基于属性签名标识的 SDN 数据包转发验证方案

常朝稳，金建树，韩培胜，祝现威
（信息工程大学，河南 郑州 450001）

摘 要：针对软件定义网络（SDN）中数据包缺乏有效转发验证机制的问题，提出了一种基于属性签名标识的数据包转发验证方案。首先，根据用户的身份属性生成属性签名标识，并为数据包打上属性签名标识。然后，使用 P4 转发设备对数据包进行精确控制与采样，控制器对采样数据包进行属性签名验证，OpenFlow 转发设备根据控制器下发的流表对转发异常的数据包进行处理。最后，构建了多控制器架构，避免了控制器单点失效故障。实验结果表明，所提方案实现了对数据包的精确控制与采样，能有效检测数据包篡改、伪造等异常行为，其网络时延处于可行通信时延范围内。

关键词：软件定义网络；属性签名；转发验证；P4 转发设备
中图分类号：TP393
文献标识码：A
DOI: 10.11959/j.issn.1000−436x.2021079

Software-defined network packet forwarding verification scheme based on attribute-based signatures identification

CHANG Chaowen, JIN Jianshu, HAN Peisheng, ZHU Xianwei
Information Engineering University, Zhengzhou 450001, China

Abstract: Aiming at the lack of effective forwarding verification mechanism for packet in software defined network (SDN), a data packet forwarding verification scheme based on attributed-based signatures identification was proposed. First, the attribute signature identification was generated according to the user's identity attribute, and the data packet was marked by the attribute signature identification. Then, the P4 forwarding device was used to control accurately and sample the data packet. The controller verified the attribute signature of the sampled data packet. The OpenFlow forwarding device processes the abnormal data packets according to the flow table issued by the controller. Finally, a multi-controllers architecture was constructed to avoid the single point failure of the controller. The results of the experiment indicate that the scheme can achieve accurate control and sampling of data packet, effectively detect the forwarding abnormal behaviors such as packet tampering and forgery, and the network delay is within the range of feasible communication delay.

Keywords: software-defined network, attribute signature, forwarding verification, P4 forwarding device

1 引言

软件定义网络（SDN, software-defined networking）[1]通过采用集中的控制平面与分散的数据转发平面，并将 2 个平面解耦，从而实现了高度集中的控制转发能力以及较灵活的扩展能力[2-3]。但是，随着 SDN 的深入发展，产生了不少安全问题，如：1) 对于 SDN 中的数据流缺乏安全性验证的缺陷，会导致恶意用户发送一些非法数据流被 SDN 正常转发；2) SDN 与传统网络相比更加开放，一旦有恶意入侵，由于其隐蔽性极强，对于入侵者的流量篡改以及恶意窃听等攻击方式，现

收稿日期：2020−12−03；**修回日期：**2021−03−01
基金项目：国家自然科学基金资助项目（No.61572517）
Foundation Item: The National Natural Science Foudation of China (No.61572517)

有的 SDN 体系很难识别以及防御[4-5]；3) 数据完整性的保护较弱，正常用户的数据包被篡改后很难被检测[6-7]。

王首一等[8]提出了一种数据包安全转发验证方案，通过 SDN 特有的消息机制以及统计数据流转发设备的流转发量，由控制器对数据包的 MAC 值和其他统计信息进行对比，从而达到检测异常转发行为的功能，但该方案需要同时控制交换机的出入口，且适用的匹配字段数量有限。Sasaki 等[9]提出了一种 SDNsec 方法，在数据流中添加特定的流标签，从而验证数据是否安全转发，但需要通过修改交换机内部原有实现机制来实现。秦晰等[10]提出了一种基于密码标识的数据包验证模型，通过在数据包中嵌入密码标识并设计 SDN 匿名认证协议，由认证交换机对数据流进行验证，但该方案需要在交换机上开发安全模块，且在密钥的存储与管理方面存在问题。

在传统的基于身份的密码体制中，通常使用一段字符串来标识用户的身份。通信方式是一对一的，验证者通过验证签名者的签名来确认签名方的身份。但是，将这种方式应用到 SDN 这样的分布式网络中，不仅在密钥管理和分配上较为烦琐，缺乏有效的管理方式确保密钥的安全性，而且会引入较大的通信开销。而属性签名作为数字签名的延伸，采用一对多的通信方式，被验证者使用自身的属性集对消息进行签名，验证者通过判断得到的属性集合是否与被验证者声称的属性集合一致来判断签名的真伪[11]。这种方案可以细粒度地划分身份特征，具有很强的匿名性和灵活性，很适合 SDN 环境下的数据安全性验证。

P4 语言[12]是一种高级编程语言，它主要使用可编程包处理器来对数据平面进行编程。基于 P4 语言的转发设备可以自定义数据包的转发处理动作，而不再受到现有协议的约束[13]。

