基于SDN的物联网边缘节点间数据流零信任管理

肖警续，郭渊博，常朝稳，吴平，杨晨立
（信息工程大学密码工程学院，河南 郑州 450001）

摘　要：针对物联网缺少对数据流传输链路中恶意交换节点检测与定位的有效手段，提出了一种基于软件定义网络（SDN）的物联网边缘节点间数据流零信任管理方法。该方法将SDN架构应用到边缘节点间数据流的传输过程，使用固定长度的报头开销对数据流、节点和路径进行零信任管理，实现轻量级的数据包转发验证和恶意交换节点定位功能。在转发路径中，交换节点对数据包进行安全验证并统计验证信息，保证数据流传输的安全性和路径的一致性。根据异常数据包类型，控制器采用二分法标记执行验证操作的交换节点，逐步缩小恶意交换节点的范围，实现对多类型恶意交换节点的定位。最后，对所提方法进行了仿真与评估。实验结果表明，所提方法引入小于10%的转发时延和低于8%的吞吐量损失。

关键词：物联网；软件定义网络；零信任管理；异常检测；异常定位
中图分类号：TP393
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2024060

Zero trust management of data flow between IoT edge nodes based on SDN

XIAO Jingxu, GUO Yuanbo, CHANG Chaowen, WU Ping, YANG Chenli
Department of Cryptogram Engineering, Information Engineering University, Zhengzhou 450001, China

Abstract: Aiming at the lack of effective means for detecting and localizing malicious nodes in the data flow transmission link in Internet of things (IoT), a zero trust management method of data flow between IoT edge nodes based on software defined network (SDN) was proposed. This method applied the architecture of SDN to the process of data flow transmission between edge nodes. A fixed-length header overhead was used for zero trust management of data flow, nodes, and paths to achieve lightweight packet forwarding verification and malicious node localization functions. In the forwarding path, the security verification of packets was performed by the switching node, and the verification information was counted to ensure the security of the data flow transmission and the consistency of the path. Based on the type of abnormal packets, the controller adopted dichotomous method to mark the switching node that performed the verification operation to gradually narrow down the scope of malicious nodes, and realized the localization of multiple types of malicious nodes. Finally, the proposed method was simulated and evaluated. The experimental results show that the method introduces a forwarding delay of less than 10% and a throughput loss of less than 8%.

Keywords: Internet of things, software defined network, zero trust management, anomaly detection, anomaly location

0　引言

物联网（IoT, Internet of things）[1]是指通过网络连接各种物理设备，使它们能够收集、传输和处理数据的技术，其基于万物互联的理念在智能家

收稿日期：2023–11–10；修回日期：2024–01–04
通信作者：肖警续，xiaojingxu2301@163.com
基金项目：河南省科技攻关基金资助项目（No.222102210070）
Foundation Item：Henan Provincial Science and Technology Research Project (No.222102210070)

居、智能交通、智能医疗等领域具有非常广泛的应用。云计算作为一种网络计算模式，已在物联网领域取得广泛应用[2]。然而，在实际应用中，由于云数据中心远离物联网终端设备，全部数据传输至云端计算容易产生较高时延和安全隐患[3]。为解决这一问题，边缘计算将计算、存储和网络服务部署在离数据源更近的边缘节点，如边缘计算节点和雾计算节点（本文将雾计算视为边缘计算的一种扩展）。边缘节点通过执行本地的计算任务，只将部分必要数据发送至云端，能有效降低数据传输时延，提高实时性[4-5]。同时，边缘计算能有效减轻云端服务器的压力，节省网络带宽和计算资源。因此，通过云计算和边缘计算的结合，有助于构建更灵活、高效、安全的物联网系统。

物联网的安全问题一直是物联网相关研究的重点。边缘节点通过建立访问控制机制或对访问物联网的设备进行身份验证保证流入物联网的数据的安全性[6]。通过对数据进行加密与安全验证保证数据传输的机密性和完整性[7]。但上述方案往往忽略了数据在边缘节点与云服务器之间，或在多接入边缘计算（MEC, multi-access edge computing）中边缘节点之间传输等情况下通信链路存在的安全威胁[8]。通信链路中恶意交换节点的出现会对数据进行窃取、篡改或删除等行为[9]，影响物联网数据的安全传输。若通信链路中缺少对数据的安全验证，会导致接收端收到异常数据。物联网中常用的端到端的数据安全验证方案虽然能识别到异常数据，但无法对通信链路中的恶意交换节点进行定位，导致恶意交换节点持续威胁数据的安全传输。因此，确保边缘节点与云端、边缘节点间传输路径的安全，以及对通信链路中的异常节点进行有效的检测与定位，是保证物联网数据安全传输的关键。

软件定义网络（SDN, software defined network）[10]是一种将网络控制层和数据层分离，使网络可以通过软件进行灵活配置和管理的技术。SDN具有集中管理、可扩展和灵活可编程的特性，在物联网中部署SDN架构，有助于实现物联网多方面的安全需求。Javanmardi等[11]对SDN在物联网雾（IoT-Fog）网络中的应用架构进行总结，通过将雾节点和雾网关划为SDN的数据层，云数据中心和云网关组成SDN控制层，控制层基于SDN集中管理的特性实现对网络中所有节点的观察与监控。雾网关对所有流使用SDN控制层注入的流规则进行处理，实现对网络配置的灵活管理，提高IoT-Fog网络的操作性和安全性。SDN与边缘计算的结合已应用于多个物联网场景中，例如，通过SDN控制层的OpenFlow消息对蜂窝基站和边缘云进行控制，有效解决了移动边缘计算中的任务卸载和资源分配问题[12]；对由“云、管、边、端”构成的配电物联网体系架构中的“边”和“端”进行融合，通过软件定义的方式对融合终端进行控制与管理，实现“端”层与“云”层的高效协同，提升了配电物联网整体的计算能力[13]；集中式SDN控制器通过对路侧设备（RSU, road side unit）和5G基站进行集中控制，实现车载物联网信息收集、交通监控、实时报告和网络控制等功能，从而保证安全可靠的车载物联网环境[14]。因此，SDN可以对边缘节点间链路的安全研究提供新思路与技术支持，通过对边缘节点SDN化，边缘节点根据控制器下发的规则对数据流进行安全验证，实现对通信链路中恶意交换节点的定位过程。此外，当通信链路中出现恶意交换节点时，SDN可以及时更新数据传输路径，避免恶意交换节点对数据传输安全的持续威胁，提高网络的弹性。

针对物联网在边缘节点与云端、边缘节点间数据传输过程中缺少完善的安全验证方案，以及不具备对通信链路中恶意交换节点的检测能力等问题，本文将SDN架构应用到物联网中，引入零信任的思想，即对数据、节点和路径执行“不信任，始终验证”的操作，提出了一种基于SDN的物联网边缘节点间数据流零信任管理方法SDN-ZTM（SDN-based zero-trust management），SDN-ZTM同样适用于数据在边缘节点与云之间传输的情况。本文将对数据包的转发验证过程与恶意交换节点的定位过程进行结合，利用SDN灵活可编程的特性，在交换节点端完成对数据包的转发验证，同时在控制器端对恶意交换节点进行有效的定位。

本文的主要工作如下。

1) 提出了SDN-ZTM方法，该方法仅分配较少且恒定的验证字段空间，并基于二分法有效减少执行验证的交换节点数量，以较小的开销实现数据包的转发验证。

2) SDN-ZTM能实时发现并阻断数据包篡改、转发路径异常和恶意丢弃等攻击行为，确保边缘节

点间数据包的安全转发；同时对转发路径中的恶意交换节点进行有效定位，保障数据包转发路径的安全性。

3) 在虚拟网络中通过P4（programming protocol-independent packet processor）[15]交换设备对SDN-ZTM进行仿真与评估，与相关方案相比，SDN-ZTM具有更全面的安全性能，并且性能开销相对较低。实验结果表明，SDN-ZTM具有有效性，是一种适合物联网应用环境的轻量级方案。

1　相关工作

为实现物联网数据的安全传输，当前的安全防御措施主要分为防火墙、异常检测、访问控制和密码技术等。Sadiq等[16]结合SDN实现对物联网中分布式拒绝服务（DDoS, distributed denial of service）攻击的防御。虽然防火墙产生的开销较低，但其无法对内部攻击和未知攻击进行有效防御。Nguyen等[17]提出的SeArch分别在物联网云、雾和边缘3个不同层级建立入侵检测系统，通过同层和不同层入侵检测系统的交互，最终在SDN控制层实现对异常流量的检测。Khan等[18]采用分布式深度学习的方法，使SDN控制器通过对物联网设备训练的数据进行存储并对威胁情报进行简化，实现对物联网数据的异常检测。但该异常检测方案需要对物联网流量进行实时的监控与统计，会产生一定的检测时延，难以实时阻断恶意流量攻击，且无法根据检测到的异常数据对通信链路中的恶意交换节点进行定位。Gao等[19]通过收集请求者发送至被请求者的流信息，并基于区块链构建身份验证的模式，实现对物联网雾网络架构的安全管理。Xie等[20]将区块链技术应用到车载网络中，系统中每辆车都包含一个道路信息，其周边车辆会对该信息进行真实性评分，从而保证信息的正确性。Oh等[21]提出了一种基于区块链的安全数据共享系统，通过雾节点将信息存储在区块链中，实现用户与云之间的身份验证。区块链虽然能保障物联网中数据的完整性，但节点间会进行大量的信息交互，造成较高的通信开销，因此不适用于需要较快反馈的物联网设备。Torres-Charles等[22]提出的SecMesh是一种用于边缘、雾、云基础设施中流处理的信息安全方法，该方法通过在边缘、雾和云间构建基于安全的虚拟通道实现数据流的安全传输，但并未对实际路径中节点的安全性进行讨论。Mohan等[23]通过SDN控制器对流规则进行加密，使用SDN交换机作为雾层服务器对流规则进行解密，从而保证流规则的完整性，但该方案无法对异常数据流进行识别。

