高	全局
	Mew	分布式协同防御	是	是	低	高	多点
脉冲波攻击防御	ACC-Turbo	在线聚类和可编程调度	是	是	低	低	单点
洪泛攻击防御	ZAPDOS	前缀特征检测	是	是	高	高	单点
	SmartCookie	Cookie检查	是	否	高	低	单点
网络内监控防御	Cerberus	网络内安全监控	是	是	高	高	单点

Zhang 等人[19]针对目前 DDoS 攻击中基于专有中间件防御[73]的高成本问题和基于 SDN/NFV 防御[74]的性能问题设计了 Poseidon, 旨在开发一个通用的、低成本且高性能的框架, 以防御各种 DDoS 攻击. Poseidon 基于可编程交换机, 提供了三重设计用以防御 DDoS 攻击. 首先, 它提供了一套用于编写 DDoS 防御策略的高级原语, 这些原语掩盖了交换机架构的低级细节, 使管理员能够以高级方式定义所需的策略. 其次, 它提供了一个交换机资源编排器, 可以将用户指定的 DDoS 防御策略最佳地映射到有限的交换机资源上. 第 3, 它设计了运行时管理器, 通过在防御策略之间调度交换机资源来处理动态攻击. 通过这三重设计, Poseidon 能够在保持低开销的同时缓解各种 DDoS 攻击.

与 Poseidon 类似, Liu 等人[75]提出了一个针对多种 DDoS 攻击的交换机原生检测和缓解框架 Jaqen. 具体来说, Jaqen 考虑了 Poseidon 的 3 个局限性. 首先, Poseidon 假设通过基于 NetFlow[76]的采样来检测 DDoS 攻击, 这种采样不可避免地影响了攻击检测的准确性. 其次, 它需要额外的资源将收集到的流量数据转换为攻击检测过程中的流量统计信息. 第 3, 在互联网服务提供商 (internet service provider, ISP) 网络的场景中, 其策略效果较差且性能不佳. 为了缓解这 3 个局限性, Jaqen 将攻击检测和缓解集成到一个完全基于可编程交换机的框架中. Jaqen 在交换机中采用基于签名的检测器和通用草图[77,78]来检测 DDoS 流量. 检测器测量包括每流包计数和熵在内的多种指标来检测 DDoS 攻击. 草图采用近似测量的方法来检测多个包特征, 如 IP 地址和 TCP 端口. Jaqen 命令交换机定期向控制平面报告检测器和草图的计数器值, 控制平面执行 DDoS 攻击检测逻辑. Jaqen 将所有可能的缓解逻辑抽象

为 3 个相互交互的组件. 在每个组件中, 它提供了一组硬件兼容的缓解功能, 这些功能可以被调用以构建 ISP 交换机的各种缓解策略, 组合多种检测和缓解策略来防御基于洪泛的 DDoS 攻击.

除了 Poseidon 和 Jaqen 这类防御多种 DDoS 攻击的通用框架之外, 许多研究人员也针对特定类型的 DDoS 攻击进行了基于可编程交换机的检测防御研究.

Li 等人[79]针对 IP 伪造攻击设计了 NetHCF, 利用可编程交换机进行伪造 IP 流量的过滤. 在 IP 伪造攻击中, 攻击者会修改特定的 IP 头字段以生成攻击流量. 研究人员发现攻击者难以确定 IP 数据包的跳数, 并基于此提出了跳数过滤 (hop count filtering, HCF) 技术[80,81], 通过 IP 到跳数 (IP to hop count, IP2HC) 表过滤伪造 IP 数据包. IP2HC 表仅维护合法流与其对应的跳数之间的映射. 如果一个伪造的攻击数据包带着正常的 IP 地址到达这个表, 表会通过初始 TTL 值减去包中记录的 TTL 值来获得跳数, 并将其与表中的条目匹配, 如果该跳数与 IP2HC 表中的记录不一致, 表就会将这个数据包检测为伪造包并将其丢弃. 然而, 以前的技术都选择将 IP2HC 表部署在终端主机上[80,81], 伪造的数据包要到达终端主机才会被过滤掉, 这无疑会带来巨大的网络带宽消耗, 没有起到防御 DDoS 攻击的目的. 对此, NetHCF 将 IP2HC 表卸载到可编程交换机上, NetHCF 能够直接在数据平面过滤伪造数据包来防止 IP 欺骗攻击. 与传统技术相比, NetHCF 将 IP2HC 表维护在可编程交换机上, 以在线速下过滤伪造的 IP 流量, 最大限度地减少 IP 伪造攻击的影响. 具体来说, 由于交换机资源的限制, NetHCF 在交换机中采用了两个组件. 首先, NetHCF 在数据平面的交换芯片上缓存 IP2HC 表中的热点规则以过滤大多数流量. 其次, 在交换机操作系统上运行的控制平面缓存用于存储大多数 IP2HC 规则并处理未命中 NetHCF 缓存的数据包. 两个组件共同帮助 NetHCF 以较小的开销节省了大量网络带宽. 评估表明 NETHCF 能够以低开销实现线速和自适应流量过滤.

基于可编程交换机进行链路洪泛攻击 (link-flooding attacks, LFA) 的检测防御也是研究热点之一. Xing 等人[82]针对现有基于 SDN 的 LFA 防御方法[83–85]在防御 DDoS 攻击方面具有一定的滞后效应和有限的防御覆盖范围等问题提出了一种可编程、去中心化, 并且能适应攻击动态性的链路洪泛防御方法 Ripple. 具体而言, Ripple 主要由策略语言、编译器和用于链路阻塞防御的分布式运行时构成. Ripple 开发了直接在可编程交换机硬件中运行的防御原语, 通过策略语言进行编程从而精确捕获防御全景即攻击波在网络中的全局、实时视图及其传播方式, Ripple 的用户根据这个全景视图编写防御程序. 然后编译器会自动在每个交换机上生成防御程序. 为了匹配自适应攻击者的动态性, Ripple 以完全去中心化的方式构建了这个全景视图, 使用分布式协议同步交换机本地视图. 这使得 Ripple 防御可以在攻击传播时对快速变化的攻击进行全景防御. 广泛的评估也显示 Ripple 可以被用于一系列防御, 并且在缓解动态攻击方面显著优于 SDN 防御.

Ripple 展现了基于可编程交换机进行 LFA 防御的优越性, 但是其在防御时仍然存在着资源受限以及不支持运行时重配置的问题. Zhou 等人[86]将目光聚集在这两个问题, 提出了基于可编程交换机的内存高效且适应性强的链路泛洪攻击防御系统 Mew. Mew 使用轻量级的分布式协议将大量的流状态调度到不同的交换机上以减少内存瓶颈, 同时设计了一组应用接口来支持多粒度和动态的协同防御, 应对复杂的 LFA. 首先在分布式存储方面, Mew 设计了一种分布式存储协议减少数据平面的存储开销, 同时使用一种状态迁移机制来缓解流量集中问题. 当激活防御模式并出现新流时, 边缘交换机记录其现有状态并启动协商协议. 在不中断流的情况下, 根据预定义的策略 (例如首选最少利用率), 选举出路由路径上的一个交换机来捕获此流. 为了灵活性, 捕获的流可以在交换机之间迁移. 其次在协同防御方面, Mew 设计了一组协同防御应用接口, 支持组内的灵活共检测和共缓解, 减少了通信和开销内存. 通过使用协作应用接口, 网络运营商在多个可编程交换机上部署协同检测和协同减轻模块. 例如, 在拥塞的链路上的交换机可以请求其他交换机的相关流状态或通知其他交换机其链路状态. 最后在运行时管理方面, 由于资源限制, Mew 先为部署的每个功能 (包括非防御功能) 分配最小资源, 以支持尽可能多种的防御. 根据实时网络条件, 网络运营商为激活的模块分配内存, 并在减轻后回收内存. 通过内存访问代理, 每个功能访问代理而不是寄存器, 代理将访问请求转换为真正的寄存器访问操作从而实现不同的功能之间共享寄存器从而使可编程交换机可以加载更多功能. 真实实现和实验表明, Mew 能有效地防御大规模和动态的 LFA.

Alcoz 等人[87]针对基于聚合的拥塞控制 (aggregate-based congestion control, ACC)[88]面对脉冲波 DDoS 攻击的局限性, 设计了 ACC-Turbo 进行脉冲波 DDoS 攻击的防御. ACC 在防御传统 DDoS 攻击上比较有效, 但是却无法

有效缓解脉冲波攻击. 主要原因是 ACC 依赖于运行在单独的服务器或控制平面上离线推断和控制机制以及基于阈值的防御激活, 这两部分减慢了 ACC 的反应时间, 增加了误报的概率, 对性能产生了较大的负面影响. 因此, ACC-Turbo 在可编程交换机上运行 ACC 机制, 通过线速运行来缓解脉冲波 DDoS 攻击. ACC-Turbo 在数据平面直接进行在线聚类以推断攻击, 并使用可编程调度来缓解攻击. 首先, ACC-Turbo 将聚类过程卸载到数据平面, 并以线速分析所有流量, 这使 ACC-Turbo 能够大幅加快反应时间. 此外, 由于数据平面处理不影响流量延迟, 无论是否存在拥塞, ACC-Turbo 都可以持续运行聚类算法, 消除了基于阈值的激活的需要, 使 ACC-Turbo 能够预测拥塞事件, 实现更快的反应时间. 其次, ACC-Turbo 使用可编程调度来降低恶意流量的优先级, 以每个数据包的粒度适应流量变化, 并迅速对攻击变化做出反应, 而不是简单丢弃或限制速率. 实现和实验评估表明, ACC-Turbo 能够以无监督的方式自主识别 DDoS 攻击向量并迅速缓解脉冲波 DDoS 攻击.