本文提出了一种基于属性签名标识的 SDN 数据包转发验证转发方案，由 P4 转发设备对用户发出的数据包添加属性签名标识，同时完成了数据包采样以及解析转发控制等功能。然后使用在 SDN 控制器上开发 App 的方法完成了对数据包的属性签名的验证，以及对于异常数据流的转发控制等功能。此外，本文还引入了多控制器架构解决了控制器单点失效问题。

本文方案主要面向工业控制系统、战术互联网以及校园网等场景。这些系统的特点是系统较为封闭、用户数量相对较少但安全性要求较高。属性签名方案可以实现数据流的真实性与完整性检测，在保证用户匿名性的同时实现了用户身份可追踪功能，但会带来一定的时间开销。这些特点决定了该方案适用于上述场景。在实验环节，本文搭建了一个与上述场景网络拓扑相似的实验环境，并测试了方案的功能和性能。

本文主要的贡献有以下 4 点。

1) 将属性签名算法应用到 SDN 数据流转发验证中，通过对用户属性的细粒度划分以及对属性集合的签名验证实现了对用户在网络中发送的数据流的精确转发验证，从而实现了数据流的真实性和完整性验证，以及用户身份可追踪等功能。

2) 引入 P4 转发设备实现了 SDN 数据平面可编程功能，通过在交换机对数据流进行解析的方式实现了添加属性签名标识、随机采样以及转发控制等功能。

3) 引入主从控制器模式，优化了控制器性能，避免单点失效故障。

4) 实际部署了属性签名验证原型系统，并在 Mininet 环境下中进行了实验验证与分析。

2 方案内容

2.1 方案描述

2.1.1 基本原理

针对 SDN 数据流缺乏有效转发验证方法以及网络中的数据流缺乏精确控制等问题，本文提出了一种基于属性签名标识的 SDN 数据包转发验证方案。通过将用户的属性与其所发送数据流进行绑定从而实现数据流的安全转发。用户的属性是指每个用户所具有的属性集合，系统对该属性进行签名操作，并由控制器对其进行签名验证，从而判断数据流在转发过程中是否出现异常情况。同时，将数据平面可编程交换机 P4 引入该方案中，实现了对数据流的精确采样控制功能。该方案的核心技术为属性签名方案以及数据平面可编程技术，二者共同构建了该安全体系。

2.1.2 整体结构

基于属性签名标识的数据包转发验证方案由以下几个部分组成，其基本结构如图 1 所示。

1) 属性签名标识认证中心。首先，生成系统公开参数与主密钥；然后，生成访问控制结构的公开

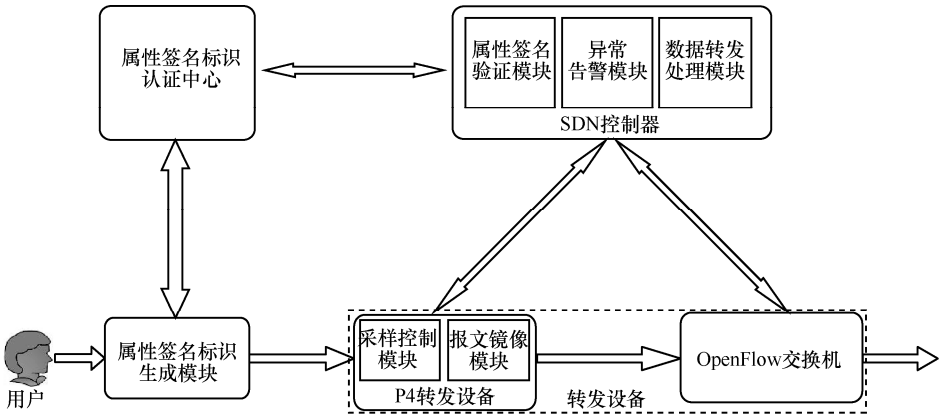

图 1　转发验证方案基本结构

参数，根据用户的属性集生成用户身份属性的访问控制结构的相关参数；最后，根据属性标识生成用于属性签名的属性私钥。

2) 用户，是指所有接入该 SDN 的访问用户。每个访问用户都会有属性信息（例如 XX 大学 XX 学院 XX 系王 XX 教授）。本文方案通过采集用户的独有属性信息并生成属性集合来实现针对每一个用户的属性签名，每个属性集合都代表着唯一的用户。

3) 属性签名标识生成模块。该模块是属性签名标识管理的核心。其功能是为用户生成属性集，并发送给认证中心；接收用户的属性私钥，并用该私钥对 IP 数据包进行签名，通过修改 IP 数据包格式的方式将签名封装到数据包中。在文献[14]中，该组件以应用程序的形式安装在用户主机上。但是，考虑到首先为每个用户所在的主机安装应用程序，增加了系统的开发成本以及管理难度；其次安装在用户主机上的组件对于用户而言是不透明的，攻击者可以通过入侵该应用程序的方式截获签名标识，从而给系统带来一定的安全威胁，由于 P4 转发设备具有数据平面可编程的功能，本文方案将属性签名标识生成组件安装在 P4 转发设备上，由 P4 交换机完成属性签名标识的生成这一功能。