SDN为物联网提供了灵活的路径选择功能，有效提升了物联网对数据的管理效率。Hu等[24]提出的FitPath方案根据带宽成本、时延和丢包率等多维因素，制定最小化链路总成本，在基于SDN的边缘计算分布式网络架构中构建边缘到边缘的数据路由路径。Absardi等[25]为物联网监控系统提出了一种基于深度学习的网络流量管理策略，并在SDN控制器实现边缘节点与云数据中心之间最优路由的预测，从而降低端到端的丢包率，提高服务质量（QoS, quality of service）。但FitPath方案和文献[25]方案都仅实现面向服务质量的路径选择，未对数据在路径中的安全性进行考虑。Liu等[26]提出了一种基于深度强化学习的车载自组网智能安全消息转发策略。该策略引入节点安全（信任分数和是否为恶意交换节点）属性作为最优转发路径的依据之一，但未对节点安全性的计算方法进行说明。

SDN架构便于实现对数据转发路径的安全验证。Zhang等[27]通过计数器对网络中的数据流进行统计，并根据流计数器方程组进行验证，从而检测出网络中的异常转发。但该方案存在滞后性，即无法对异常数据流进行实时的阻断，并且会在控制器端产生较大工作量。Hessam等[28]通过在OpenFlow协议的标准中使用cookie字段来放入相应流规则的摘要值，可检测网络中异常的交换设备，但该方案需要对所有交换设备进行验证，会产生较大的时间开销。Kim等[29]提出一种对数据包进行源认证和转发路径验证的方法OPT（origin and path trace），其通过在每个交换设备端对数据包进行验证确保数据包来源和路径的正确性。SDNsec[30]架构是一种轻量级且对数据包转发路径进行验证的方法，但其需要出口交换机向控制器发送Packet-In消息，由控制器对数据包的路径一致性进行验证，这会增加控制器的负担并对数据包传输产生较高的时延。P4Label[31]是一种基于P4的软件定义网络的数据包转发控制机制，可实现对异常数据和路径的检测，但P4Label向数据包添加的额外报头长度较长，同时产生的密钥存储开销较大。吴平等[32]使用短消息认证码对SDN架构中的数据包进行转发验证。但

当转发路径较短或恶意交换设备出现在目的主机附近时，该方案难以通过多个交换设备实现对短消息认证码的叠加认证以提升检测准确性，会造成较高的漏报率。Ren等[33]提出了一种基于多路径弹性路由的内生安全SDN架构。该架构可提供多路径比较、多路径加权和多路径随机3种数据转发模式，但仅在多路径比较数据转发模式下能对异常数据和节点进行检测，其余数据转发模式未包含对路径的安全验证过程。同时，多路径转发会占用更多的网络资源。

表1对相关研究方案进行了总结。从表1中可以看出，当前针对数据在物联网传输链路的安全研究较少。当通信链路中出现恶意交换节点时，若不及时对异常节点进行检测，会持续影响数据的安全传输，危害物联网安全。此外，相关方案在对恶意交换节点的检测与定位过程中往往产生较大的通信或验证开销，并且可能存在一定的滞后性。因此，研究一种低开销、能实时阻断异常数据并且定位链路中恶意交换节点的方法，实现数据在物联网中的安全传输是十分必要的。

2 问题描述

定义物联网中具备转发数据功能的网络设备为交换节点，当数据在物联网边缘节点间或边缘节点与云数据中心传输时，如果路径中的交换节点被入侵或攻击成为恶意交换节点，则会对数据流的安全传输造成危害。本节针对恶意交换节点对数据传输造成的安全威胁和本文的安全目标进行介绍。

2.1 威胁模型

恶意交换节点可能会对数据包进行欺骗攻击、篡改攻击、转发路径异常攻击和恶意丢弃等行为，从而威胁数据流的安全转发。以数据流在物联网边缘节点间传输的情况为例，对恶意交换节点在数据包转发过程中造成的安全威胁进行描述，如图1所示。

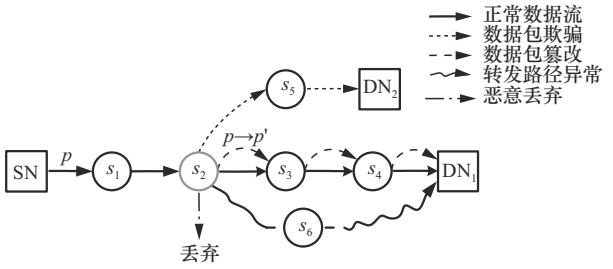


图1 恶意交换节点对数据包转发的安全威胁

图1中SN表示源边缘节点，DN_1和DN_2分别表示目的边缘节点和异常目的边缘节点，p表示待转发的数据包，s_1~s_6表示路径中的交换节点，其中s_2为恶意交换节点，具备对数据包内容和数据包转发规则进行更改的能力。图1中实直线表示正常情况下数据包的转发过程，下面分别对图中不同类型的数据包转发异常进行描述。

数据包欺骗：恶意交换节点通过对数据包源地

表1 相关研究方案总结

方案	安全功能	主要技术	存在问题
文献[16]	过滤恶意流量	防火墙	无法对内部攻击和未知攻击进行防御
文献[17-18]	检测恶意流量	异常检测	对流量的监测与统计造成较大开销，检测存在滞后性，无法定位恶意交换节点
文献[19,21]	设备身份认证	区块链	多个节点记录和认证信息，产生较高的通信开销
文献[20]	保证数据正确性	区块链	数据验证的时间成本较高
文献[22]	数据安全传输	安全虚拟通道	未对实际路径中节点的安全性进行讨论
文献[23]	流规则完整性	流规则加密	无法对异常数据流进行识别并阻断
文献[24-25]	最优路径选取	深度学习	仅考虑QoS，未考虑数据在路径中的安全性
文献[26]	安全路径选取	强加学习	未对路径中节点安全性进行分析
文献[27]	异常流量检测	流计数方程组	交换设备与控制器间通信开销较大，且无法对异常数据流进行实时阻断
文献[28]	识别异常交换设备	信息摘要	需对所有交换设备进行验证，产生较大的时间开销
文献[29-30]	源认证、转发路径一致性验证	基于消息认证码的逐跳验证	添加的报头长度随路径长度增长
文献[31]	数据包转发验证	数字签名	密钥存储开销较大，验证开销较大
文献[32]	恶意节点定位	短消息认证码	开销较小，但对不同节点的检测准确率不同
文献[33]	数据包安全验证、异常节点检测	多路径路由	数据多路径转发会增加网络资源的占用

址、目的地址在内的报头信息进行修改，导致后续交换节点对数据包执行异常的匹配转发，并将接收到的数据包转发至非法目的地址，如图1中将数据包p非法转发至DN_2。

数据包篡改：恶意交换节点会对接收到的数据包进行篡改，即将原始的数据包p篡改成异常数据包p'。对数据包载荷的篡改（如恶意代码的注入）会对目的边缘节点造成安全威胁。

转发路径异常：恶意交换节点未按照规定转发路径对数据包进行转发。转发路径异常分为路径跳转、路径绕路和路径篡改3种类型，如图2所示，其中，路径跳转指数据包不经过正常转发路径上的所有交换节点，只在规定转发路径中的部分交换节点上进行转发；路径绕路指数据包在转发过程中经过额外交换节点的转发，增加数据包的转发路径长度；路径篡改指恶意交换节点选取与正常转发路径不同的路径对数据包进行转发。转发路径异常会增加数据包的时延、影响网络性能、造成数据的丢失以及拒绝服务攻击等安全性问题。

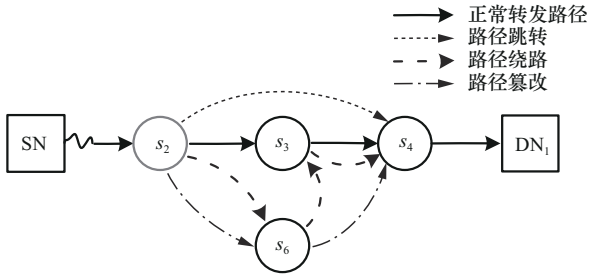


图2　转发路径异常类型

恶意丢弃：恶意交换节点对流经的数据包执行丢弃的动作，使其无法进入下一交换节点进行正常的转发。

2.2　目标

针对图1中的安全威胁以及相关研究方案存在的不足，结合物联网对数据流安全转发的需求，本文希望所提方案可实现以下目标与挑战。

1) 对异常数据包检测的同时实现对恶意交换节点的定位，单一的异常数据包检测方案虽然能阻止异常数据的流入，但未检测的恶意交换节点会持续对数据传输安全造成危害。

2) 实现恒定且轻量的额外报头开销，避免在对数据包验证过程中产生较大或与交换节点数量线性增长的报头开销，保证数据包的传输效率。

3) 实现交换节点轻量级计算，避免交换节点产生复杂、频繁的计算和通信开销，保证物联网中数据包的转发性能。

3　数据流零信任管理SDN-ZTM

本节从SDN-ZTM架构、会话密钥的生成、数据包转发验证字段（PFV, packet forwording verification）报头结构、基于PFV的数据包转发验证和恶意交换节点定位等方面对SDN-ZTM进行具体介绍，为了对SDN-ZTM方法的描述更加清晰，对表2所示的标识符进行定义。

表2　　标识符的定义

标识符	定义
p	待转发的数据包
s_i	交换节点标识，s_0和s_n分别表示传输路径中入口和出口边界的交换节点
C	控制器标识
PFV	数据包转发验证字段
FlowID	数据流标识
$\mathrm{PATH}_{\mathrm{FlowID}}$	标识为FlowID的数据流转发路径
l	$\mathrm{PATH}_{\mathrm{FlowID}}$的长度
KGC	密钥生成中心（KGC, key generation center）
$(\mathrm{kp}_{s_i},\mathrm{ks}_{s_i})$	s_i的公钥kp_{s_i}和私钥ks_{s_i}
$K_{0,i}$	s_0与s_i之间的会话密钥
K_i	控制器与s_i之间的会话密钥
$H_1(),H_2(),H_3()$	哈希函数
$\mathrm{MAC}_K()$	基于密钥K的消息认证码生成函数
h_i	转发路径一致性验证的信息，由控制器向$\mathrm{PATH}_{\mathrm{FlowID}}$中的$s_i$下发