Misa 等人[89]研究了 DDoS 防御的签名检测, 指出目前利用可编程交换机进行签名检测要么只实现了简单的启发式或基于阈值的检测[75,89,90], 准确性有限; 要么利用机器学习 (machine learning, ML) 技术可以实现高准确性[91–94], 但需要计算每个流的特征向量, 导致不可行的资源开销. 因此, 他主张攻击签名应该在前缀级别构建, 以反映和利用地址的固有聚类. 特别是, 前缀级签名能够在需要一小部分监控资源的同时, 实现与源级签名相同的准确性. 在此基础上他提出了新型前缀级容量耗尽型 DDoS 攻击签名检测方法 ZAPDOS. ZAPDOS 利用在 P4 交换机上部署的前缀监控槽收集固定数量的前缀特征, 根据结果更新要监控的前缀, 并在足够有信心将攻击前缀与良性前缀分开时立即报告攻击前缀. 控制平面则负责评估每个前缀的分类模型和细化算法. 同时, ZAPDOS 通过数据包摆渡和批量循环轮询优化 ASIC 寄存器和交换机 CPU 之间的通信. 此外, ZAPDOS 采用了一种预先检测的方法, 在分配下一周期的监控资源之前识别出哪些子前缀是活跃的, 并使用一种回溯记录的方法, 以低成本维护一份近似记录, 追踪攻击源变化时哪些未监控的前缀是活跃的, 从而优化下一周期的资源分配. 原型实现和仿真实验表明, ZAPDOS 可以在多种攻击场景下以低于现有技术的误差率有效检 DDoS 攻击.

除了上述方面之外, SYN 洪泛攻击作为 DDoS 攻击中占比最大的攻击类型[95], 至今仍是一个很大的威胁. 在大规模攻击下, 基于软件的 SYN Cookie 防御[96]在进行数据包处理和 Cookie 计算时会产生高开销, 导致 CPU 耗尽. 而纯粹的基于交换机的解决方案必须在严格的硬件限制下运行, 包括有限的计算能力和有限的内存. 可编程交换机不原生支持密码学原语, 因为其计算成本高昂. 因此, 现有的基于可编程交换机的防御方法[19,75]使用不安全的 CRC32 来计算 SYN Cookie, 放弃了安全性, 破坏了 SYN Cookie 检查的目的. 此外, 由于交换机只有几十 MB 的内存, 内存密集型的 SYN 洪泛防御在跟踪每个连接的状态时难以扩展, 导致应用程序性能降低. Yoo 等人[97]基于分工合作的思想提出了软硬协作的 SmartCookie. SmartCookie 智能划分防御工作量, 对交换机和服务器协同防御的优越性做最大化处理, 同时最小化每种方法的局限性. 具体而言, SmartCookie 交换机代理安全地执行 Cookie 检查以快速过滤恶意流量, 并大致跟踪已验证的连接. 交换机代理在数据平面上高效计算和验证 SYN Cookies, 使用安全性更高的 HalfSipHash-2-4 算法[98]结合布隆过滤器, 优化哈希计算和内存使用, 过滤恶意流量的同时确保良性流量不被阻止. SmartCookie 服务器代理处理序列号则负责转换并精确跟踪已验证的连接. 服务器代理使用 eBPF 重新设计了网络和 Linux 内核网络栈之间的 TCP 接口, 将连接设置和序列号转换卸载到服务器, 无需修改内核 TCP 栈, 实现即时部署. eBPF 程序在传入和传出数据包上执行连接设置和序列号转换, 使用 eBPF 映射跟踪连接状态和协调行为. 实验结果表明 SmartCookie 比传统基于 CPU 的软件防御能处理的攻击流量多两个数量级, 并且与现有的基于交换机的硬件防御相比, 对良性流量的端到端延迟降低了约 66.7% 到 86.7%.

基于可编程交换机的网络内监控 (in-network monitoring, INM) 系统也是 DDoS 检测防御的有效方法. 作为一种颇具前景的高性能和实时网络监控方法, INM 系统可以处理多样化和高流量的 INM 任务, 比如负载均衡、DDoS 防御等. 但是在大规模网络监控场景中现有的基于 P4 的 INM 系统[99,100]只能同时为非常少量的并发 INM 任务提供服务, 或者只能处理特定量的流量, 这使得防御多样化和大容量攻击成为一项挑战. 当网络管理员检测到新的攻击时, 他们需要重新加载新的程序到交换机, 这也会导致几十秒的停机时间, 攻击者可以通过迅速更改攻击向量导致对 INM 的拒绝服务攻击. 此外, 攻击者也会针对某个 INM 任务设计大象流攻击导致其他 INM 任务内存

不足从而不可用. Zhou 等人[101]针对这些问题提出了 Cerberus 系统, 旨在减轻混合和动态攻击的影响, 保障网络的可用性和性能. Cerberbus 系统由内存分片、共同监控和资源管理器这 3 部分组成. Cerberus 将每个 INM 任务抽象为由多个关键特征对 (key-feature, K-F) 表示, 一个 K-F 对意味着记录与给定的键值对应的特定特征, 占用部分资源, 程序员只需要组合各种 K-F 来选择对应的 INM 任务, 同时编译器可以智能评估并组合不同的 K-F 对. 流量通常有数十个键和数百个特征, 因资源限制难以同时监控. 为解决此问题, 内存分片技术将寄存器共享给多个 K-F 对, 使不同函数可同时访问其 K-F 对. 共同监控的关键思想就是数据平面存储最频繁更新的最低有效位, 而控制平面只存储最不频繁更新的最高有效位, 避免了给控制平面引入沉重的开销, 以此实现对于大象流的处理. 资源管理器部分主要是为了解决网络条件变化时重新配置程序导致的交换机停机问题, 通过流水线优化, 适应性内存空间和动态驱逐来动态重新分配 INM 任务的资源, 并调整数据平面和控制平面的负载, 而不会中断正在运行的服务. 广泛的实验评估证明 Cerberus 可以将可编程交换机的并发性和容量增加一个数量级并且在处理各种 INM 任务方面更具适应性.

2.2.3 智能数据平面 (IDP)

随着近些年人工智能的流行, 机器学习在各种网络设计中也越来越受欢迎, 例如拥塞控制、路由优化以及恶意流量检测等. 机器学习模型通常部署在终端主机或者网络控制平面上进行推理, 因为这些设备配备了灵活的 CPU 或者 GPU. 但是这样部署会导致延迟增加, 同时处理流量的速度也无法令人满意. 因此出现了智能数据平面 (IDP), IDP 是指将机器学习模型直接部署在网络数据平面上, 可以通过数据驱动模型而不是预定义的协议来实现智能化的流量分析, 并且可以实现线速级别的处理. 这有助于提高恶意流量检测等网络防御的准确性, 进一步保证网络安全. 在传统网络数据平面上实现动态和数据驱动的机器学习模型是几乎不可能的, 但是可编程交换机的出现允许用户定制数据包处理逻辑, 具有较高的灵活性, 这也使得在可编程交换机上部署学习模型变得可能. 尽管可编程交换机存在一定的计算和存储资源限制, 但是它的灵活性也激发了大量研究人员围绕可编程交换机进行智能数据平面的设计, 相关方法的对比如表 2 所示. 其中软硬件协同是指某项技术是否依赖控制平面和数据平面协同进行防御; 支持场景丰富程度是指某项技术用于处理不同类型的网络流量分析任务的能力范围; 适应流量变化是指某项技术能否根据攻击流量的变化动态调整防御策略; 资源占用是指某项技术在 TCAM、SRAM 等交换机资源方面的占用情况.

表 2 智能数据平面方法比较

方法分类	方法名称	核心技术	软硬件协同	延迟	检测准确度	支持场景丰富程度	适应流量变化	是否线速	资源占用
数据平面提取特征, 控制平面运行流量分析	NetWarden	快速/慢速路径架构	是	高	一般	低	否	否	低
	FlowLens	数据流标记收集	是	高	一般	一般	否	否	低
数据平面运行决策树/随机森林	Mousika	二叉决策树转换及部署	否	低	一般	低	否	是	低
	Dryad	自适应部署增强的决策树	是	低	一般	低	否	是	低
	SOTERIA	自动化多延迟神经网络生成和调度	是	高	一般	低	是	否	高
	In-Forest	分布式网络内多交换机集成分类	是	高	高	低	是	否	低
	NetBeacon	高效模型表示机制	是	低	高	低	否	是	低
	Leo	决策树模型抽象	是	低	高	低	是	是	低
	HorusEye	无监督算法检测	是	高	高	低	否	否	低
数据平面运行复杂神经网络	BoS	适配可编程交换机的RNN架构	是	高	高	低	是	否	低

目前已有的针对 IDP 的研究技术主要分为 3 类.

(1) 数据平面提取特征, 控制平面运行流量分析

NetWarden[102]和 FlowLens[92]等为代表的使用可编程交换机收集有用的流信息, 然后在控制平面上进行流量分析的设计.