4) 转发设备，包括 P4 转发设备与 OpenFlow 交换机。

P4 转发设备，实现了数据平面可编程功能。其主要由采样控制模块以及数据包镜像模块组成，主要负责属性签名标识的生成、数据包的采样检测以及精确转发。

OpenFlow 交换机主要负责接收控制器发出的流表，并根据流表中的策略规则，对数据包进行转发、丢弃等动作。

5) SDN 控制器（下文简称为控制器），由属性签名验证模块、异常告警模块以及数据转发处理模块组成。其中，属性签名验证模块接收待检测的数据流，从中解析出属性签名，对其进行属性签名验证，并将未通过检测的数据包转发至异常告警模块。异常告警模块在接收到未通过检测的数据包后，通过事件消息机制转发给数据转发处理模块。数据转发处理模块下发相应的流策略至 OpenFlow 转发设备来对数据进行转发控制。

模型的安全性基于以下假设。

1) 控制平面与数据平面之间采用带外的组网方式。

2) 假设控制器是安全的，恶意用户无法攻击并控制 SDN 控制器。

3) 属性签名标识认证中心与控制器采用 TLS 信道进行通信。

2.1.3　属性签名标识

属性签名标识主要用来精确定义用户的身份以及进行数据流验证，它被嵌入 IP 首部的 Option 字段中。

属性签名标识格式如图 2 所示。IP 头部由 20 B 组成；Option 字段共 40 B，其中用户属性集合字段长度为 8 B，属性签名字段长度为 32 B。

图 2　属性签名标识格式

用户属性集合字段，长度为 8 B，包含了数据包进行属性签名时所用到的该用户的所有属性。该字段传送到控制器后，控制器会对其进行解析并提

供给属性签名验证模块用来进行属性签名验证。

属性签名字段。Option 字段为属性签名字段预留了 32 B。该字段包含了属性签名生成模块对用户属性进行签名之后所产生的签名值。签名值包括了 C_1、C_2、C_3、c、$\{\mathrm{CT}_i\}_{i\in\zeta}$、$s_\alpha$、$s_\beta$、$s_x$、$s_{\delta_1}$、$s_{\delta_2}$、$\eta$，其长度是可变的，但是不会超过 32 B，因此一共为该字段预留 32 B。

2.2 属性签名方案

2.2.1 属性签名方案理论性阐述

本文方案采用 Khader[15]提出的基于属性的群签名（ABGS, attribute-based group signature）方案。该群签名方案的主要原理是验证者构造一个访问控制树 T，而所有签名者（被验证者）根据自身的属性进行签名，验证者对签名进行解密并判断该用户的属性是否满足访问控制树 T 来验证签名者的身份。通过横向对比诸多属性签名方案，该方案在实现属性签名的功能外，还引入了较低的验证开销，并细粒度地划分身份特征，能够很好地保护用户个人隐私信息[16]，故本文方案中使用群签名方案。本文方案通过属性签名标识认证中心、属性签名标识生成模块以及属性签名验证模块共同实现访问控制结构的生成以及数据包的属性签名验证。其过程如图 3 所示。

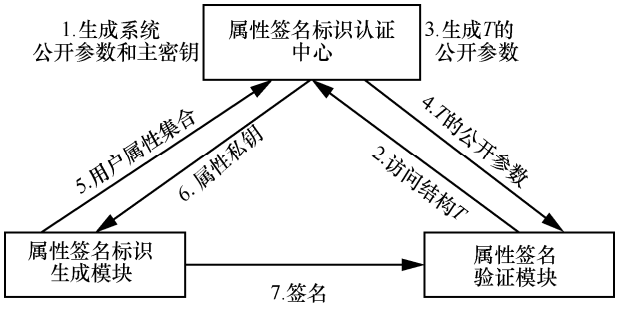

图 3 属性签名具体过程

首先，属性签名标识认证中心（下文简称为认证中心）生成系统的公开参数与主密钥；然后，属性签名验证模块生成用户的访问控制树结构 T 并上传给认证中心。认证中心根据 T 生成 T 的公开参数，并发送给属性签名验证模块。属性签名生成模块将用户的属性集合发送给认证中心，认证中心根据用户属性集合生成相应的属性私钥并发给属性签名标识生成模块。属性签名标识生成模块使用数据包消息数据、属性私钥以及系统的公开参数进行哈希计算，并将计算值通过 P4 转发设备转发给属性签名验证模块，由属性签名验证模块对其进行验证。当用户的访问控制结构发生改变时，只需由属性签名验证模块上传新的 T 给认证中心，从而获得新的 T 公开参数。