3.1　SDN-ZTM架构

SDN-ZTM架构通过部署SDN控制器，对物联网边缘节点间或边缘节点与云数据中心间的数据转发进行管理，本文只以数据在边缘节点间传输的情况为例进行描述。定义物联网中对数据具有转发功能的网络设备为交换节点，SDN-ZTM中的数据流通过多个边缘计算节点进行传输，因此SDN-ZTM中的交换节点即边缘计算节点。SDN-ZTM将交换节点进行SDN化，使其能与控制器通信。交换节点基于消息认证码[34]（MAC, message authentication code）对数据包进行安全验证并统计验证信息，保证数据流传输的安全性和路径的一致性。控制器根据异常数据包类型，采用二分法逐步缩小恶意交换节点的范围，并结合恶意丢弃阈值实现对多

类型恶意交换节点的定位。SDN-ZTM 架构如图 3 所示。其中，SN 和 DN 分别为物联网中源边缘节点和目的边缘节点，数据流通过通信链路中的边缘计算节点进行传输，后文统一用交换节点 s_i 表示。假设从 SN 向 DN 发送数据流，结合图 3 对 SDN-ZTM 的工作流程进行描述，具体步骤如下。

步骤 1 入口交换节点 s_0 根据数据流的标识信息判断是否存在该数据流的转发路径信息，若存在，则执行步骤 3；若不存在，则表明数据流首次流经 s_0，s_0 将数据流中的首个数据包 p 的报头信息发送至控制器。控制器根据报头信息对数据流制定转发路径 $\mathrm{PATH}_{\mathrm{FlowID}}=(s_0,s_1,\cdots,s_n)$，并将转发规则、会话密钥 K_i 以及用于转发路径一致性验证的信息 h_i 下发至路径中的交换节点。

步骤 2 控制器基于二分法从转发路径中选取 s_i，并与 s_0 进行密钥协商，生成会话密钥 $K_{0,i}$。

步骤 3 s_0 对接收到的 p 添加数据转发验证字段 PFV，下游交换节点基于 PFV 对 p 进行转发验证。

步骤 4 每个 s_i 首先对 p 进行路径一致性验证，验证后向 PFV 插入关于数据包转发的认证消息，声明 p 经 s_i 转发。同时，当 s_i 在步骤 2 中被选取或 s_i 为出口交换节点时，s_i 对 p 进行数据完整性验证。s_i 只对通过验证的 p 进行转发，丢弃未通过验证的 p 并将其 PFV 字段发送至控制器，控制器基于 PFV 更新步骤 2 中对 s_i 的选取，并实现对恶意交换节点的定位。

步骤 5 s_n 对通过验证的 p 的 PFV 字段进行删除并将 p 发送至 DN，完成对 p 的转发验证过程。

SDN-ZTM 的有效实现依托于如下安全假设。

1) 密码原语具备安全性，MAC、共享密钥的生成以及密钥的保存与分发过程是安全的。

2) SDN 控制器是安全的[35]，且控制器与交换节点之间的通信是安全的，可使用加密信道对重要信息进行传输实现，如传输层安全（TLS, transport layer security）协议[36]。

3) 路径两端负责数据包传入和传出的边界交换节点是安全的，且作为数据包发送方的边缘节点是安全的，可通过基于权限和身份等信息对其进行访问控制实现[37]。

3.2 会话密钥的生成

SDN-ZTM 基于 ECDHE（ephemeral elliptic curve Diffie-Hellman）算法进行密钥协商[38]，构建会话密钥。

KGC 生成系统主私钥 msk，并公开系统参数 $\langle E,n,P,H\rangle$，其中，E 表示椭圆曲线，n 为 E 的阶数，P 为 E 上的一点，H 为哈希函数。基于控制器和交换节点的标识信息得到 C 和 s_i 的公私钥对，如式(1)所示。

$$\begin{aligned}\left\langle \mathrm{kp}_C,\mathrm{ks}_C\right\rangle &= \left\langle \mathrm{ks}_C P,H(C)\mathrm{msk}\right\rangle\\ \left\langle \mathrm{kp}_{s_i},\mathrm{ks}_{s_i}\right\rangle &= \left\langle \mathrm{ks}_{s_i} P,H(s_i)\mathrm{msk}\right\rangle\end{aligned}\tag{1}$$

计算得到的会话密钥如式(2)所示。

$$\begin{aligned}K_{0,i} &= \mathrm{ks}_{s_0}\mathrm{kp}_{s_i}=\mathrm{ks}_{s_0}\mathrm{ks}_{s_i}P=\mathrm{ks}_{s_i}\mathrm{kp}_{s_0}\\ K_i &= \mathrm{ks}_C\mathrm{kp}_{s_i}=\mathrm{ks}_C\mathrm{ks}_{s_i}P=\mathrm{ks}_{s_i}\mathrm{kp}_C\end{aligned}\tag{2}$$

密钥协商过程在控制器与交换节点之间的加密信道下进行，其中，$K_{0,i}$ 和 K_i 分别用于 SDN-ZTM

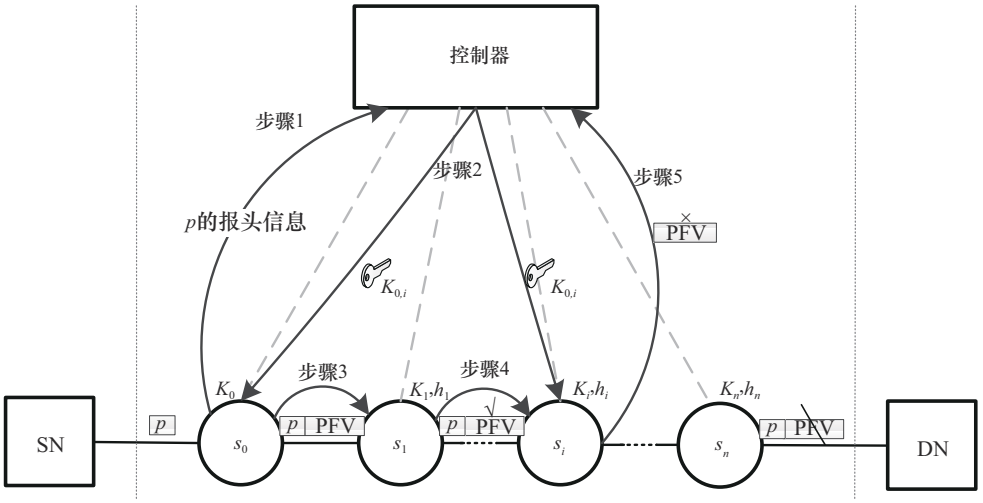


图3 SDN-ZTM 架构

对数据包的数据完整性验证和转发路径一致性验证过程。

3.3　PFV报头结构

SDN-ZTM仅考虑边缘节点间数据流基于TCP/IP协议传输的情况，SDN-ZTM通过在IP报头后面添加PFV报头，实现对数据包的转发验证和恶意交换节点的定位功能。当边缘节点间数据通过其他协议传输时，则需根据协议内容对SDN-ZTM以及PFV报头进行相应的调整，本文不对该过程展开讨论。PFV报头结构如图4所示，各字段的含义如下。

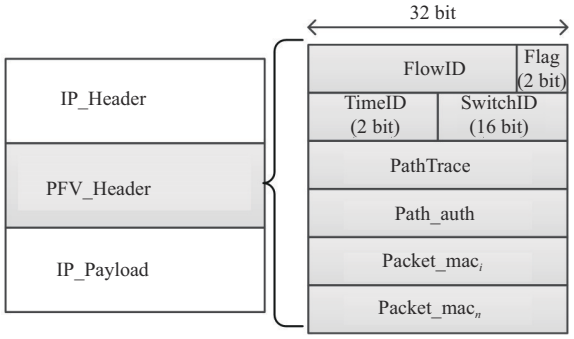


图4　PFV报头结构

FlowID：用于描述数据流类型。

TimeID：由控制器生成的时间间隔信息，用于将数据流按时间间隔进行分割。

SwitchID：标记需要对数据包进行数据完整性验证的交换节点信息。

Flag：标记数据包的状态，长度为2 bit，首位为“0”时表示数据包通过验证；第二位表示异常数据包类型，“0”表示转发路径异常，“1”表示数据完整性异常。

PathTrace：描述数据包的实际转发路径。

Path_auth：s_i对数据包添加的认证信息，标记数据包由s_i转发。

Packet_mac：用于验证数据完整性的消息认证码，Packet_mac$_i$和Packet_mac$_n$分别对应s_i和s_n处验证数据完整性的消息认证码。

PFV在入口交换节点s_0处进行生成并添加到数据包报头中，其中，FlowID的计算式为

$$\text{FlowID} = H_1(\text{srcIP}\|\text{dstIP}\|\text{padding}(\text{optional})) \quad (3)$$

其中，srcIP和dstIP分别表示IP报头中的源IP地址和目的IP地址，padding表示IP报头中的填充字段，PFV允许数据包在填充字段对数据包业务类型进行定义，若没有填充字段则忽略。

PFV中TimeID由控制器生成并下发至s_0。控制器将时间t随机分割成多个连续的时间段t_i，即$t = t_1 + t_2 + \cdots + t_n$，并通过式(4)计算得到TimeID，控制器会在上一时间段t_{i-1}结束时对s_0发送新的TimeID。

$$\text{TimeID} = H_2(t_i) \quad (4)$$

PFV中的初始SwitchID指向s_n，s_0根据控制器的指示对SwitchID进行更新，SwitchID的计算式为

$$\text{SwitchID} = H_3(s_i) \quad (5)$$

当数据流首次流入SDN-ZTM时，s_0只需在PFV中添加Packet_mac$_n$字段，如式(6)所示。

$$\text{Packet\_mac}_n = \text{MAC}_{K(s_0,s_n)}(\text{FlowID}\|\text{TimeID}\|\text{SwitchID}\|\text{payload}) \quad (6)$$