Xing 等人[102]针对网络隐蔽信道的检测和缓解问题设计了 NetWarden, 利用可编程交换机的优势构建了快速

路径 (fastpath) 和慢速路径 (slowpath) 相结合的架构. 快速路径主要是利用可编程交换机实现高效的包处理和初步检测, 慢速路径则是利用快速路径收集到的信息进行一些更为复杂的流量分析和检测操作. 具体而言, 在数据平面上进行连接监控、包间延迟 (inter-packet delay, IPD) 计算和预检查以及包头修改以阻断隐蔽存储信道. 这一过程中会对 IPD 进行简单的范围检查, 以初步检测潜在的隐蔽时间信道, 对于通过预检查标记为可疑的连接, 将其 IPD 数据发送到控制平面进行详细的统计分析. 控制平面上对可疑连接执行全面的统计 IPD 测试, 利用已有的检测器 (如 Kolmogorov-Smirnov 检验[103]或者其他自行训练的检测模型) 进行详细分析. 并且, 缓存可疑连接的数据包, 按配置的时间间隔发送, 扰乱时间调制. 此外 NetWarden 还采用了 ACK 加速和接收窗口加速两种机制来对冲隐蔽信道防御带来的性能损失. 实验评估表明, NetWarden 能够在几乎不影响性能的情况下减轻隐蔽信道攻击.

FlowLens[92]相较 NetWarden 则更为通用, 它是一个利用可编程交换机高效支持多用途基于机器学习的安全应用的系统. FlowLens 由以下组件组成: 在交换机上运行的 P4 程序和两个软件组件 (收集器和分类器)、独立的分析服务器以及提供给系统运营商的软件客户端. 在交换机上运行的组件共同负责分析流量并根据基于机器学习的安全应用进行流分类. P4 程序在数据平面上运行, 并实现了一个定制的数据结构 FMA (flow marker accumulator). FMA 用于收集流的数据包长度或数据包到达时间分布的简明编码, 称为流标记, 然后收集器从数据平面获取生成的流标记, 之后分类器根据加载的模型对其进行处理. 系统运营商可以检索分类器的结果, 然后针对特定流采取有针对性的操作, 例如删除标记的流或安排进一步的记录操作. 实现和实验结果表明, FlowLens 可以增加防御容量两个量级, 而损失的检测精度小于 3%.

这类方法有效结合了机器学习模型和可编程交换机的优势, 但是由于数据平面和控制平面之间的延迟, 很难做到对流量的线速分析, 在性能方面仍然具有一定的局限性.

(2) 数据平面运行决策树/随机森林模型

Mousika[94]和 Leo[104]等为代表的研究进一步实现 IDP, 主要集中在数据平面上部署决策树/随机森林模型. 通过使用匹配动作表表示决策树分支, 将决策树模型嵌入网络数据平面, 并且针对过程中可编程交换机的资源限制、模型的自适应性和可扩展性等进行研究和改进.

Xie 等人[94]针对这类方法进行了深入研究, 提出了 Mousika, 针对机器学习模型如决策树 (decision tree, DT)[105]或者更为复杂的模型在交换机部署时所面临的计算和存储需求问题进行优化. Mousika 通过知识蒸馏在可编程交换机中实现通用的网络内智能. 具体而言, Mousika 主要通过如下核心技术实现通用的网络内智能: 将 DT 修改为二叉决策树 (binary decision tree, BDT) 以及引入了教师-学生知识蒸馏架构[106], 它使得可以将其他学习模型通用地转换为 BDT. 与 DT 相比, BDT 支持更快的训练, 生成更少的规则, 并更好地满足交换机的资源约束. Mousika 将 BDT 的分类规则编码成三态匹配表条目, 最后将这些条目安装到交换机的 P4 程序中, 这个程序只使用了少量的程序资源, 在紧凑型交换机上足够轻量级. 此外, 通过知识蒸馏的转换, 不仅可以利用复杂模型的超级学习能力, 还可以避免在交换机上直接部署时的计算/内存约束, 以实现线速处理. Xie 等人进一步观察到已有的可编程平面中部署 DT 的方法未关注一个重要的要求: 快速适应交换机资源的变化, 如果在中央控制器上维护不同的 DT 模型以适应不同的资源是低效和缓慢的. 针对这个问题, Xie 等人[107]提出了 Dryad, 核心洞察就是可以通过增加训练时每个节点的统计分布替代为不同的资源约束维护不同的 DT 模型, 并将这个增强的 DT 称为一次训练适用于所有的 DT (one-training-for-all DT, ODT). 为了适应不同的资源约束, Dryad 修剪 ODT 以得到一个在给定一组资源约束下能够达到良好性能的 DT. 具体由训练过程和自适应部署过程两个独立的阶段构成. 训练过程在高性能的远程服务器上进行, 通过训练时增加每个节点的统计分布训练一个准确的 ODT, 而不会因资源约束而牺牲模型大小. 训练好的 ODT 中的节点已经总结了观察到的训练数据统计信息, 以供未来适应使用. 自适应部署则是对 ODT 进行剪枝并部署到可编程交换机, 当交换机资源约束发生变化时, 采用渐进式搜索算法在交换机中进行部署, 该算法遍历修剪所有可行组合以找到在分类准确性和处理延迟方面最佳的 DT, 然后, ODT 编译器生成所需的 P4 代码 (匹配动作表以及表条目) 并安装到交换机上. 除了 Mousika 和 Dryad 之外, Xie 等人[108]还针对目前神经网络 (neural networks, NN) 必须在不同设备之间定制以满足异构设置 (例如操作系统和 CPU 类型) 而且一个模型可能需要经常调整以适应同一设备的不同流量速率的问题, 设计了 SOTERIA, 一个用于快速准确检测异构硬件上流量

变化的自动化多延迟 NN 生成和调度系统. SOTERIA 首先在 GPU 服务器上进行神经结构搜索 (neural architecture search, NAS) 训练[109,110], 这部分使用进化训练算法 (evolutionary training algorithm, ETA) 和非支配排序[111,112]提升 NAS 性能和准确性, 训练完成后获得一组准确性和延迟多样性方面表现最优的 NN, 训练好的子 NN 集合可以部署到不同的网络设备上. 然后在设备上进行流量检测, 检测过程中调度器通过启发式 NN 调度算法根据流的特征和设备延迟选择最佳的 NN 进行检测推理, 从而满足当前的流量速率, 并且很好地适配不同异构设备的资源. 实验结果表明, SOTERIA 的检测准确度和 $F1$ 得分均优于其他神经网络方案且 GPU 训练资源和时间需求大幅降低, 在流量变化时检测速度显著提升.

与 Dryad 类似, Lin 等人[113]针对现有通过机器学习模型部署在可编程平面上进行分类存在的缺乏对不同网络资源场景和动态流量变化的支持或者资源不足导致准确性降低的问题, 提出了一个通用的分布式网络内分类框架 In-Forest, 利用多个交换机的可用资源部署大规模模型. 具体而言, 其设计了一个 LEGO 模型, 它可以转换为具有完整功能和不同分类知识的多个增强基础模型. 每个增强基础模型都具有完整功能, 可以单独部署在一个交换机上. 然后, In-Forest 通过一个两阶段资源感知模型分配策略来确定增强基础模型的最佳分配方案, 该策略首先通过离线拓扑感知分配选择交换机的模型, 最大化有限可选模型和交换机资源范围内的所有路径上的模型多样性. 然后, 基于深度强化学习 (deep reinforcement learning, DRL) 的在线流量感知分配通过调整部署的模型以最大化准确性来响应流量变化, 采用稳定的学习机制来确保有效性. 最后通过一个轻量级模型更新机制在可用资源发生变化时进行最佳模型扩展或缩减. 当流量经过具有不同增强基础模型的交换机时, In-Forest 通过集成学习 (例如多数投票) 聚合来自不同增强基础模型的分类结果, 以在没有单一点资源限制的情况下获得更高的准确性. 实验结果表明, In-Forest 可以将准确性提高 19.31%, 同时将交换机规则数量减少 89.98%.

Zhou 等人[114]则是在可部署的现有技术上引入流级特征提高流量分析精度, 并且提出了一种高效的模型表示机制, 解决 Mousika 在数据平面匹配表中表示决策树或随机森林模型时的表项爆炸问题. 在设计的 NetBeacon 系统中首先设计了一个数据平面感知模型, 用于生成适合交换机的学习模型. 同时考虑到流量在不同阶段携带不同的流级特征 (如数据包大小均值), 模型采用多阶段序列模型架构, 在流量不同阶段进行动态分析, 避免过早分类决策导致的错误. 然后, NetBeacon 使用高效的模型表示机制将学习到的模型转化为数据平面上的多个特征表和模型表. 通过高效的编码机制减少表示模型时的表项消耗. 其中特征表将特征值编码为称为范围标记的数据结构, 这些范围标记进一步映射到存储在模型表中的推断结果. 此外, NetBeacon 通过区分短流和长流的处理逻辑, 以及在观察到存储索引冲突时允许安全的存储复用增强了处理并发流的可扩展性. 实验结果表明, NetBeacon 比 Mousika 大幅减少了表项消耗, 在流量分析准确性和硬件消耗方面都表现得更好.