2.2.2 访问树结构的生成过程

访问控制结构 T 用来表示属性签名验证的属性集，由于其使用属性树 Γ 来表示，故称为访问树结构。这里使用 Goyal 等[17]的方法来构造访问树。在属性树中所有非叶子节点表示由其子节点和阈值所构成的门限，而每个叶子节点作为属性值与其连接。门限值表示所有与其相连的叶子节点所满足的最低条件数。举例说明访问树结构如图 4 所示。

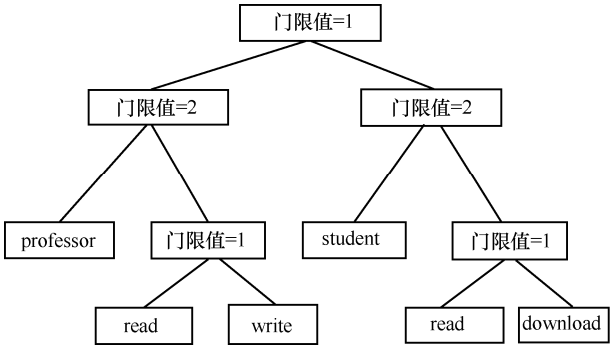

图 4 访问树结构

只有满足访问树中属性要求的用户才能通过属性签名验证。如图 4 所示，一个 professor 想要完成 read 操作，因其满足访问树要求，故可以通过属性签名验证。而如果一个 student 想要完成 write 操作，其不满足访问树要求，故不可以通过签名验证。

2.2.3 方案形式化定义

本节给出基于属性的群签名的相关定义。

定义 1 用属性树 Γ 来表示访问结构 T，属性树的顺序为“从上到下，从左到右”，非叶子节点表示为(m, n)。其中，m 表示门限值，n 表示其叶子节点的总个数，κ 表示该树的叶子节点总数。图 4 中的属性树可以表示为

$$\Gamma=\{(1,2),(2,2),(2,2),\text{professor},(1,2),\text{student},(1,2),\text{read},\text{write},\text{read},\text{download}\}$$

定义 2 γ_i 表示每个用户所拥有的属性，μ 表示属性数即 γ_i 的数量。

定义 3 ζ_i 表示用于签名时所使用的属性。它是用户所有属性的子集，即 $\zeta_i\subseteq\gamma_i$。例如 $\Gamma=\{(1,2),\text{professor},\text{student}\}$，员工 i 使用 $\zeta_i=\{\text{student}\}$ 可以通过验证，τ 表示 ζ_i 的个数。

定义 4 y_T 表示用户满足 T 的属性集合。

该算法共包括以下 4 个步骤。

步骤 1　初始化（Setup）。认证中心选择一个双线性对：$e:G_1\times G_2\to G_T$，G_1、G_2 和 G_T 是 p 阶乘法循环群，选择 G_2 的生成元 g_2，经同态映射计算得出 $g_1\leftarrow\psi(g_2)$。认证中心选择一个公开的 Hash 函数：$H:\{0,1\}^*\to Z_p^*$。对于任意的 $i\in Z_p^*$ 以及在 Z_p^* 内的一个集合 S，定义一个拉格朗日系数 $\Delta_{i,s}$，根据拉格朗日插值公式可得

$$\Delta_{i,s}(x)=\prod_{\substack{j\in S\\ j\neq i}}\frac{x-j}{i-j} \tag{1}$$

选择 $h\in G_1$ 以及随机的 ξ_1 和 ξ_2（$\xi_1,\xi_2\in Z_p^*$），选择 μ，ν 且 $\mu^{\xi_1}=\nu^{\xi_2}=h$。随机选择 $\omega\in Z_p^*$，计算 $W=g_2{}^{\omega}$。定义一个属性空间 $U=\{1,2,\cdots,m\}$，为每个属性 $j\in U$ 选择 $t_j\in Z_p^*$。

系统的公开参数 $PK=\{G_1,G_2,G_T,g_1,g_2,e,H,h,u,v,W\}$

系统的秘密参数 $\mathrm{MK}=<\{t_i\}_{i\in U},\omega,\xi_1,\xi_2>$

步骤 2　密钥生成（KeyGen），主要完成为访问结构 T 生成公共参数以及为用户生成属性私钥两部分工作。认证中心通过 γ 为用户 i $(1\leqslant i\leqslant n)$ 生成一个基私钥 $\mathrm{gsk}[i]_{\mathrm{base}}=(A_i,x_i)$，这里要求 $\mathrm{gsk}[i]_{\mathrm{base}}$ 是一个 SDH 对，即 $A^i=g_1^{1/(\omega+x_i)}\in G_1$。