其中，payload为数据报文的有效负载，不对Packet_mac$_i$字段进行填充。

当Packet_mac$_n$验证失败时，SDN-ZTM会更新PFV中的t_i，s_0在新t_i对应的PFV中添加Packet_mac$_i$字段，如式(7)所示。

$$\text{Packet\_mac}_i = \text{MAC}_{K(s_0,s_i)}(\text{FlowID}\|\text{TimeID}\|\text{srcIP}\|\text{dstIP}\|\text{payload}\|\text{Packet\_mac}_n) \quad (7)$$

在式(6)的基础上添加Packet_mac$_n$作为计算Packet_mac$_i$的参数，能有效防止s_i上游交换节点对Packet_mac$_n$的篡改，保证SDN-ZTM对恶意交换节点定位的准确性。

PFV中Flag的初始值设置为“10”，PathTrace的初始值设为FlowID，在PFV初始阶段不对Path_auth进行填充。

3.4　基于PFV的数据包转发验证

SDN-ZTM中交换节点基于PFV对数据包进行转发验证，s_i对接收到的数据包p的转发验证过程如算法1所示。其中，$\text{count}_{\text{FlowID}}^{\text{Suc}}$和$\text{count}_{\text{FlowID}}^{\text{Fal}}$分别表示通过验证和未通过验证的数据包数量，count()函数表示对数据包进行计数，delete(p.PFV)表示将PFV字段进行去除，forward(p)和drop(p)分别表示对p进行转发和丢弃操作。

算法1　交换节点对数据包的转发验证

输入　数据包p

输出　对数据包p的处理动作

1) Function Packet forwarding verification in s_i

2) PathTrace=MAC$_{K_i}$(p. PFV. PathTrace‖ p. IP. TTL)

3) if PathTrace= h_i:

4) p.PFV.PathTrace=p.PFV.Path_auth=h_i
5) if p.PFV.SwitchID= $H_3(s_i)$&& p.PFV.SwitchID≠$H_3(s_n)$:
6) Packet_mac = $\mathrm{MAC}_{K(s_0,s_i)}$(FlowID‖TimeID‖SwitchID‖payload‖p.PFV.Packet_mac$_n$)
7) if Packet_mac= p.PFV.Packet_mac$_i$:
8) count ($\mathrm{count}^{\mathrm{Suc}}_{\mathrm{FlowID}}$) and forward ($p$)
9) else:
10) p.PFV.Flag=“01” and go on to line 24
11) end if
12) else if $s_i$is export:
13) Packet_mac = $\mathrm{MAC}_{K(s_0,s_n)}$(FlowID||TimeID|| SwitchID||payload)
14) if Packet_mac= p.PFV.Packet_mac$_n$:
15) count ($\mathrm{count}^{\mathrm{Suc}}_{\mathrm{FlowID}}$), delete($p$. PFV) and forward($p$)
16) else:
17) p.PFV.Flag=“01” and go on to line 24
18) end if
19) else:
20) count ($\mathrm{count}^{\mathrm{Suc}}_{\mathrm{FlowID}}$) and forward($p$)
21) end if
22) else:
23) p.PFV.Flag=“00”
24) send p.PFV to Controller
25) count ($\mathrm{count}^{\mathrm{Fal}}_{\mathrm{FlowID}}$) and drop($p$)
26) end if
27) end Function

首先s_i对p进行转发路径一致性的验证。控制器为$\mathrm{PATH}_{\mathrm{FlowID}}$中的$s_1$~$s_n$计算并下发$h_i$，如式(8)所示。

$$\begin{aligned} h_1 &= \mathrm{MAC}_{K_1}(\mathrm{FlowID}\|\mathrm{TTL}_1)\\ h_2 &= \mathrm{MAC}_{K_2}(h_1\|\mathrm{TTL}_2)\\ &\vdots\\ h_i &= \mathrm{MAC}_{K_i}(h_{i-1}\|\mathrm{TTL}_i) \end{aligned} \tag{8}$$

其中，TTL_i表示当数据包流经s_i时IP报头中的TTL数值。

当s_i接收到数据包时，会根据式(9)计算PathTrace并与h_i进行比较，当数据包通过转发路径一致性验证时，s_i使用h_i对PFV中的Path_auth和PathTrace字段进行更新（2~4行）。

$$\mathrm{PathTrace} = \mathrm{MAC}_{K_i}(\mathrm{PathTrace}\|\mathrm{TTL}_i) \tag{9}$$

接下来s_i对p进行数据完整性验证。根据PFV中SwitchID的取值，仅当s_i的哈希值与SwitchID相等时（5~11行）或s_i为出口交换节点时，s_i需要对p进行数据完整性验证（12~18行），即分别通过式(6)和式(7)计算消息认证码并对PFV中的Packet_mac$_i$和Packet_mac$_n$进行验证。

当数据包未通过转发验证时，s_i对$\mathrm{count}^{\mathrm{Fal}}_{\mathrm{FlowID}}$进行计数并在Flag字段中记录数据包的异常类型，$s_i$不对异常数据包$p$进行转发并将$p$的PFV报头发送至控制器（10、17、23~25行）。

当p通过转发路径一致性验证和数据完整性验证后，s_i对$\mathrm{count}^{\mathrm{Suc}}_{\mathrm{FlowID}}$进行计数并对$p$进行转发。特别地，当$s_i$为出口交换节点时，$s_i$在转发$p$前将$p$中的PFV报头删除（8、15行）。

3.5 恶意交换节点定位

SDN-ZTM基于异常的PFV字段以及各交换节点的$\mathrm{count}^{\mathrm{Suc}}_{\mathrm{FlowID}}$和$\mathrm{count}^{\mathrm{Fal}}_{\mathrm{FlowID}}$数值实现对恶意交换节点的定位，以防止其对网络进行持续的安全威胁。

数据包转发路径异常。当控制器收到s_i发送的异常PFV且Flag字段为“00”时，表明数据包未通过转发路径一致性验证，即s_i的上游交换节点出现违背转发路径规则的恶意行为（包括对转发路径的篡改、绕路和跳转）。控制器将异常PFV的Path_auth与式(8)中的h进行比对，当h_j和Path_auth相等时，h_j对应的交换节点s_j即恶意交换节点。

数据包篡改。当控制器收到异常PFV且Flag字段为“01”时，表明数据包未通过数据完整性验证，即恶意交换节点对数据包进行数据篡改攻击。定义区间（low,high），其中low和high为$\mathrm{PATH}_{\mathrm{FlowID}}$中$s_i$编号，（low,high）表明恶意交换节点存在于$s_{\mathrm{low}}$~$s_{\mathrm{high}}$，（low,high）的初始值设定为（1,$n$–1）；定义num_$s$为发送异常PFV的交换节点编号，num_PFV为PFV中SwitchID字段对应的交换节点编号，篡改数据对恶意交换节点定位的过程如算法2所示。SDN-ZTM通过二分法对恶意交换节点进行定位，当区间内只包含2个交换节点时，可以定位到恶意交换节点（2~7行）；当区间内包含超过2个交换节点时，SDN-ZTM在每次收到异常PFV时执行一次二分法的查找过程，返回更新后的（low,high）和num_PFV（9~19行），同时控制器

更新t_i并将计算得到的TimeID和SwitchID发送至s_0、$s_{\text{num_PFV}}$和s_n。

算法2　篡改数据对恶意交换节点定位

输入　num_s，num_PFV，（low,high）

输出　异常交换节点s_i或需进行更新的验证信息

1) Function Locating malicious switching nodes that tamper with data
2) if high =low+1:
3) if num_s > high:
4) return ($s_{\text{high}-1}$)
5) else:
6) return ($s_{\text{low}-1}$)
7) end if
8) else：
9) if num_s=num_PFV= s_n:
10) num_PFV $= \left\lceil \dfrac{\text{low} + \text{high}}{2} \right\rceil$
11) return(num_PFV)
12) end if
13) if num_s > high:
14) low=num_s, num_PFV $= \left\lceil \dfrac{\text{num\_}s + \text{high}}{2} \right\rceil$
15) return((low,high), num_PFV)
16) else:
17) high=num_s, num_PFV $= \left\lceil \dfrac{\text{low} + \text{num\_}s}{2} \right\rceil$
18) return((low,high), num_PFV)
19) end if
20) end if
21) end Function

恶意丢弃。定义T为SDN-ZTM对恶意丢弃的检测时间间隔，SDN-ZTM在每个T内获取$\text{PATH}_{\text{FlowID}}$中各$s_i$的$\text{count}^{\text{Suc}}_{\text{FlowID}}$和$\text{count}^{\text{Fal}}_{\text{FlowID}}$，并用$\text{count}^{\text{Suc}}_{\text{FlowID},i}$和$\text{count}^{\text{Fal}}_{\text{FlowID},i}$表示，为保证统计数值不受未通过转发验证而丢弃的数据包影响，对$\text{count}^{\text{Suc}}_{\text{FlowID},i}$和$\text{count}^{\text{Fal}}_{\text{FlowID},i}$进行整合生成count[ ]={$c_0,c_1,\cdots,c_n$}，其中，$c_i$表示交换机$s_i$成功接收到的数据包数量与上游交换节点丢弃异常数据包数量的总和，计算式为

$$c_i = \text{count}^{\text{Suc}}_{\text{FlowID},i} + \sum_{j=1}^{i} \text{count}^{\text{Fal}}_{\text{FlowID},j} \tag{10}$$

数据包在传输过程中会因网络拥塞、传输错误等原因存在一定的丢包率，为防止自然情况下丢包对恶意交换节点的误判，引入恶意丢弃阈值ϕ，当$c_{i+1} < c_i(1-\phi)$时，则认为交换设备s_i为对数据包恶意丢弃的恶意交换节点。当检测到恶意丢弃行为或检测时间达到T时，控制器通知各交换节点重置计数器统计信息。特别地，第2.1节描述的数据包欺骗攻击行为等同于数据包被恶意交换节点丢弃，SDN-ZTM能对执行该行为的恶意交换节点进行有效定位。