Jafri 等人[104]分析了现有在交换机上部署决策树模型的方法, 发现它们大多数遵循决策树的自然依赖关系, 处理较大的决策树时受限于交换机阶段的数量, 不能表示所有类, 而且不支持运行时可编程, 无法在不中断交换机运行时进行更新. 因此其对之前的在线决策树部署进行改进, 设计了 Leo, 实现了基于在线机器学习的线速流量分类, 具有速度快、可扩展性和运行时可编程的特性. Leo 对决策树模型进行抽象, 由三元组 (D, L, F) 指定决策树“类”中的任何树, 其中 D 是最大树深度, L 是最大叶子节点数, F 是树节点可以使用的特征集. Leo 首先在编译时为单个决策树结构进行配置, 该结构可以在运行时进行复用, 以实现类中的任何树. 这棵树被称为代表性决策树. 然后, Leo 通过决策树节点复用和子树展平复用将代表性决策树映射到交换机. 最后运行时, 交换机控制平面为代表性决策树结构中的每个节点配置特征和约束值, 以在类别 (D, L, F) 中实现特定的决策树. 同时 Leo 通过维护代表性决策树的两副本解决更新时决策树状态不一致的问题, 从而实现运行时可重配置. 实现和评估结果表明, Leo 能够以线速对流量进行分类且能够扩展到比现有数据平面 ML 分类系统大一倍以上的模型尺寸, 同时实现了与离线流量分类器相当的分类准确性.

与上述专注于如何更好地将决策树这类模型部署到交换机上不同, Dong 等人[115]从有监督和无监督的区别角度出发, 认为现有部署决策树模型是一种需要大规模异常数据集的有监督方法, 但是很难获取高质量的网络入侵异常数据集, 因此他基于孤立森林 (isolation forest, iForest) 无监督算法[116]设计了一个名为 HorusEye 的两阶段物联网异常检测框架. 两阶段分别部署在数据平面 (即可编程交换机) 和控制平面 (例如 x86 服务器) 上. 第 1 阶段在

数据平面上设计了一个无监督模型——Gulliver Tunnel, 该模型通过规则生成算法将训练好的 iForest 模型转换为白名单规则安装在数据平面上, 几乎产生与原始模型相同的检测结果, 然后使用特征提取器从流入流量中提取特征进行检测, 以线速过滤出少量可疑流量. 第 2 阶段就是通过控制平面上的无监督深度学习模型 Magnifier 对从数据平面接收到的可疑流 ID 进行进一步检测, Magnifier 首先将流 ID 与其可疑流表中的流 ID 进行匹配, 如果流的可疑异常频率超过阈值, Magnifier 将使用轻量级深度学习模型从历史镜像流量中进一步分析流模式. 最后, Magnifier 将确认的恶意流 ID 添加到数据平面上的黑名单中, 以阻止或限制将来此流的速度. 实验结果表明, HorusEye 的数据平面可以检测到 99% 的异常, 并将 76% 的流量从控制平面卸载.

总体而言, 此类方法通过在可编程交换机上部署决策树/森林模型实现了高速的智能流量分析, 一定程度上实现性能和准确性的权衡. 但是由于可编程交换机上的内存资源和计算资源的限制, 可编程交换机无法为较复杂的决策树提供完整的匹配动作表支持, 并且在高速条件下可计算的流特征 (比如数据包长度的标准差、频率和百分数等) 比较有限. 这种局限性使得模型进行流量分析的准确性降低, 仍需进一步的改进.

(3) 数据平面运行复杂神经网络模型

这类技术聚焦于更为复杂的神经网络 (如循环神经网络 RNN[117]和 Transformer[118]), 将这些先进的模型融入可编程交换机, 进一步提高分析性能.

Yan 等人[119]就是聚焦于更为复杂的循环神经网络 (recurrent neural network, RNN), 提出树模型在准确性和效率的不足, 设计了 BoS (brain-on-switch), 在数据平面实现复杂的 RNN 计算, 并且通过设计的流量升级机制来容纳基于 Transformer 的全精度流量分析模块. 为了实现适配数据平面的 RNN 推理架构, BoS 通过二值 RNN 架构在数据平面上保留全精度模型权重, 通过匹配动作表实现复杂的层前向传播函数编码. 此外, 其使用基于滑动窗口的计算方案, 用有限的交换机转发阶段执行无限的 RNN 时间步长. 而为了解决比较复杂的分类情况, 比如多类流量分类中, 可能没有任何类别占主导地位, 这个时候就需要通过更为复杂的交换机外 Transformer 类型流量分析模块进行处理. 这部分首先设计了一种聚合算法, 核心是通过累积每个类别的预测概率来聚合多个中间推理结果, 并以此决定每个数据包的分类以及是否将整个流量升级至更高级的分析模块来避免大量流量升级导致的速率下降. 然后, 通过集成模型推理系统 (integrated model inference system, IMIS) 进行更为复杂的流量分析, 它能够快速在线推理全精度的基于 Transformer 的模型. IMIS 协调 4 种有状态的和单线程的任务来实现非阻塞的流量处理流水线, 从而使复杂的在线分析能维持较高的网络转发速度. 原型实现和实验表明, BoS 在多个流量分析任务上的准确性和可扩展性方面都优于最新技术.

这类方法通过将更为复杂的神经网络部署在可编程交换机上, 进一步提高了模型分类的准确性, 但是复杂模型的推理一定程度上也导致了延迟的增加, 影响实时流量分析的效果. 随着网络规模和流量的增加, 维持高效的流量分析也变得更加困难, 可能需要更多资源以及更为复杂的管理机制.

2.3 预防、检测及响应

远程直接内存访问 (remote direct memory access, RDMA) 在云数据中心越来越受欢迎[120]. 在 RDMA 中, 客户端绕过服务器 CPU 直接读写远程内存. 最近的研究显示 RDMA 存在着大量漏洞[121–123], 可能导致数据包注入、拒绝服务以及信道泄露等攻击. Xing 等人[124]针对 RDMA 的这些问题设计了 Bedrock 系统, 利用可编程数据平面构建了 RDMA 源认证、访问控制以及监控和记录的防御原语, 可以有效缓解许多攻击. 具体组成如下.

(1) Bedrock 通过可编程交换机和 eBPF 技术实现了 RDMA 源认证. 对于来自不同机器的数据包, 可编程交换机利用网络拓扑不变量进行认证, 通过匹配动作表检查每个数据包的 IP 地址与预设的拓扑信息 (如交换机入口端口 ID) 是否一致从而确保数据包源自正确的网络位置. 而对于来自同一机器的数据包, Bedrock 则是利用 eBPF 技术在内核层面提供更细粒度的控制. Bedrock 利用 eBPF 技术拦截 RDMA 操作, 确保通过正确地创建硬件队列、管理客户端 QPN (queue pair numbers) 列表以及替换服务器 QPN 为客户端 QPN, 来绑定 RDMA 连接到合法的进程 ID 并阻止未认证的通信. 这些机制结合起来为 RDMA 提供了一套全面的源认证机制.

(2) Bedrock 通过将 ACL 卸载到可编程交换机, 允许用户以编程方式指定和修改 ACL 策略, 解决了传统 RDMA ACL 硬编码和集成性问题. 它在服务器上使用一个 RDMA 外部库来管理 ACL 组和 QPN, 并通过交换机

控制平面的守护程序将策略部署到可编程交换机中. 在数据平面, P4 程序根据 ACL 表中的规则来执行访问控制决策. 为了解决交换机资源限制问题, Bedrock 采用可调整的 ACL 粒度、表分解技术, 以及 TCAM 和 SRAM 之间的权衡, 来压缩 ACL 并优化资源使用. 这些措施共同提升了访问控制的灵活性和效率, 同时保持了高性能的 RDMA 操作.

(3) Bedrock 通过在网络中监控和记录 RDMA 流量, 使 RDMA 系统恢复可见性, 增强安全保障. Bedrock 在交换机中执行 RDMA 监控, 并跟踪和记录 RDMA 请求以构建审计日志. 它利用计数-最小草图和布隆过滤器等近似数据结构来节省空间并实现有效的监控. 日志包含足够的元数据 (例如内存地址、操作码), 可以用于 NetFlow 样式的记录和审计. 通过分析这些记录, Bedrock 能够检测和减轻 DDoS 攻击、侧信道攻击和数据泄露攻击. 此外, Bedrock 还提供定制化的监控和记录能力, 以应对特定安全场景, 例如通过跟踪 rkey 探测来防止侦察攻击.

真实场景的评估表明, Bedrock 能够在几乎没有开销的情况下缓解一系列 RDMA 攻击.

3 总结和展望

可编程交换机因其在灵活性和性能上的显著优势受到越来越多研究人员的关注, 基于可编程交换机的网络安全防御技术也成为该领域的研究热点之一. 本文调研了近年来基于可编程交换机进行网络安全防御的最新研究成果及论文, 覆盖了当前该领域的最新进展, 并基于网络安全防御的基本三元组——预防、检测和响应, 对这些技术进行了系统地归纳和总结, 详细分析了各类技术的设计理念、实现机制、系统优势以及局限性. 尽管当前基于可编程交换机的防御技术已经取得了显著成功, 但是随着可编程网络的不断演进和各类新兴场景的涌现, 基于可编程交换机的网络安全防御仍然还有很多研究方向值得进一步探索.

