一种基于数据平面可编程的软件定义网络报文转发验证机制

左志斌 常朝稳* 祝现威
(信息工程大学 郑州 450001)

摘 要：针对软件定义网络(SDN)中OpenFlow协议匹配字段固定且数量有限，数据流转发缺少有效的转发验证机制等问题，该文提出一种基于数据平面可编程的软件定义网络报文转发验证机制。通过为数据报文添加自定义密码标识，将P4转发设备加入基于OpenFlow的软件定义网络，在不影响数据流正常转发的基础上，对网络业务流精确控制和采样。控制器验证采样业务报文完整性，并针对异常报文下发流规则至OpenFlow转发设备，对恶意篡改、伪造等异常数据流进行转发控制。最后，构建基于开源BMv2的P4转发设备和基于OpenFlow的Open vSwitch转发设备的转发验证原型，并构建仿真网络进行实验。实验结果表明，该机制能够有效检测业务报文篡改、伪造等转发异常行为，与同类验证机制相比，在安全验证处理开销保持不变的情况下，能够实现更细粒度的业务流精确控制采样和更低的转发时延。

关键词：软件定义网络；转发验证；数据平面可编程；P4转发设备
中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2020)05-1110-08
DOI: 10.11999/JEIT190381

A Software-Defined Networking Packet Forwarding Verification Mechanism Based on Programmable Data Plane

ZUO Zhibin CHANG Chaowen ZHU Xianwei
(*Information Engineering University*, *Zhengzhou* 450001, *China*)

Abstract: For the fixed and limited number of OpenFlow protocol matching fields, and the lack of effective forwarding verification mechanism for data packet forwarding in the Software-Defined Networking (SDN), a SDN packet forwarding verification mechanism based on programmable data plane is proposed. By adding a cipher identification to the data packet, the P4 forwarding device joins the OpenFlow-based SDN network to control accurately and sample network traffic flow without affecting the normal forwarding of the data flow. The controller verifies the integrity of the sampled packet, and sends flow rules to the OpenFlow forwarding device to control the abnormal data flow such as malicious tampering and forgery. Finally, the forwarding verification prototype and simulation network based on P4 forwarding device and Open vSwitch forwarding device are constructed and tested. The experimental results show that the mechanism can effectively detect the forwarding abnormal behaviors such as packet tampering and forgery. Compared with similar verification mechanisms, in the case of the same security verification processing overhead, it can achieve more fine-grained flow precise control sampling and lower forwarding delay.

Key words: Software-Defined Networking (SDN); Forwarding verification; Programmable data plane; P4 forwarding device

1 引言

软件定义网络(Software-Defined Networking, SDN)[1]通过网络控制逻辑(控制平面)与转发流量(数据平面)的分离，从而将传统封闭的网络体系解耦为数据平面、控制平面和应用平面，简化了策略实施和网络配置[2,3]，但存在许多安全隐患和问题，如：转发设备数据转发时缺乏有效数据来源验证机制，不能有效监测数据包伪造等攻击行为且攻击者可对其行为否认[4,5]；缺乏数据完整性保护，数据信息在存储或传输过程中存在被攻击者偶然或蓄意地删除、修改、伪造、插入等破坏的可能[6,7]；此外现有的OpenFlow协议只能根据网络前4层特征

收稿日期：2019-05-24；改回日期：2019-09-28；网络出版：2020-01-31
*通信作者： 常朝稳 changchaowen5@163.com
基金项目：国家自然科学基金(61572517)
Foundation Item: The National Natural Science Foundation of China (61572517)

信息控制数据转发行为，供其操作的协议首部虽然从12个增加到41个，但控制粒度有限、缺少灵活性，难以满足网络业务数据流精确控制的需求[8]。

针对这些问题，文献[9]提出一种轻量级的软件定义网络数据包转发验证方案，该方案利用SDN本身的packet-in消息机制以及组表读取转发结点的流转发统计值，通过比较入口和出口的数据报文统计值来确定报文异常，但该方案需要同时部署于入口和出口交换机，且支持的匹配字段数量有限。Shin等人[10]提出了一种云环境下的SDN流量监控方法CloudWatcher，将网络流量自动导入相应的安全设备，以实现必要的网络报文检查。但是将流量重定向到安全设备实现数据来源验证，其实现复杂，需要综合考虑安全设备的位置、不同安全设备的协作水平和SDN流量控制的粒度等多方面因素。文献[11,12]通过在交换机转发的数据包头中嵌入一个密码标签，通过检查数据包头部的密码标签验证数据包是否正常转发，但方案均需要修改交换机原有内部实现机制才能实现。