1) 生成 T 的公开参数（$\mathrm{KeyGen}_{\mathrm{public}}(T)$）。首先，为 T 中每个非叶子节点选择一个多项式 q_x，其构造方式如下：对于树中的任意节点 x，其多项式 q_x 的次数 d_x 小于其阈值 k_x，即 $d_x=k_x-1$；对于树中的所有根节点 r，设定 $q_r(0)=\omega$；然后，随机选取多项式 q_r 的构造，对于其他节点令其满足 $q_n(0)=q_{\mathrm{parent}}(\mathrm{index}(x))$，随机选取多项式 q_n 进行递归构造属性树多项式；最后，通过计算 $D_x=g_2{}^{q_x(0)/t_i}$、$h_n=h^{t_i}$ 和 $i=\mathrm{att}(x)$ 获取 T 的公开参数 $\mathrm{TPK}=<\{D_x,h_x\}_{x\in\gamma_T}>$，将 TPK 和系统公开参数 PK 发送给控制器中的属性签名验证模块。

2) 生成用户属性私钥（$\mathrm{KeyGen}_{\mathrm{private}}\cdot(\mathrm{gsk}[i]_{\mathrm{base}},\gamma_i)$）。为拥有属性 γ_i 的用户颁发私钥。对于属性 $j\in\gamma_i$，计算 $\Gamma_{i,j}=A^{t_j}$，并获得用户私钥 $\mathrm{SK}=<A_i,x_i,\{T_{i,j}\}_{j\in\gamma_i}>$。

步骤 3　签名（Sign）。对于用户的属性集合 $\gamma_i\subseteq U$，$j\subseteq\gamma_i$，访问树 T 的公开参数和消息 m，签名的步骤如下。

1) 选择签名属性 $\zeta\subseteq\gamma_i$ 和随机数 α、β、$\mathrm{rnd}\in Z_p^*$。

2) 计算 A_i 和 T_{ij} 的线性加密。$C_1=u^a$，$C_2=v^{\beta}$，$C_3=A_ih^{\alpha+\beta}$，$\mathrm{CT}_j=(T_{i,j}h_j{}^{\alpha+\beta})^{\mathrm{rnd}}$，其中 $j\in\zeta$。

3) 计算 $\delta_1=x_i\alpha$，$\delta_2=x_i\beta$，然后选择随机数 r_α，r_β，r_x，r_{δ_1} 和 r_{δ_2}，通过随机数计算 $R_1=\mu^{\gamma_\alpha}$，$R_2=\nu^{\gamma_\beta}$，$R_3=e(C_3,g_2)^{\gamma_x}e(h,\omega)^{-\gamma_\alpha-\gamma_\beta}e(h,g_2)^{-\gamma_{\delta_1}-\gamma_{\delta_2}}$，$R_4=C_1^{\gamma_x}u^{-\gamma_{\delta_1}}$ 和 $R_5=C_2^{\gamma_x}v^{-\gamma_{\delta_2}}$。

4) $c=H(M,C_1,C_2,C_3,R_1,R_2,R_3,R_4,R_5)\in Z_p{}^*$。

5) 构造值 $s_\alpha=(\gamma_\alpha+c\alpha)$，$s_\beta=(\gamma_\beta+c\beta)$，$s_x=(\gamma_x+cx)$，$s_{\delta_1}=(\gamma_{\delta_1}+c\delta_1)$ 和 $s_{\delta_2}=(\gamma_{\delta_2}+c\delta_2)$。

6) 使 $\eta=W^{\mathrm{rnd}}$，得出属性签名 $\sigma=(m,C_1,C_2,C_3,c,\{\mathrm{CT}_i\}_{i\in\zeta},s_\alpha,s_\beta,s_x,s_{\delta_1},s_{\delta_2},\eta)$。将签名 σ 以及用户的属性集合 ζ 发给属性签名验证模块。

步骤 4　验证（Verify）。属性签名验证模块收到签名后，分以下两步进行验证。

首先，验证属性集合是否满足 T。如果节点 x 是属性树中的一个叶子节点，则根据递归算法 $\mathrm{Ver}_{\mathrm{Node}}$。

$$\mathrm{Ver}_{\mathrm{Node}}(x)=\begin{cases}e(\mathrm{CT}_j,D_j),j=\mathrm{att}(x)且j\in\zeta\\ \perp,其他\end{cases} \tag{2}$$

其中，$e(\mathrm{CT}_j,D_j)=e(A_ih^{\alpha+\beta},g_2^{\mathrm{rnd}})^{q_j(0)}$。

假如 x 节点是非叶子节点，则算法过程如下。对于非叶子节点 x 的子节点 z，调用 $\mathrm{Ver}_{\mathrm{Node}}$ 算法并将结果放入函数 F_z 中；递归求出父节点的 F_x 值。在 $j\in\{\mathrm{index}(z):z\in s_x-\mathrm{index}(z)\}$ 条件下，计算 $\Delta s_{x,\mathrm{index}(z)}=\prod(-j/(\mathrm{index}(z)-j))$，并且进行如下计算获取父节点值。

$$F_x=\prod_{z\in s_x}F_z{}^{\Delta s_{x,\mathrm{index}(z)}}=$$
$$\prod_{z\in s_x}(e(A_ih^{\alpha+\beta},g_2^{\mathrm{rnd}})^{q_z(0)})^{\Delta s_{x,\mathrm{index}(z)}}=$$
$$\prod_{z\in s_x}(e(A_ih^{\alpha+\beta},g_2^{\mathrm{rnd}})^{q_{\mathrm{parent}(z)}(\mathrm{index}(z))})^{\Delta s_{x,\mathrm{index}(z)}}=$$
$$e(A_ih^{\alpha+\beta},g_2^{\mathrm{rnd}})^{q_x(0)}$$