(1) 基于可编程交换机的复杂神经网络防御研究. 目前随着深度学习的发展, 越来越多深度学习算法被用于流量分析与检测, 并且实现了极高的准确性. 但是这些模型大多部署在终端主机或者控制平面上进行推理, 无法实现高速流量分析, 因此近年来许多学者研究了基于可编程交换机的智能数据平面技术. 但是目前大多数智能平面技术主要还是部署决策树/森林模型这类简单神经网络, 准确性有待进一步提高. 最新的部署 RNN 网络模型的技术则是展现了复杂神经网络在准确性和效率上的优越性. 未来研究者们可以进一步研究使用可编程交换机实现更为复杂的神经网络模型的推理, 进一步提高智能平面技术以线速进行网络分析检测的准确性和效率.

(2) 可编程网络系统本身安全性研究. 现有基于可编程交换机进行安全防御的工作主要利用可编程交换机的灵活性、低成本、高性能等特性, 相比传统的防御方法在性能、成本等方面取得了较大优势. 但是这些工作在可编程网络系统本身的安全性上考虑不多, 如果攻击者针对可编程网络系统进行攻击, 比如通过恶意消耗可编程交换机的内存以及计算资源使得部署在数据平面上的防御策略资源耗尽, 无法正常运行, 这就导致了可用性漏洞. 因此, 未来针对可编程网络系统本身的安全性进行深入研究也是一个具有前景的方向, 研究人员可以针对可编程数据平面的攻击进行定义和分类, 并设计相应的预防和检测措施以减轻潜在的风险.

(3) 可编程网络协同防御技术研究. 近年来随着可编程网络的进一步演进, 除了可编程交换机之外, 可编程智能网卡 DPU (data processing unit) 以及可编程内核技术 eBPF 也逐渐兴起. 因为可编程交换机的资源及架构限制, 很多防御任务可能无法完全由交换机进行实现, 因此利用 DPU 及 eBPF 等可编程网络元素进行协同防御也是未来有潜力的研究方向之一. 通过对防御任务的分解将其分散部署到不同的可编程网络元素上, 充分利用各部分的优点, 克服相应的缺点, 实现高性能、灵活的防御架构和策略, 从而提升网络安全防御的能力.

(4) 基于可编程交换机的加解密协议卸载研究. 随着网络的快速发展, 加密通信已经成为保障数据安全的重要手段. 传统加解密操作通常在终端设备完成, 导致计算资源紧张和延迟增加. 可编程交换机凭借其高性能、灵活性以及位置的独特性为加解密协议卸载提供了新可能. 已有研究表明[125], 可编程交换机可以高效支持对称加密算法如 AES. 然而如何实现更复杂的加解密算法以及保证安全性和稳定性仍需进一步研究. 未来研究人员可以进一步探索基于可编程交换机的加解密协议卸载以支持多种加解密协议的高速处理从而提升网络安全防御的整体性能.

References:

[1] Casado M, Freedman MJ, Pettit J, Luo JY, McKeown N, Shenker S. Ethane: Taking control of the enterprise. ACM SIGCOMM

Computer Communication Review, 2007, 37(4): 1–12. [doi: 10.1145/1282427.1282382]

[2] McKeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, Shenker S, Turner J. OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69–74. [doi: 10.1145/1355734.1355746]

[3] Bosshart P, Daly D, Gibb G, Izzard M, McKeown N, Rexford J, Schlesinger C, Talayco D, Vahdat A, Varghese G, Walker D. P4: Programming protocol-independent packet processors. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 87–95. [doi: 10.1145/2656877.2656890]

[4] Bosshart P, Gibb G, Kim HS, Varghese G, McKeown N, Izzard M, Mujica F, Horowitz M. Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 99–110. [doi: 10.1145/2534169.2486011]

[5] Chole S, Fingerhut A, Ma S, Sivaraman A, Vargaftik S, Berger A, Mendelson G, Alizadeh M, Chuang ST, Keslassy I, Orda A, Edsall T. dRMT: Disaggregated programmable switching. In: Proc. of the Conf. of the ACM Special Interest Group on Data Communication. Los Angeles: ACM, 2017. 1–14. [doi: 10.1145/3098822.3098823]

[6] Sivaraman V, Narayana S, Rottenstreich O, Muthukrishnan S, Rexford J. Heavy-Hitter detection entirely in the data plane. In: Proc. of the 2017 Symp. on SDN Research. Santa Clara: ACM, 2017. 164–176. [doi: 10.1145/3050220.3063772]

[7] Popescu DA, Antichi G, Moore AW. Enabling fast hierarchical heavy hitter detection using programmable data planes. In: Proc. of the 2017 Symp. on SDN Research. Santa Clara: ACM, 2017. 191–192. [doi: 10.1145/3050220.3060606]

[8] Harrison R, Cai QZ, Gupta A, Rexford J. Network-Wide heavy hitter detection with commodity switches. In: Proc. of the 2018 Symp. on SDN Research. Los Angeles: ACM, 2018. 1–7. [doi: 10.1145/3185467.3185476]

[9] Miao R, Zeng HY, Kim C, Lee J, Yu ML. SilkRoad: Making stateful layer-4 load balancing fast and cheap using switching ASICs. In: Proc. of the 2017 Conf. of the ACM Special Interest Group on Data Communication. Los Angeles: ACM, 2017. 15–28.

[10] Lee J, Miao R, Kim C, Yu ML, Zeng HY. Stateful layer-4 load balancing in switching ASICs. In: Proc. of the 2017 SIGCOMM Posters and Demos. Los Angeles: ACM, 2017. 133–135. [doi: 10.1145/3123878.3132012]

[11] Jin X, Li XZ, Zhang HY, Soulé R, Lee J, Foster N, Kim C, Stoica I. NetCache: Balancing key-value stores with fast in-network caching. In: Proc. of the 26th Symp. on Operating Systems Principles. Shanghai: ACM, 2017. 121–136. [doi: 10.1145/3132747.3132764]

[12] Liu M, Luo L, Nelson J, Ceze L, Krishnamurthy A, Atreya K. IncBricks: Toward in-network computation with an in-network cache. In: Proc. of the 22nd Int'l Conf. on Architectural Support for Programming Languages and Operating Systems. Xi'an: ACM, 2017. 795–809. [doi: 10.1145/3037697.3037731]

[13] Sun C, Bi J, Zheng ZL, Yu H, Hu HX. NFP: Enabling network function parallelism in NFV. In: Proc. of the Conf. of the ACM Special Interest Group on Data Communication. Los Angeles: ACM, 2017. 43–56. [doi: 10.1145/3098822.3098826]

[14] Li BJ, Tan K, Luo L, Peng YQ, Luo RQ, Xu NY, Xiong YQ, Cheng P, Chen EH. ClickNP: Highly flexible and high performance network processing with reconfigurable hardware. In: Proc. of the 2016 ACM SIGCOMM Conf. Florianopolis: ACM, 2016. 1–14. [doi: 10.1145/2934872.2934897]

[15] Martins J, Ahmed M, Raiciu C, Olteanu V, Honda M, Bifulco R, Huici F. ClickOS and the art of network function virtualization. In: Proc. of the 11th USENIX Conf. on Networked Systems Design and Implementation. Seattle: USENIX Association, 2014. 459–473.

[16] Patel P, Bansal D, Yuan LH, Murthy A, Greenberg A, Maltz DA, Kern R, Kumar H, Zikos M, Wu HY, Kim C, Karri N. Ananta: Cloud scale load balancing. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 207–218. [doi: 10.1145/2534169.2486026]

[17] Gandhi R, Liu HH, Hu YC, Lu GH, Padhye J, Yuan LH, Zhang M. Duet: Cloud scale load balancing with hardware and software. ACM SIGCOMM Computer Communication Review, 2014, 44(4): 27–38. [doi: 10.1145/2740070.2626317]

[18] Jose L, Yan LS, Varghese G, McKeown N. Compiling packet programs to reconfigurable switches. In: Proc. of the 12th USENIX Symp. on Networked Systems Design and Implementation (NSDI 2015). Oakland: USENIX Association, 2015. 103–115.

[19] Zhang MH, Li GY, Wang SC, Liu C, Chen A, Hu HX, Gu GF, Li Q, Xu MW, Wu JP. Poseidon: Mitigating volumetric DDoS attacks with programmable switches. In: Proc. of the 2020 Network and Distributed Systems Security Symp. San Diego, 2020. [doi: 10.14722/ndss.2020.24007]

[20] da Costa Cordeiro WL, Marques JA, Gaspary LP. Data plane programmability beyond OpenFlow: Opportunities and challenges for network and service operations and management. Journal of Network and Systems Management, 2017, 25(4): 784–818. [doi: 10.1007/s10922-017-9423-2]

[21] Lin YSX, Bi J, Zhou Y, Zhang C, Wu JP, Liu ZZ, Zhang YR. Research and applications of programmable data plane based on P4. Chinese Journal of Computers, 2019, 42(11): 2539–2560. (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2019.02539]

[22] Michel O, Bifulco R, Rétvári G, Schmid S. The programmable data plane: Abstractions, architectures, algorithms, and applications. ACM Computing Surveys (CSUR), 2021, 54(4): 82. [doi: 10.1145/3447868]

[23] Hauser F, Häberle M, Merling D, Lindner S, Gurevich V, Zeiger F, Frank R, Menth M. A survey on data plane programming with P4:

Fundamentals, advances, and applied research. Journal of Network and Computer Applications, 2023, 212: 103561. [doi: 10.1016/j.jnca.2022.103561]

[24] Kaur S, Kumar K, Aggarwal N. A review on P4-Programmable data planes: Architecture, research efforts, and future directions. Computer Communications, 2021, 170: 109–129. [doi: 10.1016/j.comcom.2021.01.027]

[25] AlSabeh A, Khoury J, Kfoury E, Crichigno J, Bou-Harb E. A survey on security applications of P4 programmable switches and a STRIDE-based vulnerability assessment. Computer Networks, 2022, 207: 108800. [doi: 10.1016/j.comnet.2022.108800]

[26] Shevchenko N, Chick TA, O'Riordan P, Scanlon T, Woody C. Threat modeling: A summary of available methods. Pittsburgh: Carnegie Mellon University, Software Engineering Institute, 2018.