2014年，Bosshart等人[13]提出高级“协议独立数据包处理编程语言”P4(Programming Protocol-independent Packet Processors)。这种语言支持完全可编程的解析器，可以定义并解析新的协议首部，实现协议无关的数据转发[14]。结合基于OpenFlow的SDN网络特点[15]及P4的优点，本文提出一种基于数据平面可编程的软件定义网络报文转发验证机制。通过在数据报文中添加密码标识，将P4转发设备加入基于OpenFlow的软件定义网络，在不影响数据流正常转发的基础上，由P4转发设备解析密码标识实现对网络业务数据流精确控制和采样。控制器验证采样数据报文完整性，并针对异常数据报文下发流规则至OpenFlow转发设备实现对恶意篡改、伪造等异常数据流的转发控制。实验结果表明，本文所提机制能够有效检测业务报文篡改、伪造等转发异常行为，与同类验证机制相比，在安全验证处理开销保持不变的情况下，能够实现更细粒度的业务流精确控制采样和更低的转发时延。

2　机制描述

2.1　整体结构

基于数据平面可编程的软件定义网络报文转发验证机制由终端用户、数据转发层和控制层3部分组成，其基本结构如图1所示。

(1) 终端用户：发送端用户为待发送数据报文添加数据报文密码标识，并将其存放在IP首部的Options可选字段中，该密码标识不影响数据报文正常转发与接收。数据报文密码标识的结构和功能详见2.2节。

(2) 数据转发层：数据转发层分为P4转发设备和OpenFlow转发设备。

P4转发设备主要包括采样控制表、报文镜像表。采样控制表实现基于数据密码标识中FlowID字段的业务精确控制和设置检测因子进行采样控制。报文镜像表实现将预采样的业务报文在不影响报文正常转发的情况下，镜像采样至连接控制器的端口。P4转发设备根据收到的流规则匹配转发业务流，利用数据平面可编程能力解析数据报文密码标识，实现精确控制的业务报文转发与按比例采样。

OpenFlow转发设备根据控制层中的控制器下发的流规则对数据流进行转发或拦截。

(3) 控制层：控制层中的控制器包括安全验证模块、异常监听模块、转发处理模块。安全验证模块根据业务报文中的密码标识对P4转发设备发送给控制器的采样数据包进行安全验证，确保业务报文完整性和真实性。异常监听模块监听与安全验证模块的连接，当接收到未通过验证的异常数据报后，通过Ryu控制器事件机制将异常报文发送给转发处理模块进行处理。转发处理模块负责将控制策

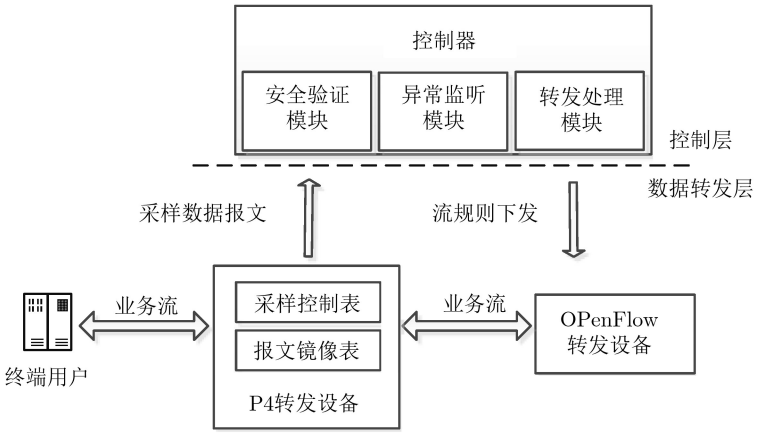

图 1 体系结构

略编译为流规则下发给数据转发层中的OpenFlow转发设备进行转发控制。

2.2 密码标识

密码标识主要用于精确标识数据流中的网络业务和验证数据报文完整性、真实性，其存放在IP首部的Options可选字段中。利用P4转发设备允许对SDN数据平面进行重新编程来实现报文转发的特点，由P4转发设备解析数据报文密码标识，将数据报文密码标识作为转发设备匹配字段，克服OpenFlow不能自定义匹配字段的不足，实现数据报文转发过程中基于数据报文密码标识的数据报文解析和匹配。

数据报文密码标识格式如图2所示，由8位组选项代码字段、业务标识字段、验证字段组成。

选项代码字段：8位组选项代码由1位拷贝标志、2位选项类和5位选项号组成。按照RFC791[16]规定，选项类为1和3的字段由于没有标准定义，适合用作自定义业务标识。在选项代码字段的保留值1或3 中选取保留值1，作为自定义密码标识类型；5位选项号选取11111作为自定义选项号，故8位组选项代码字段值为0x3F。

业务标识字段：占用3 Byte。根据业务内容等特征信息生成的业务流标识字段，具有唯一性，索引不同业务流，实现对网络业务数据流的精确控制和采样。

验证字段：占用32 Byte。存放终端用户根据HMAC-SHA256算法对数据报文生成的消息验证码，可以用于验证采样数据报文完整性和真实性，有效防止数据报文中数据篡改和伪造。