然后，使用拉格朗日插值法递归求出根节点 F_r 的值，并验证 $F_r=e(C_3,\eta)$。如果成立，表示签名满足访问树 T，则开始第二步验证，继续进行如下计算过程；否则拒绝接收此签名。

$\overline{R}_1 = u^{s_\alpha} C_1^{-\kappa}$ ， $\overline{R}_2 = v^{s_\beta} C_2^{-\kappa}$ ， $\overline{R}_4 = C_1^{s_x} u^{-s_{\delta_1}}$ ，

$\overline{R}_3 = e(C_3, g_2)^{s_x} e(h, W)^{-s_\alpha - s_\beta} e(h, g_2)^{-s_{\delta_1} - s_{\delta_2}} \left(\dfrac{e(C_3, W)}{e(g_1, g_2)} \right)^c$ ，

$\overline{R}_5 = C_2^{s_x} v^{-s_{\delta_2}}$

若满足 $c = H(m, C_1, C_2, C_3, \overline{R}_1, \overline{R}_2, \overline{R}_3, \overline{R}_4, \overline{R}_5)$，则签名通过验证；否则，则视为签名未通过验证。

2.2.4 时间复杂度分析

本文方案引入属性签名算法，会引入额外的时间开销。下面就该签名算法本身进行时间复杂度分析。假设 T_{em} 表示群上的点乘运算，T_{ea} 表示群上的点加运算，T_{ep} 表示群上的幂运算，T_{bp} 表示双线性对运算，T_{pn} 表示群上的多项式求值运算。而其他运算（例如 Hash 运算）的时间开销远小于这些计算，故忽略不计。本文方案一共分为 2 个过程，即签名过程和验证过程。方案的签名过程需要 $2k$+9 个 G_1 群上的幂运算（其中 k 为属性集合 ζ 的大小）、k+4 个 G_1 群上的点乘运算、一个 G_2 群上的幂运算、3 个双线性对运算和 3 个 G_T 上的幂运算，总时间为 $3T_{\text{bp}} + (2k+13)T_{\text{ep}} + (k+4)T_{\text{em}}$。而方案的签名过程需要 8 个 G_1 群上的幂运算、4 个 G_1 群上的点乘运算、4 个 G_T 群上的点乘运算、k+6 个双线性对计算和至多 hk 个 G_T 群上的多项式运算（h 为访问树的高度），总时间为 $8T_{\text{ep}} + 8T_{\text{em}} + (k+6)T_{\text{bp}} + hkT_{\text{pn}}$。可以看出，由于需要进行双线性对运算以及群上的运算，该方案的实施过程会产生一些时间开销。

2.3 数据包转发验证方案具体实现

本章将具体介绍基于 P4 可编程交换机的数据流转发过程以及控制器上各个模块的具体运行过程。

2.3.1 基于 P4 转发设备的数据平面转发过程

P4 转发设备作为数据平面可编程的中间件，可以实现对用户发出的数据包的精确解析，同时通过设置检测因子 θ 来完成对数据包的检测采样。被选中的数据包经 P4 转发设备发送至控制器，供控制器进行属性签名验证，整个采样过程并不影响其他数据流的转发[18]。下面介绍 P4 转发设备的主要模块。

首部（Header）。首部指定了属于该首部的字段、字段大小以及首部中字段的顺序。

解析器（Parser）。解析器的功能是将数据包映射成以状态机形式编写的首部，然后按照规定顺序解析首部中的所有协议。在将属性签名数据包加载至 Option 字段后，解析顺序为：Ethernet、Vlan、IPv4_base、Option。这里根据 IPv4_base 中的 ihl 值来验证 Option 首部的有效性。当 IPv4_base.ihl 值为 0x05 时，即 IP 首部长度为 20 B，说明该数据包未携带 Option 首部，即该数据包未携带属性签名，因此为非法数据包；当值不为 0x05 时，说明 Option 字段携带了属性签名，在解析 Option 字段后按照规定顺序继续解析。解析状态一共有 3 种：开始（Start）、接收（Accept）、拒绝（Reject）。解析从开始状态开始，如果解析正常则进入接收状态，进行下一步操作。如果解析出现错误，则转入拒绝状态，从而解析结束，输出错误信息。若解析正常，解析器将数据流信息解析为对应的协议数据包，用于后续的流表项匹配和动作转发执行[19]。