[27] Chen X, Wu CM, Liu X, Huang Q, Zhang D, Zhou HF, Yang Q, Khan MK. Empowering network security with programmable switches: A comprehensive survey. IEEE Communications Surveys & Tutorials, 2023, 25(3): 1653–1704. [doi: 10.1109/COMST.2023.3265984]

[28] Kang Q, Xue L, Morrison A, Tang YX, Chen A, Luo XP. Programmable in-network security for context-aware BYOD Policies. In: Proc. of the 29th USENIX Security Symp. Virtual Event: USENIX Association, 2020. 595–612.

[29] Wong W. BYOD: The risks of bring your own device. Risk Management, 2012, 59(5): 9.

[30] Dang-Pham D, Pittayachawan S. Comparing intention to avoid malware across contexts in a BYOD-enabled Australian university: A Protection Motivation Theory approach. Computers & Security, 2015, 48: 281–297. [doi: 10.1016/j.cose.2014.11.002]

[31] Zahadat N, Blessner P, Blackburn T, Olson BA. BYOD security engineering: A framework and its analysis. Computers & Security, 2015, 55: 81–99. [doi: 10.1016/j.cose.2015.06.011]

[32] Hong S, Baykov R, Xu L, Nadimpalli S, Gu GF. Towards SDN-Defined Programmable BYOD (Bring Your Own Device) security. In: Proc. of the 2016 Network and Distributed System Security Symp. (NDSS 2016). San Diego: Internet Society, 2016. [doi: 10.14722/ndss.2016.23458]

[33] Shin S, Yegneswaran V, Porras P, Gu GF. AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. Berlin: ACM, 2013. 413–424. [doi: 10.1145/2508859.2516684]

[34] Bajaber O, Ji B, Gao P. P4Control: Line-rate cross-host attack prevention via in-network information flow control enabled by programmable switches and eBPF. In: Proc. of the 2024 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2024. 4610–4628. [doi: 10.1109/SP54263.2024.00147]

[35] CrowdStrike. Lateral Movement Explained. 2025. https://www.crowdstrike.com/cybersecurity-101/lateral-movement/

[36] Ma SQ, Zhang XY, Xu DY. ProTracer: Towards practical provenance tracing by alternating between logging and tainting. In: Proc. of the 2016 Network and Distributed System Security Symp. (NDSS 2016). San Diego: Internet Society, 2016.

[37] Hossain MN, Milajerdi SM, Wang J, Eshete B, Gjomemo R, Sekar R, Stoller S, Venkatakrishnan VN. SLEUTH: Real-time attack scenario reconstruction from COTS audit data. In: Proc. of the 26th USENIX Security Symp. Vancouver: USENIX Association, 2017. 487–504.

[38] Fang PC, Gao P, Liu CL, Ayday E, Jee K, Wang T, Ye YF, Liu ZT, Xiao XS. Back-propagating system dependency impact for attack investigation. In: Proc. of the 31st USENIX Security Symp. Boston: USENIX Association, 2022. 2461–2478.

[39] Myers AC, Liskov B. A decentralized model for information flow control. ACM SIGOPS Operating Systems Review, 1997, 31(5): 129–142. [doi: 10.1145/269005.266669]

[40] Jung C, Kim S, Jang R, Mohaisen D, Nyang D. A scalable and dynamic ACL system for in-network defense. In: Proc. of the 2022 ACM SIGSAC Conf. on Computer and Communications Security. Los Angeles: ACM, 2022. 1679–1693. [doi: 10.1145/3548606.3560606]

[41] Aviram N, Schinzel S, Somorovsky J, Heninger N, Dankel M, Steube J, Valenta L, Adrian D, Halderman JA, Dukhovni V, Käsper E, Cohney S, Engels S, Paar C, Shavitt Y. DROWN: Breaking TLS using SSLv2. In: Proc. of the 25th USENIX Security Symp. Washington: USENIX Association, 2016. 689–706.

[42] Beurdouche B, Bhargavan K, Delignat-Lavaud A, Fournet C, Kohlweiss M, Pironti A, Strub PY, Zinzindohoue JK. A messy state of the union: Taming the composite state machines of TLS. Communications of the ACM, 2017, 60(2): 99–107. [doi: 10.1145/3023357]

[43] Checkoway S, Fredrikson M, Niederhagen R, Everspaugh A, Green M, Lange T, Ristenpart T, Bernstein DJ, Maskiewicz J, Shacham H. On the practical exploitability of dual EC in TLS implementations. In: Proc. of the 23rd USENIX Security Symp. San Diego: USENIX Association, 2014. 319–335.

[44] Amann J, Gasser O, Scheitle Q, Brent L, Carle G, Holz R. Mission accomplished? HTTPS security after diginotar. In: Proc. of the 2017 Internet Measurement Conf. London: ACM, 2017. 325–340. [doi: 10.1145/3131365.3131401]

[45] Holz R, Amann J, Mehani O, Wachs M, Ali Kaafar M. TLS in the wild: An Internet-wide analysis of TLS-based protocols for electronic communication. arXiv:1511.00341, 2016. [doi: 10.48550/arXiv.1511.00341]

[46] Richter P, Smaragdakis G, Plonka D, Berger A. Beyond counting: New perspectives on the active IPv4 address space. In: Proc. of the

2016 Internet Measurement Conf. Santa: ACM, 2016. 135–149. [doi: 10.1145/2987443.2987473]

[47] Li GY, Zhang MH, Guo C, Bao H, Xu MW, Hu HX, Li FH. IMap: Fast and Scalable In-Network scanning with programmable switches. In: Proc. of the 19th USENIX Symp. on Networked Systems Design and Implementation. Renton: USENIX Association, 2022. 667–681.

[48] Meier R, Tsankov P, Lenders V, Vanbever L, Vechev M. NetHide: Secure and practical network topology obfuscation. In: Proc. of the 27th USENIX Conf. on Security Symp. The USENIX Association, Baltimore, 2018. 693–709.

[49] Meier R, Lenders V, Vanbever L. ditto: WAN traffic obfuscation at line rate. In: Proc. of the 2022 Network and Distributed Systems Security Symp. (NDSS 2022) San Diego, 2022. [doi: 10.14722/ndss.2022.24056]

[50] Acar A, Fereidooni H, Abera T, Sikder AK, Miettinen M, Aksu H, Conti M, Sadeghi AR, Uluagac S. Peek-a-boo: I see your smart home activities, even encrypted! In: Proc. of the 13th ACM Conf. on Security and Privacy in Wireless and Mobile Networks. Linz: ACM, 2020. 207–218. [doi: 10.1145/3395351.3399421]

[51] Cai X, Nithyanand R, Johnson R. CS-BuFLo: A congestion sensitive website fingerprinting defense. In: Proc. of the 13th Workshop on Privacy in the Electronic Society. Scottsdale: ACM, 2014. 121–130. [doi: 10.1145/2665943.2665949]

[52] Chen C, Asoni DE, Barrera D, Danezis G, Perrig A. HORNET: High-speed onion routing at the network layer. In: Proc. of the 22nd ACM SIGSAC Conf. on Computer and Communications Security. Denver: ACM, 2015. 1441–1454. [doi: 10.1145/2810103.2813628]

[53] Piotrowska AM, Hayes J, Elahi T, Meiser S, Danezis G. The loopix anonymity system. In: Proc. of the 26th USENIX Conf. on Security Symp. Vancouver: USENIX Association, 2017. 1199–1216.

[54] Barman L, Dacosta I, Zamani M, Zhai EN, Ford B, Hubaux JP, Feigenbaum J. PriFi: A low-latency local-area anonymous communication network. arXiv:1710.10237, 2021. [doi: 10.48550/arXiv.1710.10237]

[55] Wang W, Motani M, Srinivasan V. Dependent link padding algorithms for low latency anonymity systems. In: Proc. of the 15th ACM Conf. on Computer and Communications Security. Alexandria: ACM, 2008. 323–332. [doi: 10.1145/1455770.1455812]

[56] Kon PTJ, Gattani A, Saharia D, Cao TY, Barradas D, Chen A, Sherr M, Ujcich BE. NetShuffle: Circumventing censorship with shuffle proxies at the edge. In: Proc. of the 2024 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2024. 3497–3514. [doi: 10.1109/SP54263.2024.00036]

[57] Dingledine R, Mathewson N, Syverson P. Tor: The second-generation onion router. In: Proc. of the 13th Conf. on USENIX Security Symp. San Diego: USENIX Association, 2004.