2.3 转发验证过程

如图3所示，以终端用户H1向终端用户H2发送业务数据为例，分析在该架构下的SDN网络数据转发验证过程。其中，终端用户H1通过密钥分配方案与控制器共享HMAC密钥。

(1) 终端用户H1根据业务内容等特征信息生成业务流标识FlowID，使用HMAC密钥对数据报文负载Data字段应用HMAC-SHA256算法生成消息验证码存入验证字段，根据业务流标识字段FlowID、消息验证码等信息生成数据报文密码标识，打入数据包，然后转发给P4转发设备。

(2) P4转发设备根据收到的流规则匹配转发数据流，利用数据平面可编程能力解析数据报文密码标识，在不影响数据报文正常转发的情况下，将相同业务流中的数据报文按照检测因子θ采样并发送至控制器。

(3) 控制器接收到采样数据报文后，根据密码标识中的各字段对数据报文进行安全验证，确保数据报文真实性和完整性。

(4) 对验证未通过的异常数据报文，控制器通过将控制策略编译为流规则下发给数据转发层中的OpenFlow转发设备进行转发控制。

(5) OpenFlow转发设备接收到控制器下发的流规则后，按照流规则要求对数据流进行转发或拦截。

3 实现方案

3.1 基于P4的匹配转发

P4转发设备在不影响数据流正常转发的基础上，解析数据报文密码标识，在实现对网络业务数据流精确控制的基础上，按照检测因子θ对业务报文进行采样，并将采样业务报文发送给控制器。

(1) Header(首部)：到达P4转发设备的数据包首先被包解析器处理。解析器从包首部中提取P4程序中定义的交换机所支持的协议首部，即Ethernet, Vlan, IPv4_base, Options等。数据报协议首部本质上就是有序排列的字段序列，其由有序的字段名称和对应的字段长度组成。存放密码标识的Options协议首部格式如图2所示。

(2) Parser(解析器)：在定义了首部之后，还需要定义首部之间的关系，及数据包首部解析的对应关系。由于将Options首部添加在IPv4_base首部之后，故解析顺序为Ethernet, Vlan, IPv4_base, Options。根据IPv4_base.ihl值可以判断Options首部的有效性。当IPv4_base.ihl值为0x05时，即IPv4首部长度为20个字节，未携带Options首部，故不解析Options首部，跳转到后续进行解析；当IPv4_base.ihl值不为0x05时，即携带Options首部，故解析Options首部后跳转到后续进行解析。所有的解析均从start状态开始，并在stop状态或者错误之后结束。解析器将字节流的信息解析为对应

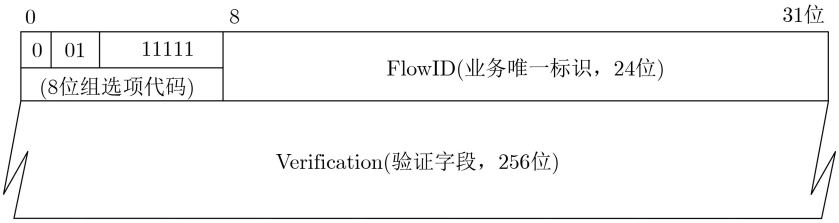

图 2 密码标识结构图

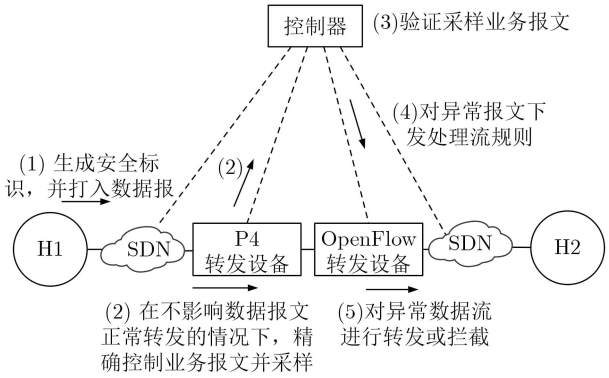

图 3 转发验证过程

的协议报文，用于后续的流表项匹配和动作执行。

(3) Table(表)：“匹配–动作”表是执行数据包处理的机制。P4程序定义的协议首部可能会用于匹配，或者在其上执行特定的动作。表的格式为Match+Action，即匹配域和对应的执行动作，而具体的流表项需要通过控制器来编程下发，从而完成对应数据流的处理。实现基于网络业务流精确控制的报文采样过程，主要涉及采样控制表和报文镜像表。

采样控制表(sampling_control表)：实现在基于Options首部中FlowID字段对业务流精确控制的基础上，以检测因子θ控制业务流的采样和检测比例。其根据设置的采样检测因子θ，应用modify_field_rng_uniform原语，按设置比例对指定业务流中的数据报文进行采样，并将采样报文的元数据字段custom_metadata.val设置为1，标示该报文为预采样报文。