4　分析与讨论

本节从消息认证码安全、基于二分法定位恶意交换节点、恶意丢弃阈值、计算复杂度、报头和通信开销、安全功能等方面对SDN-ZTM进行理论分析与讨论。

4.1　消息认证码安全

SDN-ZTM中交换节点通过对MAC的数据完整性验证实现对数据包的转发验证功能，因此，消息认证码的不可伪造性是保证转发验证安全性的基础。SDN-ZTM对PFV中的PathTrace、Packet_mac_i和Packet_mac_n字段设置长度为32 bit的消息认证码，若恶意交换节点想通过伪造消息验证码以隐藏其恶意行为不被发现，成功概率为$\dfrac{1}{2^{32}}$，该概率足够小，因此，异常数据包和恶意交换节点无法躲避SDN-ZTM的检测。

4.2　基于二分法定位恶意交换节点

SDN-ZTM使用二分法对数据篡改攻击的恶意交换节点进行异常定位，产生更少且稳定的验证次数，完成每次定位进行的验证次数最多为log(l)。

假设在长度为l的转发路径中存在恶意交换节点，为对恶意交换节点进行定位，每次随机选取交换节点进行验证，定义$f(X)$为定位到恶意交换节点时所需的最大验证次数，如式(11)所示。

$$f(X) = \max_{1 \le i \le l} (f(X|k=i) + 1) \tag{11}$$

其中，$f(X|k=i)$表示选取编号为i的交换节点进行验证时需要的最大验证次数，可以表示为

$$f(X|k=i) = \max(f(X[1,i-1]), f(X[i=1,l])) \tag{12}$$

其中，$X[1,i-1]$表示在编号1至编号i–1交换节点间定位恶意交换节点需要的验证次数，为使$f(X)$取值最小，式(12)中i应取值为$\left\lceil \dfrac{l}{2} \right\rceil$，此时$f(X)$为

$$f(X)=1+\frac{1}{2}f\left(\left[\frac{l}{2}\right]\right) \tag{13}$$

由数学归纳法可得$f(X)$的最大值为$\log(l)$，因此，当SDN-ZTM使用二分法对恶意交换节点定位时，定位过程所需的最大验证次数最少，为$\log(l)$，即恶意交换节点只要在检测时间间隔内对数据包篡改次数达到$\log(l)$次，SDN-ZTM即可实现对该恶意交换节点的有效定位。

4.3 恶意丢弃阈值

SDN-ZTM通过恶意丢弃阈值ϕ对丢弃数据包的恶意交换节点进行检测与定位，ϕ的取值决定检测与定位的准确性，现对ϕ的取值与漏报率（FNR, false negative rate）之间的关系进行分析。

假设网络中数据包在2个交换节点之间的平均丢失率为α，用α'表示数据包在交换节点s_i和s_{i-1}间的丢包率，当$\alpha'>\phi$时，根据式(14)判定交换节点s_i为对数据包进行恶意丢弃的恶意交换节点。

$$c_i=(1-\alpha')(1-\alpha)^{i-1}c_0<(1-\phi)(1-\alpha)^{i-1}c_0 \tag{14}$$

其中，c_0和c_i表示s_0和s_i接收到的数据包数量，i为s_i在$\mathrm{PATH}_{\mathrm{FlowID}}=(s_0,s_1,\cdots,s_n)$上的编号。

本文希望对恶意交换节点的检测漏报率FNR $\leqslant\lambda$，则可以得到

$$\begin{aligned}&\mathrm{FNR}=P[c_i<(1-\phi)(1-\alpha)^{i-1}c_0]=\\&P\left[c_i<\frac{(1-\phi)}{(1-\alpha)}(1-\alpha)^ic_0\right]=\\&P\left[c_i<\left(1-\frac{\phi-\alpha}{1-\alpha}\right)(1-\alpha)^ic_0\right]\leqslant\lambda\end{aligned} \tag{15}$$

基于切诺夫界，由式(15)可得到

$$\mathrm{FNR}=\mathrm{e}^{-\frac{\left(\frac{\phi-\alpha}{1-\alpha}\right)^2(1-\alpha)^ic_0}{2}}\leqslant\lambda \tag{16}$$

化简后得到ϕ与λ的关系如式(17)所示，ϕ与误报率（FPR, false positive rate）的关系可得到相同的结果。

$$\phi=\alpha+\sqrt{\frac{2\ln\left(\frac{1}{\lambda}\right)}{(1-\alpha)^{i+2}c_0}} \tag{17}$$

从式(17)中可以得出ϕ与λ呈负相关，若要得到更小的漏报率与误报率，SDN-ZTM需设定更高的ϕ。当c_0足够大时，ϕ的取值趋近于α且与λ无关，因此当SDN-ZTM选取固定的ϕ后，可以通过提高检测时间增加c_0，从而有效降低检测的漏报率和误报率。

4.4 计算复杂度

SDN-ZTM首次接收到数据流时，控制器与转发路径中每个交换节点之间进行一次椭圆曲线上的乘法运算用于会话密钥的生成，并通过$n-1$次消息认证码的计算生成路径一致性认证信息。此外，入口交换节点会根据控制器下发的信息进行一次椭圆曲线上的乘法运算，用于与某一下游交换节点间会话密钥的生成。

当SDN-ZTM对数据包进行转发时，入口交换节点需要进行1或2次消息认证码的计算，分别用于PFV字段中$\mathrm{Packet_mac}_i$和$\mathrm{Packet_mac}_n$字段的生成；中间交换节点和出口交换节点分别需要进行1或2次和2次消息认证码的计算，分别用于Path-Trace的更新和$\mathrm{Packet_mac}_i$的验证。当SDN-ZTM未检测到数据篡改时，入口交换节点和中间交换节点不需要进行$\mathrm{Packet_mac}_i$的生成与验证过程。将SDN-ZTM中交换节点对数据包转发的计算复杂度与相关方案进行比较，如表3所示，其中，M表示一次消息认证码的运算，sca表示在椭圆曲线上的一次标量乘法计算，P表示一次双线性对运算，l表示转发路径的长度。经测试，平均一次M和P的基础运算时间约为10 μs和0.5 ms，从表3可以得出SDN-ZTM在交换节点端引入更低的计算复杂度，可有效降低对数据包转发过程的影响。

表3 交换节点计算复杂度

方案	源终端/入口交换节点	中间交换节点	目的终端/出口交换节点
OPT[29]	$2lM$	$2M$	$(l+1)M$
SDNsec[30]	$2lM$	$2M$	$2M$
P4Label[31]	$3P$	—	$3P$
SDN-ZTM	$M/2M$	$M/2M$	$2M$

SDN-ZTM对转发路径异常、数据篡改和恶意丢弃的恶意交换节点的定位过程的计算复杂度分别为$O(1)$、$O(\log(l))$和$O(l)$。

4.5 报头和通信开销

SDN-ZTM通过对数据包添加PFV报头实现其功能，PFV报头长度为24 B。将SDN-ZTM的PFV报头开销率与相关方案进行比较，假设数据包负载为600 B，得到的额外报头开销率（额外报头与数据包负载的比值）如图5所示。

从图5中可以看出，SDN-ZTM产生的报头开销为4%且不随路径长度l的变化而改变，OPT和

SDNsec方案增加的额外报头开销与l的关系分别为(52+16l) B和(22+8l) B，P4Label方案为每个数据包添加268 B的定长额外报头。相较于其他方案，SDN-ZTM产生的额外报头开销率更小，具有更高的数据包转发效率。

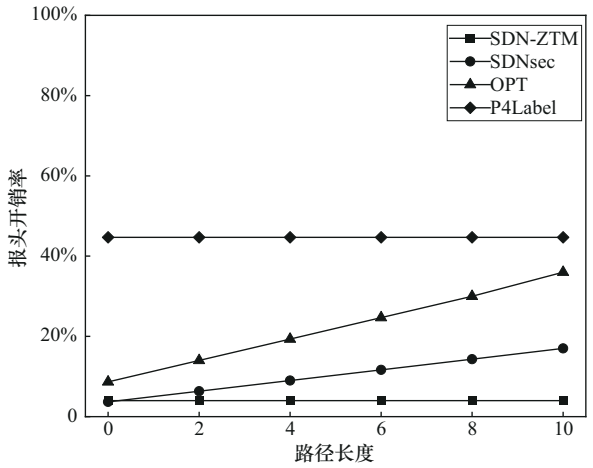


图5　额外报头开销率

此外，SDN-ZTM中控制器与交换节点间仅在初始阶段或链路中出现异常时进行通信，并且当链路中出现异常时，SDN-ZTM会及时对恶意交换节点进行定位，从而减少链路异常的可能。因此，SDN-ZTM在执行过程中控制器与交换节点间的通信次数较少，产生的通信开销较小。

4.6　安全功能

SDN-ZTM能有效检测并防御第2节中的数据包篡改、数据包欺骗、数据包转发路径异常、恶意丢弃数据包等安全威胁，并能对转发路径中的恶意交换节点进行有效的定位。同时，SDN-ZTM能实现对数据包的源认证和抗重放攻击。SDN-ZTM中的PFV报头由入口交换节点s_0与其他交换节点s_i的会话密钥$K(s_0,s_i)$进行初始化，来源不可信的数据包因为不具备$K(s_0,s_i)$，无法对PFV报头进行有效构建，因此SDN-ZTM能实现源认证。SDN-ZTM通过对时间分段并使用TimeID在PFV中进行标记，当攻击者对数据包进行重放攻击时，数据包无法通过交换节点对Packet_mac$_i$和Packet_mac$_n$字段进行验证，因此，SDN-ZTM能对重放攻击进行有效防御。

表4将SDN-ZTM与相关对数据传输路径安全验证的方案的安全功能进行对比，其中，√表示具备该功能，×表示不具备该功能。从表4可以看出，相较于其他方案，SDN-ZTM具备更全面的安全功能。

表4　安全功能对比

方案	源认证	防数据篡改	防恶意丢弃	路径一致性	防重放攻击	异常定位
文献[23]	×	×	√	√	×	√
文献[29]	√	√	×	√	√	×
文献[30]	×	×	×	√	√	×
文献[31]	√	√	×	×	×	×
文献[33]	×	√	×	√	×	√
SDN-ZTM	√	√	√	√	√	√