表（Table）。P4 转发设备采用“匹配–动作”表机制来处理数据包的转发。此表包括 3 项内容：key 值、Action 与 ActionData。其中，key 值为匹配域，表示动作要匹配的具体实例（例如源 IP 地址等）；Action 为具体行为，P4 语言可以通过自主编程来定义转发行为；ActionData 表示具体的动作数据。在本文机制中重点定义了 2 种表，即数据包采样表和数据包镜像表。

数据包采样表。其设置检测因子 θ 为其他正常数据包/采样检测数据包，根据检测因子 θ 应用 modify_field_rng_uniform 原语，对数据包进行采样。设置一个自定义元数据字段 sample_metadata 用于标识采样数据包，当数据包确定为预采样数据包后，其 sample_metadata 字段的 val 值被设为 1。

数据包镜像表。其主要功能是根据采样结果，将预采样数据包复制至连接控制器的端口。其应用 clone_ingress_pkt_to_egressaction 原语将 sample_metadata.val 为 1 的数据包设置镜像 ID 为特定值 10，并镜像到连接控制器的端口。所有镜像 ID 为 10 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。

流控制程序（Control Program）。主要包括以下几个部分：Input、Parser、Ingress、Egress、Output。控制程序流程如图 5 所示。

Ingress 过程中，首先加载 θ 值，然后判断 Option 字段是否存在。若存在，则将数据包加载至数据包采样表进行数据包采样；否则视为无效数据包，控制器负责将此数据包丢弃。数据包采样表根据预设的 θ 值确定采样比例后，进行采样操作。预采样数据包的自定义元数据字段

sample_metadata 中的 val 值被设为 1。所有 sample_metadata.val 为 1 的数据包将被加载至数据包镜像表，然后镜像至控制器，其余数据包正常转发。所有具有属性签名标识的数据包正常转发至输出端口。

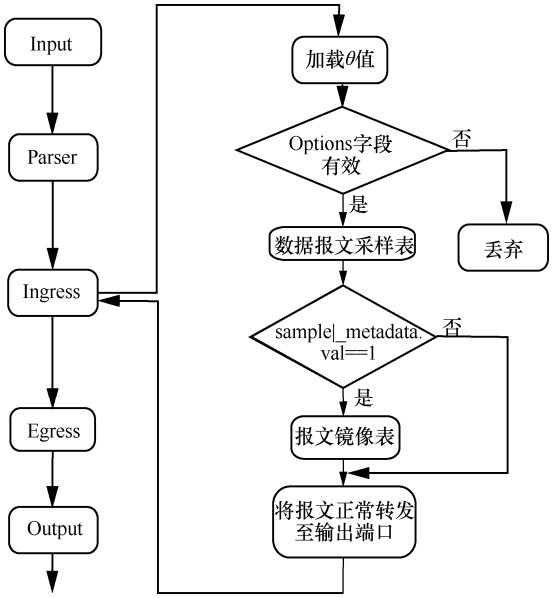

图 5　控制程序流程

2.3.2　控制器各个模块的运行过程

控制器中主要包括了 3 个模块：属性签名验证模块、异常告警模块以及数据转发处理模块。前 2 个模块之间使用 AF_UNIX 域上的 socket 协议进行通信。而异常告警模块与数据转发处理模块通过控制器的事件机制进行通信[20]。

1) 属性签名验证模块

该模块收到数据包后，首先提取用户用于属性签名的集合，看其是否满足访问树结构 T。若满足，进行下一步验证；否则，视为无效数据包，将其发送给监听告警模块。接着，从属性签名标识中提取出 C_1、C_2、C_3、$\{\mathrm{CT}_i\}_{i\in\zeta}$、$s_\alpha$、$s_\beta$、$s_x$、$s_{\delta_1}$、$s_{\delta_2}$、$\eta$，计算 $\bar{R}_1$、$\bar{R}_2$、$\bar{R}_3$、$\bar{R}_4$、$\bar{R}_5$。然后，计算 $H(m,C_1,C_2,C_3,\bar{R}_1、\bar{R}_2、\bar{R}_3、\bar{R}_4、\bar{R}_5)$，将其与签名 c 进行比较，若相等，则验证通过；否则，发送给异常告警模块处理。

2) 异常告警模块

当收到无效数据包时，该模块发送自定义异常数据包告警事件 EventWarning 至 Ryu 控制器。控制器通过 Ryu 的事件机制将 EventWarning 事件发送至数据转发处理模块进行下一步处理。

3) 数据转发处理模块

数据转发处理模块在正常通信阶段，接收来自监听告警模块的异常数据包告警事件 EventWarning，然后下发相应的流规则至 OpenFlow 交换机对异常数据包进行相应处理。若发送过来的数据包是正常数据包，则直接由 OpenFlow 交换机进行正常的转发工作。数据转发处理模块工作过程如图 6 所示。