报文镜像表(mirror_select表)：实现将预采样数据报文在不影响报文正常转发的情况下，镜像采样至连接控制器的端口。其应用clone_ingress_pkt_to_egressaction原语，将custom_metadata.val为1的数据报文设置镜像ID(CPU_MIRROR_SESSION_ID)为特定值100，并镜像到连接控制器的端口。根据控制器下发的流规则，所有镜像ID为100的采样数据包将被发送到通向控制器的指定端口进行处理。

(4) Control Program(控制程序)：基于Options的P4转发设备的处理过程如图4所示。

在ingress过程中，首先加载采样业务报文FlowID值和采样检测因子θ。然后判断数据报文中Options首部的有效性，即是否存在该首部。若存在，则该数据报文为有效数据报文，否则为无效数据报文。根据Options首部的FlowID字段，判断该业务流是否为采样业务流，若是，则应用采样控制表根据检测因子θ设置采样比例，并对预采样报文设置元数据字段custom_metadata.val为1。若custom_metadata.val为1，则该报文为发向控制器的预采样报文，将该报文应用报文镜像表镜像至控制器；否则，正常转发。所有具有有效Options首部的数据报文正常转发至P4转发设备出端口。报文采样不影响数据报文正常转发。

3.2 基于控制器的验证处理

控制器实现验证处理的决策过程，主要包括：安全验证模块、异常监听模块、转发处理模块。3个模块均运行在控制器上并彼此独立，安全验证模块与异常监听模块之间通过AF_UNIX通信机制实现彼此通信，异常监听模块与转发处理模块之间通过Ryu事件机制实现彼此通信。

(1) 安全验证模块：安全验证模块根据数据报文中的密码标识对P4转发设备发送给Ryu控制器的采样数据报文进行安全验证。安全验证模块收到加载有密码标识的数据报文后，提取数据报文中的Data字段和密码标识中的FlowID字段和验证字段。根据密码标识中的FlowID字段确定密钥K，使用基于哈希的消息认证算法HMAC-SHA256对报文的Data字段进行运算，生成摘要值。当摘要值与验证字段值相同，则安全模块验证通过，确认数据报文的真实性和完整性。否则，安全验证模块将验证未通过的数据报文通过AF_UNIX通信机制发送给异常监听模块进行后续处理。

(2) 异常监听模块与转发处理模块：异常监听模块和转发处理模块均基于Ryu应用实现。Ryu是基于事件机制的，其本身可以视为一个事件分配器，协助不同的Ryu应用模块传递事件。当来自转发设备的OpenFlow消息进入Ryu控制器后，都会被封装成一个事件。Ryu应用模块可以事先注册对应需要处理的事件，当有相应事件出现时，Ryu会将事件传递给所有注册该事件的应用模块，应用模块调用注册该事件的函数进行处理。Ryu也可以自定义事件类型，Ryu应用模块通过向Ryu发送自定义事件，以便与其他应用模块通信，这些事件由Ryu进行转发[17]。

异常监听模块继承自ryu.base.app_manager.RyuApp。该模块通过AF_UNIX通信机制监听与安全验证模块的连接。在Ryu正常通信阶段(main_dispatcher阶段)中，当从安全验证模块收到未通过验证的异常数据报时，生成自定义异常报文告警事件EventAlert，并使用send_event_to_observers命令将该事件发送给控制器Ryu。由Ryu控制器将该事件传递给注册该事件的转发处理模块调用处理函数进行处理。