5　仿真与评估

通过构建仿真环境，在模拟网络中验证SDN-ZTM的有效性，并通过仿真实验对SDN-ZTM的性能进行评估。

5.1　仿真环境

SDN-ZTM在P4可编程软件交换机中进行实现，使用P4语言编写并通过P4c（p4 compiler）生成JSON格式描述文件，并将其导入P4行为模型（BMv2, behavioral-model version 2）中运行。控制器通过P4 Runtime接口对数据平面进行控制。

SDN-ZTM通过ECPy（elliptic curve python）库实现会话密钥的生成过程，并基于哈希函数的消息认证码（HMAC, hash-based message authentication code）进行消息认证码的生成，其中哈希算法选取MD5（message-digest algorithm 5），SDN-ZTM对HMAC得到的128 bit消息认证码取前32 bit，以符合PFV结构。

构建实验对SDN-ZTM的有效性以及方案性能进行测试，实验选取分布式拓扑结构，使用Mininet对模拟网络环境进行构建，由30台P4软件交换机和若干台虚拟主机组成，分别模拟物联网中的交换节点和边缘节点。实验平台设置为Intel i7-11370H 4.266 GHz，32 GB内存的主机，操作系统为Ubuntu 14.06。

5.2　有效性分析

在模拟网络中选取一条长度为10的传输路径PATH = (s_0,s_1,⋯,s_9)，s_i为传输路径中的交换机，规定数据包由源主机经过PATH发送到目的主机。令s_5为恶意交换节点，分别讨论当s_5对数据包进行如第2.1节所示的攻击行为时，SDN-ZTM能否有效实现对攻击行为的检测与恶意交换节点的定位，并使用漏报率和误报率对SDN-ZTM的有效性进行评估。根据SDN-ZTM的工作原理可知，数据包欺骗攻击等同于数据包在该恶意交换节点上被丢弃，与

恶意丢弃攻击相同，因此将第2.1节所提攻击行为分为数据包篡改攻击、转发路径异常攻击和恶意丢弃攻击3种类型。

定义恶意交换节点在随机时间间隔T_i内对数据包进行的恶意攻击为一次攻击事件，其余事件则为非攻击事件，当控制器检测到恶意交换节点发出的异常数据包并对恶意交换节点成功定位时，认为是一次有效的防御。定义在规定时间内，恶意交换节点执行的攻击事件和非攻击事件次数分别为N和M，SDN-ZTM未检测到攻击的次数为FN_1，SDN-ZTM检测到攻击但未正确定位恶意交换节点的次数为FN_2，SDN-ZTM检测到攻击事件但将正常交换节点定义为恶意交换节点的次数为FP_1，恶意交换节点未攻击时SDN-ZTM检测并给出异常定位的次数为FP_2，分别得到漏报率$\mathrm{FNR}=\dfrac{\mathrm{FN}_1+\mathrm{FN}_2}{N}$，误报率$\mathrm{FPR}=\dfrac{\mathrm{FP}_1+\mathrm{FP}_2}{M}$。

实验1 源主机通过iperf向目的主机发送数据，持续1 000 s，s_5分别以0.5%、0.75%、1%、2%和4%的恶意攻击概率对数据包进行攻击，攻击的时间间隔为$T_i(100\,\mathrm{ms}\leqslant T_i\leqslant 200\,\mathrm{ms})$，攻击方式包括数据包篡改攻击、转发路径异常攻击和恶意丢弃攻击，其中转发路径异常攻击包括对传输路径进行篡改、跳转和绕路3种。网络中数据包的平均丢失率为$\alpha\approx 0.5\%$，设置SDN-ZTM恶意丢弃阈值$\phi=0.5\%$，SDN-ZTM对恶意攻击检测的时间间隔T=100 ms，控制器对检测结果与异常定位情况进行统计，得到的结果如图6和图7所示。

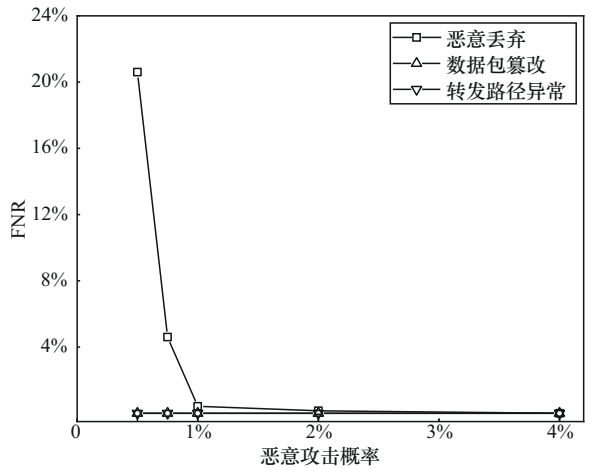


图6 对不同恶意攻击的检测漏报率

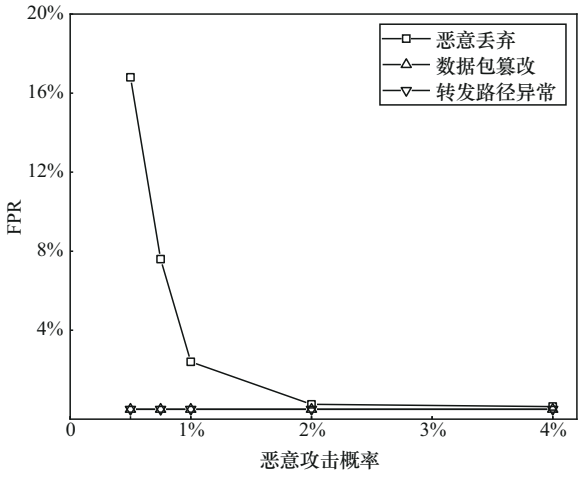


图7 对不同恶意攻击的检测误报率

从图6和图7中可以看出，SDN-ZTM对恶意交换节点发出的数据包篡改攻击和转发路径异常攻击的检测与异常定位的漏报率和误报率都为0，因为SDN-ZTM能对恶意交换节点发出的每个异常数据包进行转发验证，与4.1节分析的异常数据包躲避SDN-ZTM检测的概率小于$\dfrac{1}{2^{32}}$相一致。并且检测效果与设备的恶意攻击概率无关，因为当恶意交换节点发起任何转发路径异常攻击时，SDN-ZTM即可完成对其的检测与定位过程；当SDN-ZTM对恶意交换节点发起的数据包篡改攻击进行定位时，至多需要4次（$\lceil \mathrm{lb}\,10\rceil=4$）对该异常行为的检测即可完成定位过程。当恶意交换节点以较小的概率对数据包发起恶意丢弃攻击时，SDN-ZTM产生较高的漏报率和误报率，当恶意攻击概率为0.5%时，漏报率和误报率分别为20.6%和16.8%，由于恶意攻击概率较低，SDN-ZTM在检测时间间隔内统计到的丢弃数据包数量较少，难以超过恶意丢弃阈值ϕ引起漏报；同时少量的恶意丢弃行为和数据包自然丢弃情况较难区分，会产生误报。少量被恶意丢弃的数据包不会对目的主机造成攻击行为，本文认为进行恶意丢弃行为的恶意交换节点在被有效定位前产生的危害是较小的。随着恶意丢弃概率的增加，漏报率和误报率明显减少，当恶意丢弃概率为2%时，漏报率和误报率分别降至0.15%和0.25%。

实验2 考虑到现实情况中恶意交换节点会对数据包同时进行多种攻击，将实验1中的攻击方式进行组合，构建实验2对SDN-ZTM检测组合攻击的能力进行测试。s_5分别以0.5%、0.75%、1%、

2%和4%的恶意攻击概率对数据包进行组合攻击，攻击方式包含数据包篡改攻击、转发路径异常攻击和恶意丢弃攻击，SDN-ZTM的参数设置与实验1相同，控制器对检测结果与异常定位情况进行统计，得到的结果如图8所示。

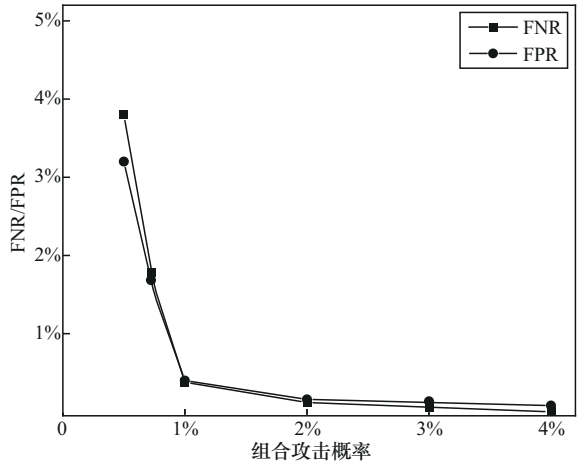


图8　组合攻击的检测效果

从图8中可以看出，与实验1相比，SDN-ZTM对组合攻击的恶意交换节点具有更低的漏报率和误报率。当组合攻击概率较低时，漏报率和误报率的下降幅度十分明显，当组合攻击概率为0.5%时，漏报率和误报率分别为3.8%和3.2%，与单一的恶意丢弃攻击相比分别降低81.56%和80.95%，对于其他组合攻击概率，漏报率和误报率也有明显的降低，当组合攻击概率为1%和2%时，漏报率和误报率分别为0.38%、0.12%和0.4%、0.16%。因为SDN-ZTM能对篡改过的数据包与转发路径异常的数据包进行实时发现并对实施组合攻击的恶意交换节点进行定位，在对恶意丢弃攻击检测的时间间隔T内即完成对恶意交换节点的检测与异常定位过程，减少对恶意丢弃检测过程产生的漏报率与误报率，提升对恶意交换节点的检测与定位能力。

实验3　验证检测时间间隔T与SDN-ZTM对恶意交换节点的检测与定位的关系，在实验2的基础上，分别考虑T=100 ms和T=200 ms的2种情况，对SDN-ZTM的检测结果与异常定位情况进行统计，得到的结果如图9所示。