[58] Lantern. https://getlantern.org/

[59] Houmansadr A, Nguyen GTK, Caesar M, Borisov N. Cirripede: Circumvention infrastructure using router redirection with plausible deniability. In: Proc. of the 18th ACM Conf. on Computer and Communications Security. Chicago: ACM, 2011. 187–200. [doi: 10.1145/2046707.2046730]

[60] Karlin J, Ellard D, Jackson AW, Jones CE, Lauer G, Mankins DP, Strayer WT. Decoy routing: Toward unblockable internet communication. In: Proc. of the 2011 USENIX Workshop on Free and Open Communications on the Internet (FOCI 2011). San Francisco: USENIX Association, 2011.

[61] Fifield D, Lan C, Hynes R, Wegmann P, Paxson V. Blocking-resistant communication through domain fronting. Proc. on Privacy Enhancing Technologies, 2015(2): 46–64.

[62] Hypolite J, Sonchack J, Hershkop S, Dautenhahn N, DeHon A, Smith JM. DeepMatch: Practical deep packet inspection in the data plane using network processors. In: Proc. of the 16th Int'l Conf. on Emerging Networking Experiments and Technologies. 2020. 336–350. [doi: 10.1145/3386367.3431290]

[63] Cisco firepower threat defense. https://www.cisco.com/c/en/us/support/security/secure-firewall-4200-series/series.html

[64] Sonicwall Supermassive Series. https://www.sonicwall.com/medialibrary/en/datasheet/datasheet-sonicwall-supermassive-series.pdf

[65] Fortinet Fortigate Series. https://www.fortinet.com/products/next-generation-firewall

[66] Wang SC, Zhang MH, Li GY, Liu C, Liu Y, Jia XY, Xu MW. Making multi-string pattern matching scalable and cost-efficient with programmable switching ASICs. In: Proc. of the IEEE INFOCOM 2021—IEEE Conf. on Computer Communications. Vancouver: IEEE, 2021. 1–10. [doi: 10.1109/INFOCOM42981.2021.9488796]

[67] Aho AV, Ullman JD. The Theory of Parsing, Translation, and Compiling. Englewood Cliffs: Prentice-Hall, 1972.

[68] Gupta S, Gosain D, Kwon M, Acharya HB. DeeP4R: Deep packet inspection in P4 using packet recirculation. In: Proc. of the IEEE INFOCOM 2023— IEEE Conf. on Computer Communications. New York: IEEE, 2023. 1–10. [doi: 10.1109/INFOCOM53939.2023.10228996]

[69] Mirkovic J, Reiher P. A taxonomy of DDoS attack and DDoS defense mechanisms. ACM SIGCOMM Computer Communication Review, 2004, 34(2): 39–53. [doi: 10.1145/997150.997156]

[70] CloudFlare. The Real Cause of Large DDoS - IP Spoofing. 2018. https://blog.cloudflare.com/the-root-cause-of-large-ddos-ip-spoofing

[71] Kang MS, Lee SB, Gligor VD. The crossfire attack. In: Proc. of the 2013 IEEE Symp. on Security and Privacy. Berkeley: IEEE, 2013. 127–141. [doi: 10.1109/SP.2013.19]

[72] Hidden Threat of Pulse Wave DDoS Attacks. https://ddos-guard.net/en/info/blog-detail/hidden-threat-of-pulse-wave-ddos-attacks

[73] Mahimkar A, Dange J, Shmatikov V, Vin H, Zhang Y. dFence: Transparent network-based denial of service mitigation. In: Proc. of the 4th USENIX Conf. on Networked Systems Design & Implementation. Cambridge: USENIX Association, 2007. 327–340.

[74] Fayaz SK, Tobioka Y, Sekar V, Bailey M. Bohatei: Flexible and elastic DDoS defense. In: Proc. of the 24th USENIX Security Symp. Washington: USENIX Association, 2015. 817–832.

[75] Liu ZX, Namkung H, Nikolaidis G, Lee J, Kim C, Jin X, Braverman V, Yu ML, Sekar V. Jaqen: A high-performance switch-native approach for detecting and mitigating volumetric DDoS attacks with programmable switches. In: Proc. of the 30th USENIX Security Symp. Virtual Event: USENIX Association, 2021. 3829–3846.

[76] Cisco IOS NetFlow. https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/index.html

[77] Braverman V, Ostrovsky R. Zero-one frequency laws. In: Proc. of the 42nd ACM Symp. on Theory of Computing. Cambridge: ACM, 2010. 281–290. [doi: 10.1145/1806689.1806729]

[78] Liu ZX, Manousis A, Vorsanger G, Sekar V, Braverman V. One sketch to rule them all: Rethinking network flow monitoring with univmon. In: Proc. of the 2016 ACM SIGCOMM Conf. Florianopolis: ACM, 2016. 101–114. [doi: 10.1145/2934872.2934906]

[79] Li GY, Zhang MH, Liu C, Kong X, Chen A, Gu GF, Duan HX. NETHCF: Enabling line-rate and adaptive spoofed IP traffic filtering. In: Proc. of the 27th Int'l Conf. on Network Protocols (ICNP). Chicago: IEEE, 2019. 1–12. [doi: 10.1109/ICNP.2019.8888057]

[80] Jin C, Wang HN, Shin KG. Hop-count filtering: An effective defense against spoofed DDoS traffic. In: Proc. of the 10th ACM Conf. on Computer and Communications Security. Washington: ACM, 2003. 30–41. [doi: 10.1145/948109.948116]

[81] Wang HN, Jin C, Shin KG. Defense against spoofed IP traffic using hop-count filtering. IEEE/ACM Trans. on Networking, 2007, 15(1): 40–53. [doi: 10.1109/TNET.2006.890133]

[82] Xing JR, Wu WQ, Chen A. Ripple: A programmable, decentralized link-flooding defense against adaptive adversaries. In: Proc. of the 30th USENIX Security Symp. Virtual Event: USENIX Association, 2021. 3865–3880.

[83] Wang J, Wen R, Li JQ, Yan F, Zhao B, Yu FJ. Detecting and mitigating target link-flooding attacks using SDN. IEEE Trans. on Dependable and Secure Computing, 2019, 16(6): 944–956. [doi: 10.1109/TDSC.2018.2822275]

[84] Wang L, Li Q, Jiang Y, Jia XY, Wu JP. Woodpecker: Detecting and mitigating link-flooding attacks via SDN. Computer Networks, 2018, 147: 1–13. [doi: 10.1016/j.comnet.2018.09.021]

[85] Zheng J, Li Q, Gu GF, Cao JH, Yau DKY, Wu JP. Realtime DDoS defense using COTS SDN switches via adaptive correlation analysis. IEEE Trans. on Information Forensics and Security, 2018, 13(7): 1838–1853. [doi: 10.1109/TIFS.2018.2805600]

[86] Zhou HC, Hong SM, Liu YY, Luo XP, Li WC, Gu GF. Mew: Enabling large-scale and dynamic link-flooding defenses on programmable switches. In: Proc. of the 2023 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2023. 3178–3192. [doi: 10.1109/SP46215.2023.10179404]

[87] Alcoz AG, Strohmeier M, Lenders V, Vanbever L. Aggregate-based congestion control for pulse-wave DDoS defense. In: Proc. of the 2022 ACM SIGCOMM Conf. Amsterdam: ACM, 2022. 693–706. [doi: 10.1145/3544216.3544263]

[88] Mahajan R, Bellovin SM, Floyd S, Ioannidis J, Paxson V, Shenker S. Controlling high bandwidth aggregates in the network. ACM SIGCOMM Computer Communication Review, 2002, 32(3): 62–73. [doi: 10.1145/571697.571724]

[89] Misa C, Durairajan R, Gupta A, Rejaie R, Willinger W. Leveraging prefix structure to detect volumetric DDoS attack signatures with programmable switches. In: Proc. of the 2024 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2024. 4535–4553. [doi: 10.1109/SP54263.2024.00267]

[90] da Silveira Ilha A, Lapolli ÂC, Marques JA, Gaspary LP. Euclid: A fully in-network, P4-based approach for real-time DDoS attack detection and mitigation. IEEE Trans. on Network and Service Management, 2021, 18(3): 3121–3139. [doi: 10.1109/TNSM.2020.3048265]

[91] Doriguzzi-Corin R, Millar S, Scott-Hayward S, Martinez-Del-rincon J, Siracusa D. LUCID: A practical, lightweight deep learning solution for DDoS attack detection. IEEE Trans. on Network and Service Management, 2020, 17(2): 876–889. [doi: 10.1109/TNSM.2020.2971776]

[92] Barradas D, Santos N, Rodrigues L, Signorello S, Ramos FMV, Madeira A. FlowLens: Enabling efficient flow classification for ML-based network security applications. In: Proc. of the 2021 Network and Distributed Systems Security (NDSS) Symp. Virtual, 2021. [doi: 10.14722/ndss.2021.24067]

[93] Akem ATJ, Gucciardo M, Fiore M. Flowrest: Practical flow-level inference in programmable switches with random forests. In: Proc. of the 2023 IEEE Conf. on Computer Communications (INFOCOM 2023). New York: IEEE, 2023. 1–10. [doi: 10.1109/INFOCOM53939.2023.10229100]

[94] Xie GR, Li Q, Dong YT, Duan GL, Jiang Y, Duan JP. Mousika: Enable general in-network intelligence in programmable switches by knowledge distillation. In: Proc. of the 2022 IEEE Conf. on Computer Communications (INFOCOM 2022). London: IEEE, 2022. 1938–1947. [doi: 10.1109/INFOCOM48880.2022.9796936]

[95] Kupreev O, Gutnikov A, Shmelev Y. DDoS Attacks in Q3 2022. Kaspersky Lab Technical Report. 2022. https://securelist.com/ddos-report-q3-2022/107860/

[96] Microsoft TechNet. Syn Attack Protection on Windows Vista, Windows 2008, Windows 7, Windows 2008 R2, Windows 8/8.1, Windows 2012, and Windows 2012 R2. 2014. https://docs.microsoft.com/en-us/answers/questions/144446/synattackprotect.html

[97] Yoo S, Chen XQ, Rexford J. SmartCookie: Blocking large-scale SYN floods with a split-proxy defense on programmable data planes. In: Proc. of the 33rd USENIX Conf. on Security Symp. Philadelphia: USENIX Association, 2024. 217–234.