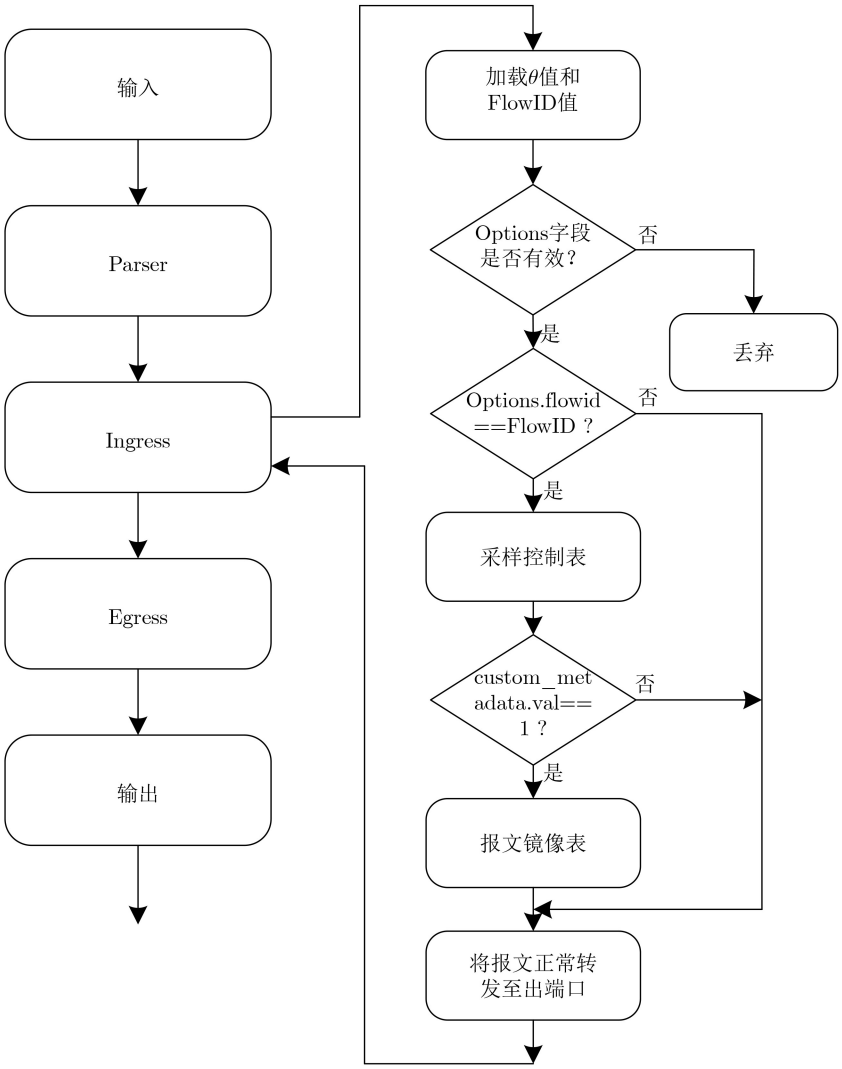

图 4 控制程序流程图

转发处理模块也继承自ryu.base.app_manager.RyuApp，对应对ofp_event.EventOFPSwitchFeatures, monitor_event.EventAlert两事件进行注册监听。通过对上述事件进行注册监听，转发处理模块实现两个主要功能：一是设置缺省转发路由，在等待接收交换机特性阶段(config_dispatcher阶段)，完成Tablemiss流表的添加。二是在正常通信阶段(main_dispatcher阶段)，使用Ryu提供的装饰器ryu.controller.handler.set_ev_cls，使其注册监听来自异常监听模块的异常报文告警事件EventAlert，并对发送过来的异常数据报下发处理流表。转发处理模块处理过程如图5所示。

4 仿真实验与分析

4.1 实验环境

使用软件交换机BMv2, Open vSwitch和Mininet构建基于P4和OpenFlow的SDN网络报文转发验证原型。将P4程序使用编译器p4c进行编译，生成数据平面的JSON格式描述文件，在启动软件交换机BMv2时将JSON描述文件导入P4交换机。对Ryu控制器增加了安全验证模块、异常监听模块和转发处理模块。图6显示了本文设置的实验评估环境，用于测试报文转发验证机制的转发延迟、安全功能和性能。

实验环境如下：主机：Intel Core i3-4170 CPU 3.70 GHz，内存：16.00 GB，其上运行：BMv2软件交换机，Open vSwitch软件交换机和Mininet。使用Scapy在发送端H1生成满足要求的数据报文，使用网络嗅探工具Wireshark抓取记录交换机端口数据报文并分析。