从图9中可以看出，当组合攻击概率为0.5%、T为200 ms时，漏报率和误报率分别为1.86%和2.02%，较T为100 ms时分别降低51.05%和36.86%。当选取其他组合攻击概率时得到相同结果，即随着检测时间间隔T的提升，SDN-ZTM的漏报率和误报率均有明显下降。随着检测时间间隔T的提升，SDN-ZTM对恶意丢弃数据包的统计数量增加，从而降低检测的漏报率和误报率，与4.3节的结论相一致。SDN-ZTM可以通过动态调整检测时间间隔T来提升对恶意丢弃攻击的检测性能，当恶意交换节点攻击频率较低时，选取更大的T以降低检测的漏报率和误报率；当恶意交换节点攻击频率较高时，选取更小的T能对恶意交换节点进行更迅速的定位。

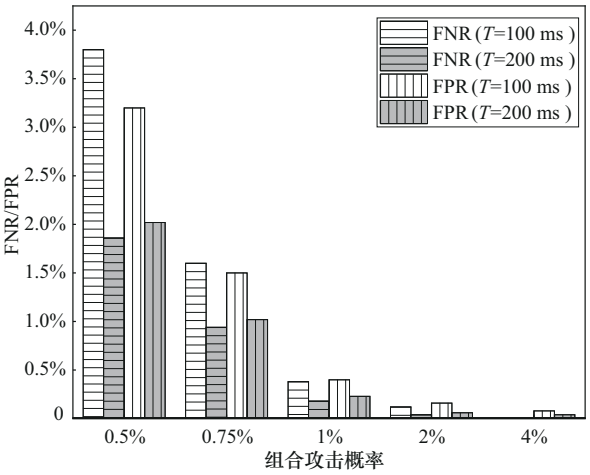


图9　检测效果与检测时间间隔T关系

从实验1~实验3的结果可以得出，SDN-ZTM能对数据包篡改、转发路径异常、恶意丢弃以及组合攻击等恶意攻击实现有效的检测，并能对恶意交换节点实现有效的定位。因此，通过将SDN-ZTM应用到边缘节点间数据传输链路的安全验证，能有效保证边缘节点间数据传输的安全性，并能通过对传输链路中异常节点的检测与定位，及时发现并阻止恶意交换节点对物联网造成进一步危害。

5.3　性能评估

本节从数据包转发时延和数据吞吐量2个方面对SDN-ZTM的网络性能进行评估，并将SDN-ZTM与未使用SDN-ZTM方案的常规网络以及同类型方案OPT[29]、SDNsec[30]和P4Label[31]进行对比。

实验4　令l表示$\mathrm{PATH_{FlowID}}$的长度，即$\mathrm{PATH_{FlowID}}$中包含l个交换机。在模拟网络中分别选取l=2,4,6,8,10的转发路径，从源主机持续向目的主机发送数据包，对数据包在SDN-ZTM架构中的平均转发时延进行测试，测试结果如图10~图12所示。

图10为当l=6时前100个数据包在SDN-ZTM

与常规网络中的平均转发时延。从图10中可以看出，在常规网络中，前100个数据包的平均转发时延为5.37 ms；在SDN-ZTM中，首个数据包的平均转发时延为12.52 ms，平均转发时延随着数据包数量的增加而降低并逐渐趋于稳定，前100个数据包的平均转发时延为5.87 ms。首个数据包较后续数据包产生的额外转发时延用于SDN-ZTM中会话密钥的生成以及转发路径一致性验证信息的下发等过程，该过程在同一FlowID的数据流中只进行一次，随着数据流中数据包的传输，得到的平均转发时延会对该过程的时间开销进行淡化，最终平均转发时延数值会趋近于非首个数据包的平均转发时延。

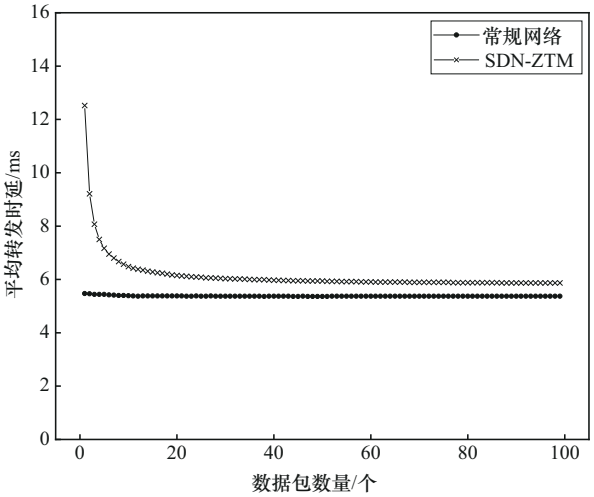


图10　数据包平均转发时延（l=6）

当l=6时，分别对1 000个数据包在SDN-ZTM与常规网络中的平均转发时延进行统计，在常规网络中，数据包平均转发时延的均值和标准差分别为5.35 ms和0.21 ms；在SDN-ZTM中，数据包转发时延的均值和标准差分别为5.83 ms和0.34 ms。图11为使用高斯函数对数据包转发时延进行拟合得到的概率密度曲线，在常规网络和SDN-ZTM中，数据包转发时延95%的置信区间分别为5.347±0.013 ms和5.826±0.021 ms，可以得出数据包的转发时延分布在SDN-ZTM中与常规网络相近，SDN-ZTM较常规网络产生的转发时延开销较低。

图12在SDN-ZTM和相关方案中对一段时间内数据包的转发时延进行测试并取平均值。从图12中可以看出，数据包的平均转发时延随转发路径长度的增加而增加，增加幅度较常规网络由低至高分别为SDN-ZTM、OPT、SDNsec和P4Label。当l=10时，SDN-ZTM、OPT、SDNsec和P4Label对数据包的平均转发时延分别为9.52 ms、9.86 ms、10.03 ms和11.46 ms，相较于常规网络中8.72 ms的平均转发时延分别增加了9.17%、13.07%、15.02%和31.42%，SDN-ZTM引入的额外转发时延较低，OPT和SDNsec次之，P4Label因为每个交换节点都需要进行双线性对运算，因此引入最高的转发时延。

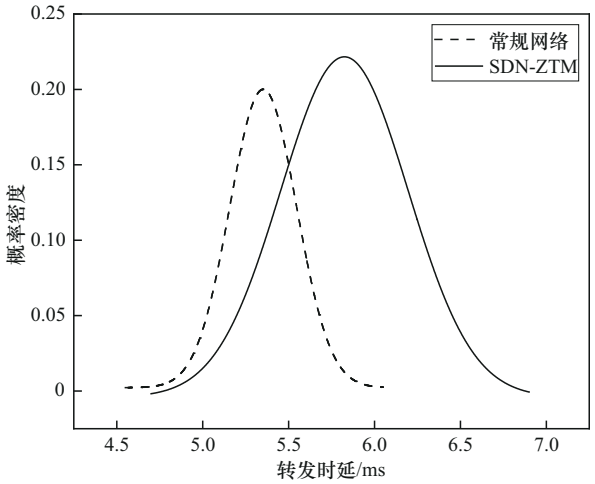


图11　数据包转发时延概率密度（l=6）

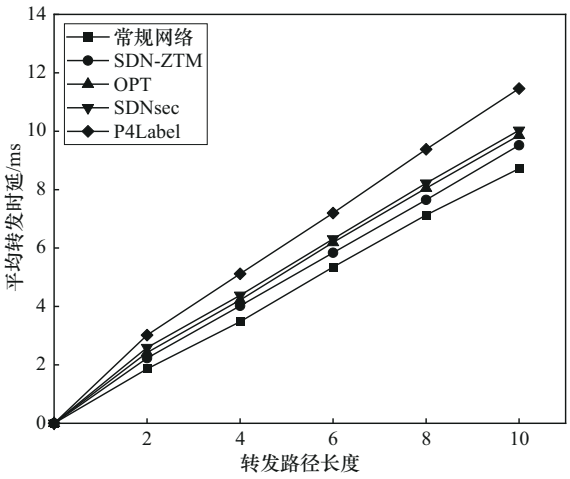


图12　当l=2,4,6,8,10时不同方案数据包平均转发时延

实验5　在模拟网络中选取l=10的转发路径，从源主机向目的主机发送数据包，数据包负载长度分别选取300 B、600 B、900 B和1 200 B，分别对SDN-ZTM和相关方案中的网络吞吐量进行测试并分析，结果如图13所示。

从图13中可以得出，网络吞吐量随数据包负载长度的增加而增加，不同方案中的吞吐量较常规网络的吞吐量都有所降低，其中SDN-ZTM的吞吐量

较常规网络吞吐量降低的最少，平均仅下降7.75%，OPT、SDNsec和P4Label的吞吐量较常规网络的吞吐量平均分别下降11.69%、12.91%和18.46%。

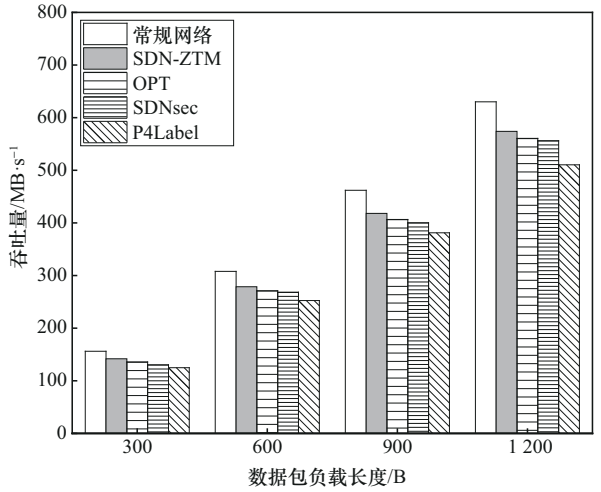


图13　数据包负载长度为300 B、600 B、900 B和1 200 B时网络吞吐量（l=10）

从实验4和实验5的结果可以得出，SDN-ZTM在实现其安全功能的同时产生更低的性能开销。与相关方案相比，当SDN-ZTM对边缘节点间数据传输链路进行安全验证时，对物联网的网络性能影响较小，更适用于物联网的实际应用环境。

6　结束语

为解决物联网边缘节点间以及边缘节点与云数据中心间数据安全传输与恶意交换节点定位等安全问题，本文提出了一种基于SDN的物联网边缘节点间数据流零信任管理方法，并基于该方法构建SDN-ZTM架构。SDN-ZTM将SDN应用到物联网边缘节点间的数据传输过程，通过交换节点对数据的转发验证实现对数据篡改、转发路径异常和恶意丢弃等异常行为的检测与阻断，同时实现对恶意交换节点的定位功能。理论分析和仿真实验表明，相较于相关方案，SDN-ZTM能实现更全面的安全功能。同时，SDN-ZTM引入固定长度的报头和较低的性能开销，是一种轻量级、适合物联网应用场景的可实用方案。

参考文献：

[1] GUO F X, YU F R, ZHANG H L, et al. Enabling massive IoT toward 6G: a comprehensive survey[J]. IEEE Internet of Things Journal, 2021, 8(15): 11891-11915.