[98] Aumasson JP, Bernstein DJ. SipHash: A fast short-input PRF. In: Proc. of the 12th Int'l Conf. on Cryptology in India. Berlin: Springer, 2012. 489–508. [doi: 10.1007/978-3-642-34931-7_28]

[99] Zhou Y, Sun C, Liu HH, Miao R, Bai S, Li B, Zheng ZL, Zhu LJ, Shen Z, Xi YQ, Zhang PC, Cai D, Zhang M, Xu MW. Flow event telemetry on programmable data plane. In: Proc. of the 2020 Annual Conf. of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication. Virtual Event: ACM, 2020. 76–89. [doi: 10.1145/3387514.3406214]

[100] Huang Q, Sun HF, Lee PPC, Bai W, Zhu F, Bao YG. OmniMon: Re-architecting network telemetry with resource efficiency and full accuracy. In: Proc. of the 2020 Annual Conf. of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication. Virtual Event: ACM, 2020. 404–421. [doi: 10.1145/3387514.3405877]

[101] Zhou HC, Gu GF. Cerberus: Enabling efficient and effective in-network monitoring on programmable switches. In: Proc. of the 2024 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2024. 4424–4439. [doi: 10.1109/SP54263.2024.00016]

[102] Xing JR, Kang Q, Chen A. NetWarden: Mitigating network covert channels while preserving performance. In: Proc. of the 29th USENIX Security Symp. Virtual Event: USENIX Association, 2020. 2039–2056.

[103] Peng P, Ning P, Reeves DS. On the secrecy of timing-based active watermarking trace-back techniques. In: Proc. of the 2006 IEEE Symp. on Security and Privacy (S&P'06). Berkeley/Oakland: IEEE, 2006. 334–349. [doi: 10.1109/SP.2006.28]

[104] Jafri SU, Rao S, Shrivastav V, Tawarmalani M. Leo: Online ML-based traffic classification at multi-terabit line rate. In: Proc. of the 21st USENIX Symp. on Networked Systems Design and Implementation. Santa Clara: USENIX Association, 2024. 1573–1591.

[105] Loh WY. Classification and regression trees. WIREs Data Mining and Knowledge Discovery, 2011, 1(1): 14–23. [doi: 10.1002/widm.8]

[106] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv:1503.02531, 2015.

[107] Xie GR, Li Q, Lin JY, Antichi G, Zhao D, Yuan ZH, Li RY, Jiang Y. Dryad: Deploying adaptive trees on programmable switches for networking classification. In: Proc. of the 31st Int'l Conf. on Network Protocols (ICNP). Reykjavik: IEEE, 2023. 1–11. [doi: 10.1109/ICNP59255.2023.10355629]

[108] Xie GR, Li Q, Yan HL, Zhao D, Antichi G, Jiang Y. Efficient attack detection with multi-latency neural models on heterogeneous network devices. In: Proc. of the 31st Int'l Conf. on Network Protocols (ICNP). Reykjavik: IEEE, 2023. 1–12. [doi: 10.1109/ICNP59255.2023.10355579]

[109] Liu H, Simonyan K, Yang YM. DARTS: Differentiable architecture search. arXiv:1806.09055, 2019.

[110] Cai H, Gan C, Wang TZ, Zhang ZK, Han S. Once-for-all: Train one network and specialize it for efficient deployment. arXiv:1908.09791, 2020.

[111] Bao CT, Xu LH, Goodman ED, Cao LL. A novel non-dominated sorting algorithm for evolutionary multi-objective optimization. Journal of Computational Science, 2017, 23: 31–43. [doi: 10.1016/j.jocs.2017.09.015]

[112] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation, 2002, 6(2): 182–197. [doi: 10.1109/4235.996017]

[113] Lin JY, Li Q, Xie GR, Jiang Y, Yuan ZH, Jiang CL, Yang Y. In-Forest: Distributed in-network classification with ensemble models. In: Proc. of the 31st Int'l Conf. on Network Protocols (ICNP). Reykjavik: IEEE, 2023. 1–12. [doi: 10.1109/ICNP59255.2023.10355602]

[114] Zhou GM, Liu ZT, Fu CP, Li Q, Xu K. An efficient design of intelligent network data plane. In: Proc. of the 32nd USENIX Conf. on Security Symp. Anaheim: USENIX Association, 2023. 6203–6220.

[115] Dong YT, Li Q, Wu KD, Li RY, Zhao D, Tyson G, Peng JK, Jiang Y, Xia ST, Xu MW. HorusEye: A realtime iot malicious traffic detection framework using programmable switches. In: Proc. of the 32nd USENIX Security Symp. Anaheim: USENIX Association, 2023. 571–588.

[116] Liu FT, Ting KM, Zhou ZH. Isolation forest. In: Proc. of the 8th IEEE Int'l Conf. on Data Mining. Pisa: IEEE, 2008. 413–422. [doi: 10.1109/ICDM.2008.17]

[117] Mandic DP, Chambers J. Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability. New York: John Wiley & Sons, Inc. , 2001.
[118] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In: Proc. of the 31st Int'l Conf. on Neural Information Processing Systems. Long Beach: Curran Associates Inc. , 2017. 6000–6010.
[119] Yan JZ, Xu HT, Liu ZT, Li Q, Xu K, Xu MW, Wu JP. Brain-on-switch: Towards advanced intelligent network data plane via NN-Driven traffic analysis at line-speed. In: Proc. of the 21st USENIX Symp. on Networked Systems Design and Implementation. Santa Clara: USENIX Association, 2024. 419–440.
[120] Availability of Linux RDMA on Microsoft Azure. https://azure.microsoft.com/en-us/blog/azure-linux-rdma-hpc-available/
[121] Rothenberger B, Taranov K, Perrig A, Hoefler T. ReDMArk: Bypassing RDMA security mechanisms. In: Proc. of the 30th USENIX Security Symp. Virtual Event: USENIX Association, 2021. 4277–4292.
[122] Simpson AK, Szekeres A, Nelson J, Zhang I. Securing RDMA for high-performance datacenter storage systems. In: Proc. of the 12th USENIX Conf. on Hot Topics in Cloud Computing. USENIX Association, 2020.
[123] Taranov K, Rothenberger B, Perrig A, Hoefler T. sRDMA--Efficient NIC-based authentication and encryption for remote direct memory access. In: Proc. of the 2020 USENIX Annual Technical Conf. USENIX Association, 2020. 691–704.
[124] Xing JR, Hsu KF, Qiu YM, Yang ZY, Liu HY, Chen A. Bedrock: Programmable network support for secure RDMA systems. In: Proc. of the 31st USENIX Security Symp. Boston: USENIX Association, 2022. 2585–2600.
[125] Chen XQ. Implementing AES encryption on programmable switches via scrambled lookup tables. In: Proc. of the Workshop on Secure Programmable Network Infrastructure. Virtual Event: ACM, 2020. 8–14. [doi: 10.1145/3405669.3405819]

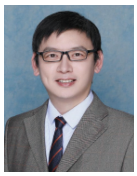
附中文参考文献:

[21] 林耘森箫, 毕军, 周禹, 张程, 吴建平, 刘争争, 张乙然. 基于 P4 的可编程数据平面研究及其应用. 计算机学报, 2019, 42(11): 2539–2560. [doi: 10.11897/SP.J.1016.2019.02539]

邹志凯(2001－), 男, 硕士生, 主要研究领域为云网络.

沃天宇(1978－), 男, 博士, 教授, CCF 高级会员, 主要研究领域为分布式计算, 工业互联网, 系统软件.

张梦豪(1994－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为计算机网络, 网络系统, 深度学习系统, 网络安全.



胡春明(1977－), 男, 博士, 教授, CCF 杰出会员, 主要研究领域为分布式计算, 云计算, 软件工程.

李冠宇(1996－). 男, 博士, 主要研究领域为高性能网络, 可编程网络, 网络芯片架构.

徐明伟(1971－), 男, 博士, 教授, CCF 高级会员, 主要研究领域为互联网体系结构, 大规模路由, 网络空间安全.

杨任宇(1989－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为分布式计算, 软件可靠性, 深度学习系统.