4.2 转发延迟

本文在主机H1上使用自定义发包脚本发送1000个数据包的正常业务流A，分别在相同网络环境下，对数据报文携带密码标识和数据报文不携带

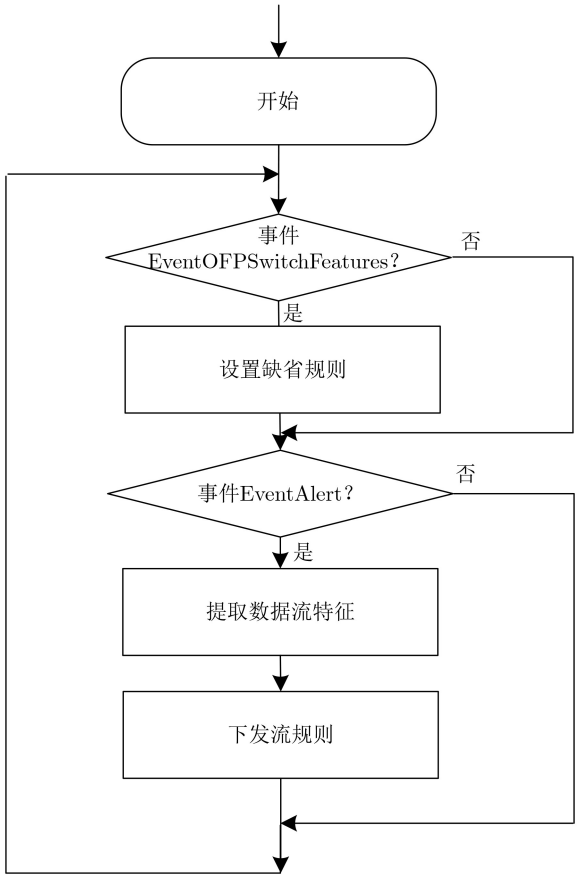

图 5 转发处理模块处理过程

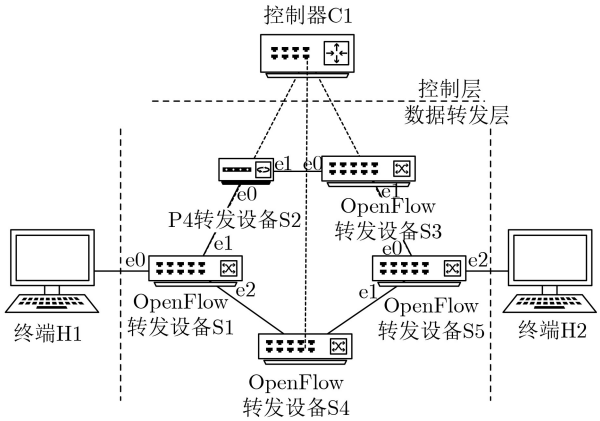

图 6 实验拓扑图

密码标识的两种情况进行验证，比较基于密码标识的数据报文和正常数据报文的转发延迟，对添加密码标识后对转发设备数据转发的影响进行评估。使用Wireshark 抓包程序分别对网络出入口S1-eth0和S5-eth2抓取数据包，计算数据报文转发延迟，描绘转发延迟的CDF图像(图7)。

本转发验证系统中，传输的数据报文比正常IP数据报文大，主要原因在于数据报文中添加密码标识额外占用288 bit(36 Byte)，其用于实现基于数据流ID定义报文转发采样行为和验证数据报文真

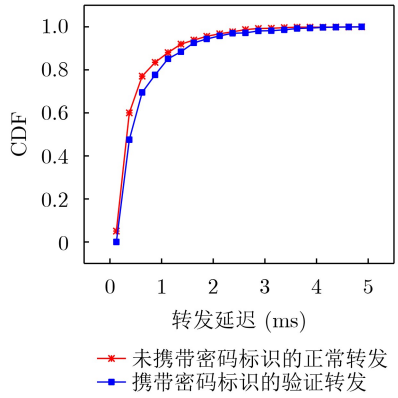

图 7 转发延迟CDF

实性和完整性。这一定程度上对转发设备数据加载效率及数据转发产生影响。如图7所示，可以看出在未携带密码标识正常转发的情况下，0.375 ms以下延迟占60%，在携带密码标识验证转发的情况下0.375 ms 以下延迟占47.5%，即在验证转发的系统中，有额外约12.5%的数据包延迟大于0.375 ms。在正常转发的系统中，有91.9%的数据包延迟在1.375 ms 以下，验证转发的系统中，这一比例为88.4%。即在验证转发的系统中，有额外约3.5%的数据包延迟大于1.375 ms。正常转发系统的报文平均转发延迟为0.74 ms，使用验证转发系统的报文平均转发延迟为0.83 ms，转发延迟增加0.09 ms，平均增加12.2%。结果显示，在该转发验证系统中，虽然为确保报文验证转发增加了密码标识，造成报文大小相比正常IP报文增加了36 Byte，但该密码标识对转发设备的数据加载和转发影响有限。

4.3 检测准确度

在主机H1上使用自定义发包程序对系统发起攻击，分别以10%的概率模拟发送伪造错误验证字段的业务流B和随机篡改数据字段的业务流C。各业务流数据报文分别为1000个，攻击各做100次，实验结果取平均值，测试在不同检测因子下报文转发验证机制的检测准确率。其中检测因子θ分别设为10, 7和3。如果报文转发验证系统检测到非法报文并下发流表，则记录为成功；未检测非法报文，则记录为漏报。实验结果如图8所示。

从图8知，在相同的检测因子θ下，伪造验证字段和篡改数据字段报文的漏报率相差不大，这是由于该机制通过验证报文携带的密码标识识别非法报文，对于伪造验证字段和篡改数据字段的报文攻击都会使密码标识验证失败。此外，检测因子θ的值越高，其伪造和篡改报文的漏报率越高，这也就意味着较高的采样比例可以获得较高的安全性能，但是同时也带来控制器CPU的使用率增大。实验发

现在检测因子θ分别设为10, 7和3的情况下，控制器CPU平均使用率分别为7.1%, 9.5%和15%。综合控制器CPU使用率和报文采样比例，当检测因子θ=7时，可以获得较好的检测准确率，同时不需要付出较高的控制器开销。