[2] CHEN F, LUO D M, XIANG T, et al. IoT cloud security review: a case study approach using emerging consumer-oriented applications[J]. ACM Computing Surveys, 2021, 54(4): 1-36.

[3] OMETOV A, MOLUA O L, KOMAROV M, et al. A survey of security in cloud, edge, and fog computing[J]. Sensors (Basel), 2022, 22(3): 927.

[4] CAO K Y, LIU Y F, MENG G J, et al. An overview on edge computing research[J]. IEEE Access, 2020, 8: 85714-85728.

[5] COSTA B, BACHIEGA J Jr, DE CARVALHO L R, et al. Orchestration in fog computing: a comprehensive survey[J]. ACM Computing Surveys, 2023, 55(2): 1-34.

[6] XIAO Y H, JIA Y Z, LIU C C, et al. Edge computing security: state of the art and challenges[J]. Proceedings of the IEEE, 2019, 107(8): 1608-1631.

[7] ALWAKEEL A M. An overview of fog computing and edge computing security and privacy issues[J]. Sensors, 2021, 21(24): 8226.

[8] RANAWEERA P, JURCUT A D, LIYANAGE M. Survey on multi-access edge computing security and privacy[J]. IEEE Communications Surveys & Tutorials, 2021, 23(2): 1078-1124.

[9] KANG J J, FAHD K, VENKATRAMAN S, et al. Hybrid routing for man-in-the-middle (MITM) attack detection in IoT networks[C]//Proceedings of the 2019 29th International Telecommunication Networks and Applications Conference (ITNAC). Piscataway: IEEE Press, 2019: 1-6.

[10] MCKEOWN N. Software-defined networking[C]//Proceedings of the IEEE International Conference on Computer Communications. Piscataway: IEEE Press, 2009: 30-32.

[11] JAVANMARDI S, SHOJAFAR M, MOHAMMADI R, et al. An SDN perspective IoT-Fog security: a survey[J]. Computer Networks, 2023, 229: 109732.

[12] KIRAN N, PAN C Y, WANG S H, et al. Joint resource allocation and computation offloading in mobile edge computing for SDN based wirelessnetworks[J]. Journal of Communications and Networks, 2020, 22(1): 1-11.

[13] 吕军, 栾文鹏, 刘日亮, 等. 基于全面感知和软件定义的配电物联网体系架构[J]. 电网技术, 2018, 42(10): 3108-3115.
LYU J, LUAN W P, LIU R L, et al. Architecture of distribution Internet of things based on widespread sensing & software defined technology[J]. Power System Technology, 2018, 42(10): 3108-3115.

[14] DAS D, BANERJEE S, DASGUPTA K, et al. Blockchain enabled sdn framework for security management in 5g applications[C]//Proceedings of the 24th International Conference on Distributed Computing and Networking. Piscataway: IEEE Press, 2023: 414-419.

[15] BOSSHART P, DALY D, GIBB G, et al. P4: programming protocol-independent packet processors[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 87-95.

[16] SADIQ K A, THOMPSON A F, AYENI O A. Mitigating DDoS attacks in cloud network using fog and SDN: a conceptual security framework[J]. International Journal of Applied Information Systems, 2020, 12(32): 11-16.

[17] NGUYEN T G, PHAN T V, NGUYEN B T, et al. SeArch: a collaborative and intelligent NIDS architecture for SDN-based cloud IoT networks[J]. IEEE Access, 2019, 7: 107678-107694.

[18] KHAN M T, AKHUNZADA A, ZEADALLY S. Proactive defense for fog-to-things critical infrastructure[J]. IEEE Communications Magazine, 2022, 60(12): 44-49.

[19] GAO J B, OBOUR AGYEKUM K O B, SIFAH E B, et al. A blockchain-SDN-enabled Internet of vehicles environment for fog computing and 5G networks[J]. IEEE Internet of Things Journal, 2020, 7(5): 4278-4291.

[20] XIE L X, DING Y, YANG H Y, et al. Blockchain-based secure and trustworthy Internet of things in SDN-enabled 5G-VANETs[J]. IEEE Access, 2019, 7: 56656-56666.

[21] OH J, LEE J, KIM M, et al. A secure data sharing based on key aggregate searchable encryption in fog-enabled IoT environment[J]. IEEE Transactions on Network Science and Engineering, 2022, 9(6): 4468-4481.

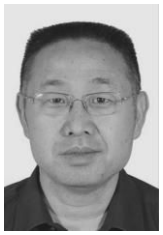
[22] TORRES-CHARLES C A, CARRIZALES-ESPINOZA D E, SANCHEZ-

GALLEGOS D D, et al. SecMesh: an efficient information security method for stream processing in edge-fog-cloud[C]//Proceedings of the 2022 7th International Conference on Cloud Computing and Internet of Things. New York: ACM Press, 2022: 8-16.
[23] MOHAN K V M, KODATI S, KRISHNA V. Securing SDN enabled IoT scenario infrastructure of fog networks from attacks[C]//Proceedings of the 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS). Piscataway: IEEE Press, 2022: 1239-1243.
[24] HU C L, HSU C Y, SUNG W M. FitPath: QoS-based path selection with fittingness measure in integrated edge computing and software-defined networks[J]. IEEE Access, 2022, 10: 45576-45593.
[25] ABSARDI Z N, JAVIDAN R. A QoE-driven SDN traffic management for IoT-enabled surveillance systems using deep learning based on edge cloud computing[J]. The Journal of Supercomputing, 2023, 79 (17): 19168-19193.
[26] LIU B Y, XU G A, XU G S, et al. Deep reinforcement learning-based intelligent security forwarding strategy for VANET[J]. Sensors, 2023, 23(3): 1204.
[27] ZHANG P, XU S M, YANG Z R, et al. FOCES: detecting forwarding anomalies in software defined networks[C]//Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS). Piscataway: IEEE Press, 2018: 830-840.
[28] HESSAM G, SABA G, ALKHAYAT M I. A new approach for detecting violation of data plane integrity in software defined networks[J]. Journal of Computer Security, 2021, 29(3): 341-358.
[29] KIM T H J, BASESCU C, JIA L M, et al. Lightweight source authentication and path validation[C]//Proceedings of the 2014 ACM Conference on SIGCOMM. New York: ACM Press, 2014: 271-282.
[30] SASAKI T, PAPPAS C, LEE T, et al. SDNsec: forwarding accountability for the SDN data plane[C]//Proceedings of the 2016 25th International Conference on Computer Communication and Networks (ICCCN). Piscataway: IEEE Press, 2016: 1-10.
[31] ZUO Z B, CHANG C W, ZHANG Y, et al. P4Label: packet forwarding control mechanism based on P4 for software-defined networking[J]. Journal of Ambient Intelligence and Humanized Computing, 2020, doi: 10.1007/s12652-020-01719-3.
[32] 吴平, 常朝稳, 左志斌, 等. 基于地址重载的SDN分组转发验证[J]. 通信学报, 2022, 43(3): 88-100.
WU P, CHANG C W, ZUO Z B, et al. Address overloading-based packet forwarding verification in SDN[J]. Journal on Communications, 2022, 43(3): 88-100.
[33] REN Q, HU T, WU J X, et al. Multipath resilient routing for endogenous secure software defined networks[J]. Computer Networks, 2021, 194: 108134.
[34] BELLARE M, KILIAN J, ROGAWAY P. The security of the cipher block chaining message authentication code[J]. Journal of Computer and System Sciences, 2000, 61(3): 362-399.
[35] SCOTT-HAYWARD S. Design and deployment of secure, robust, and resilient SDN controllers[C]//Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft). Piscataway: IEEE Press, 2015: 1-5.
[36] MIDHA S, TRIPTAHI K. Extended TLS security and defensive algorithm in OpenFlow SDN[C]//Proceedings of the 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence). Piscataway: IEEE Press, 2019: 141-146.
[37] MAHALLE P N, ANGGOROJATI B, PRASAD N R, et al. Identity authentication and capability based access control (IACAC) for the Internet of things[J]. Journal of Cyber Security and Mobility, 2013, 1(4): 309-348.
[38] MIELE A, LENSTRA A K. Efficient ephemeral elliptic curve cryptographic keys[C]//International Conference on Information Security. Berlin: Springer, 2015: 524-547.

[作者简介]

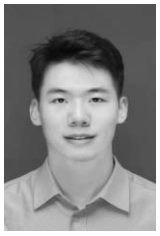
肖警续（1994– ），男，吉林长春人，信息工程大学博士生，主要研究方向为网络安全、SDN安全等。

郭渊博（1975– ），男，陕西周至人，博士，信息工程大学教授、博士生导师，主要研究方向为大数据安全、态势感知。



常朝稳（1966– ），男，河南滑县人，博士，信息工程大学教授、博士生导师，主要研究方向为移动信息安全、物联网安全等。

吴平（1979– ），男，安徽宿松人，信息工程大学博士生，主要研究方向为SDN安全、网络安全、数据平面编程。



杨晨立（1997– ），男，河南郑州人，信息工程大学硕士生，主要研究方向为网络安全防御。