4.4 验证开销

关于控制器验证模块报文处理能力，本文通过控制器对1000个携带报文密码标识的伪造或篡改的数据报文进行验证，并下发流表。记录控制器中验证模块对数据报文的验证时间、控制器接收异常数据报文后总处理时间，如图9所示。

从图9中可知，数据报文在控制器安全验证模块中进行验证处理的平均时间为0.19 ms，控制器从接收采样报文到验证异常并下发流表的总的平均处理时间为2.12 ms，验证数据报文的时间占整个控制器处理时间的9.0%。由于控制器对数据报文验证并下发流表的平均时间为2.12 ms，这意味着在该仿真环境中控制器最多可以在每秒钟处理470个业务流。根据文献[18]可知，斯坦福大学300台主机网络(平均每5 min 120台活跃主机)，平均每秒30～40个流请求，因而该机制可以满足一般小型网络的数据流报文验证需求。

4.5 机制比较

将本文转发验证机制与最近的报文转发验证机制[9,12]进行比较，这些验证机制均是关于SDN数据平面中数据报文验证转发的。如表1所示，本文机制在对数据流的采样粒度和转发时延上均优于机制1[9]和机制2[12]，原因在于本文机制在基于OpenFlow的SDN中加入P4转发设备来对数据流进行精确控制和采样，因而其能够实现控制粒度比OpenFlow更细粒度的报文控制和采样。由于机制1和本文机制均通过控制器验证消息验证码HMAC的方式对数据报文进行验证，故其安全验证处理开销基本相当，分别为0.15 ms和0.19 ms；机制2通过在交换机内部嵌入数据验证模块来实现报文验证，且采用基于椭圆曲线上离散对数问题的无证书公钥密码体制，故其报文安全验证处理开销远高于机制1和本文机制。3种机制均构建Mininet模拟环境进行仿真实验测试报文转发时延，机制1在3层树形结构(最多经5台转发设备)下的转发延迟为33.17 ms，每台转发设备平均转发延迟为6.6 ms；机制2在4层Fattree结构(最多经7台转发设备)下转发延迟为33.65 ms，每台转发设备平均转发延迟为4.81 ms，而由4.3节可知，本文机制在3台OpenFlow转发设备和1台P4转发设备(转发路径所经过的转发设备)结构下，每台转发设备平均转发延迟仅为0.2 ms，这是由于机制1和机制2通过OpenFlow转发设备设置组表读取报文信息并对报文标识采样，且机制2在转发设备中嵌入验证模块验证报文安全，而本文机制中转发设备正常匹配转发数据报文，故对报文转发延迟基本没有影响。本文机制的不足在于，与机制1相比未提供异常报文定位功能，且机制1和机制2对SDN中任意交换机均可实现报文采样，而本文机制仅由添加的P4转发设备实现报文采样。

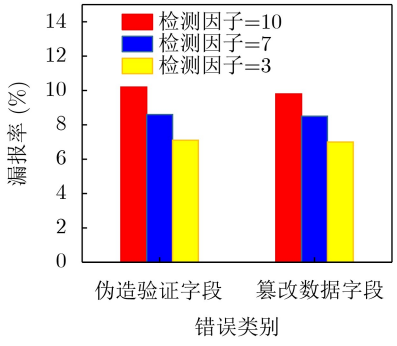

图 8 检测漏报率

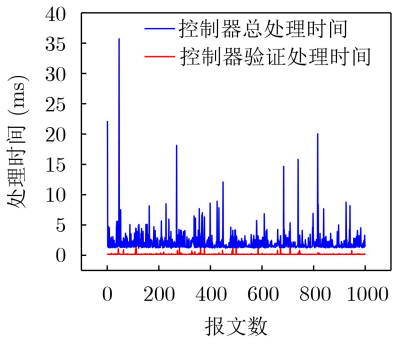

图 9 控制器处理时间

表 1 不同机制特点比较

机制	采样设备及粒度	验证设备及验证开销	转发时延	实现功能
机制1(文献[9])	任意OpenFlow交换机，OpenFlow匹配字段	控制器，0.15 ms	33.17 ms(3层树形结构)	定位并检测伪造、篡改报文
机制2(文献[12])	任意OpenFlow交换机，OpenFlow匹配字段	交换机，远大于其它	33.65 ms(4层Fattree结构)	检测伪造、篡改报文
本文机制	P4交换机，自定义匹配字段	控制器，0.19 ms	0.83 ms(3台OpenFlow转发设备和1台P4转发设备)	检测伪造、篡改报文

5　结束语

针对匹配域范围有限不能满足网络业务精确控制的需求和SDN网络数据报文转发时缺乏有效的报文验证方法等问题，本文在软件定义网络中应用P4技术，由P4转发设备解析数据报文密码标识，在不影响数据流正常转发的基础上，对网络业务流精确控制和随机抽样验证，在确保较低安全验证处理开销的情况下，实现更细粒度的业务流精确控制采样和更低的转发时延，提供解决SDN网络报文安全转发的一种新机制。

参 考 文 献

[1] MCKEOWN N. Software-defined networking[J]. *INFOCOM Keynote Talk*, 2009, 17(2): 30–32.

[2] PALIWAL M, SHRIMANKAR D, and TEMBHURNE O. Controllers in SDN: A review report[J]. *IEEE Access*, 2018, 6: 36256–36270. doi: 10.1109/ACCESS.2018.2846236.

[3] KARAKUS M and DURRESI A. Economic viability of Software Defined Networking (SDN)[J]. *Computer Networks*, 2018, 135: 81–95. doi: 10.1016/j.comnet.2018.02.015.

[4] GAO Shang, LI Zecheng, XIAO Bin, *et al*. Security threats in the data plane of software-defined networks[J]. *IEEE Network*, 2018, 32(4): 108–113. doi: 10.1109/MNET.2018.1700283.

[5] DARGAHI T, CAPONI A, AMBROSIN M, *et al*. A survey on the security of stateful SDN data planes[J]. *IEEE Communications Surveys & Tutorials*, 2017, 19(3): 1701–1725. doi: 10.1109/COMST.2017.2689819.

[6] RANA D S, DHONDIYAL S A, and CHAMOLI S K. Software Defined Networking (SDN) challenges, issues and solution[J]. *International Journal of Computer Sciences and Engineering*, 2019, 7(1): 884–889. doi: 10.26438/ijcse/v7i1.884889.

[7] SHAGHAGHI A, KAAFAR M A, BUYYA R, *et al*. Software-Defined Network (SDN) data plane security: Issues, solutions and future directions[EB/OL]. https://arxiv.org/pdf/1804.00262.pdf, 2018.

[8] OPEN Networking Foundation. OpenFlow switch specification version 1.4.0[EB/OL]. https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.4.0.pdf, 2013.

[9] 王首一, 李琦, 张云. 轻量级的软件定义网络数据包转发验证[J]. 计算机学报, 2019, 42(1): 176–189. doi: 10.11897/SP.J.1016.2019.00176.

WANG Shouyi, LI Qi, and ZHANG Yun. LPV: Lightweight packet forwarding verification in SDN[J]. *Chinese Journal of Computers*, 2019, 42(1): 176–189. doi: 10.11897/SP.J.1016.2019.00176.

[10] SHIN S and GU Guofei. CloudWatcher: Network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?)[C]. The 20th IEEE International Conference on Network Protocols, Austin, USA, 2012: 1–6. doi: 10.1109/ICNP.2012.6459946.

[11] SASAKI T, PAPPAS C, LEE T, *et al*. SDNsec: Forwarding accountability for the SDN data plane[C]. The 25th IEEE International Conference on Computer Communication and Networks, Waikoloa, USA, 2016: 1–10. doi: 10.1109/ICCCN.2016.7568569.

[12] 秦晰, 唐国栋, 常朝稳, 等. 软件定义网络中基于密码标识的报文转发验证机制[J]. 电子与信息学报, 2018, 40(9): 2042–2049. doi: 10.11999/JEIT171226.

QIN Xi, TANG Guodong, CHANG Chaowen, *et al*. Packet forwarding authentication mechanism based on cipher identification in software-defined network[J]. *Journal of Electronics & Information Technology*, 2018, 40(9): 2042–2049. doi: 10.11999/JEIT171226.

[13] 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. doi: 10.1145/2656877.2656890.

[14] The P4 Language Consortium. The P4 language specification version 1.0.5[EB/OL]. https://p4lang.github.io/p4-spec/p4-14/v1.0.5/tex/p4.pdf, 2018.

[15] PRAJAPATI A, SAKADASARIYA A, and PATEL J. Software defined network: Future of networking[C]. The 2nd IEEE International Conference on Inventive Systems and Control, Coimbatore, India, 2018: 1351-1354. doi: 10.1109/ICISC.2018.8399028.

[16] Defense Advanced Research Projects Agency. RFC 791: Internet protocol[EB/OL]. http://www.faqs.org/rfcs/rfc791.html, 1981.

[17] Ryu Development Team. Ryu documentation release 4.30[EB/OL]. https://ryu.readthedocs.io/en/latest/library_packet.html, 2019.

[18] CASADO M, FREEDMAN M J, PETTIT J, *et al*. Ethane: Taking control of the enterprise[C]. 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Kyoto, Japan, 2007: 1–12. doi: 10.1145/1282380.1282382.

左志斌：男，1979年生，博士生，研究方向为SDN、网络安全.

常朝稳：男，1965年生，教授，博士生导师，研究方向为网络安全、态势感知.

祝现威：男，1991年生，博士生，研究方向为SDN、信息安全.