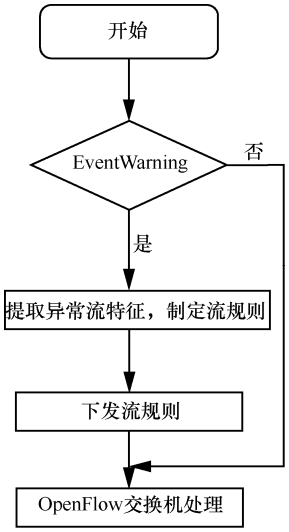

图 6　数据转发处理模块工作过程

2.4　基于容灾机制的多控制器架构

单控制器的单点失效问题一直是困扰 SDN 控制器性能方面的一个重要问题[20]。在本文方案中，由于向控制器中添加了部分功能模块，控制器除了要完成正常的数据转发控制功能外还需要完成属性签名验证以及异常处理等功能，因此其受到单点失效问题的影响比正常控制器要大。针对这一问题，本文设计了一种多控制器集群的控制器容灾机制[21]。当主控制器失效时，通过选举机制将备份控制器切换为主控制器并接管原主控制器的工作，从而解决了控制器单点失效问题。基于目前已经成熟的分布式系统集群管理技术 Zookeeper，解决了多控制器之间信息共享与数据一致性问题，并根据 OpenFlow1.5 协议支持多控制器协同工作的特性，实现了交换机与多控制器通信的功能。

2.4.1　基本架构

多控制器架构如图 7 所示。首先，采用 Zookeeper 管理服务器来负责管理集群内部所有控制器。Zookeeper[22]是雅虎研究院的一个分布式数据管理系统项目，该项目通过允许用户调用其接口的

方式来实现分布式一致性服务。使用 Zookeeper 管理多控制器集群，可以提供分布式协调管理服务、数据一致性机制等。所有控制器均与 Zookeeper 服务器建立通信连接，并通过开发通信模块的方式完成对 Zookeeper 消息的监听与响应[21]。

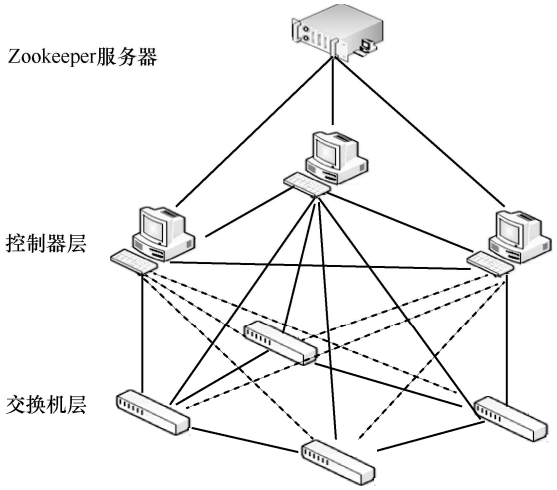

图 7 多控制器架构

在控制器层使用 3 个控制器进行集群部署（根据系统的可扩展性，可以向系统动态增加控制器数量，但本文机制使用 3 个控制器已经能够满足系统的可用性）。控制器间采用主从（Master/Slaves）结构（主从控制器模式），其中一个控制器为主控制器，2 个控制器为从控制器。在正常通信阶段，由主控制器负责控制数据平面的数据转发。从控制器在主控制器工作时处于热备份模式，即只备份 SDN 数据信息并与主控制器保持同步，但不下发流表给交换机。控制器之间需要建立通信链路用来进行信息交互。控制器间交互数据包主要有 3 种类型，即同步数据包、请求转发数据包以及保活数据包[23]。同步数据包由主控制器发送给从控制器，内容包括控制器自身状态变化信息。请求转发数据包将发送至控制器的数据包转发至负载比它小的其他控制器。保活数据包主要用来测试连接状态和网络时延。

在交换机层中每个交换机可以与所有控制器进行通信。在 OpenFlow 协议中，控制器有 MASTER、SLAVE、EQUAL 这 3 种角色，其中 MASTER 表示主控制器，可以接收交换机发送的消息以及下发流表。SLAVE 表示从控制器，只能监听来自交换机的消息。EQUAL 表示默认状态。只有控制器可以修改角色，交换机没有修改角色的权限。一个交换机同一时刻只有一个 MASTER，当主控制器出现故障时，通过选举机制选出新的主控制器，并将该控制器的角色切换为 MASTER，从而保证了网络的可用性和可靠性。

2.4.2 信息共享以及一致性问题解决方法

Zookeeper 服务器负责多控制器的集群管理以及分布式协调，实现了多控制器间的信息共享与数据一致性。Zookeeper 使用树形结构数据单元 ZNode 来存储数据，服务器通过监控 ZNode 节点中数据的变化来感知事件，并做出响应。ZNode 节点关系如图 8 所示。

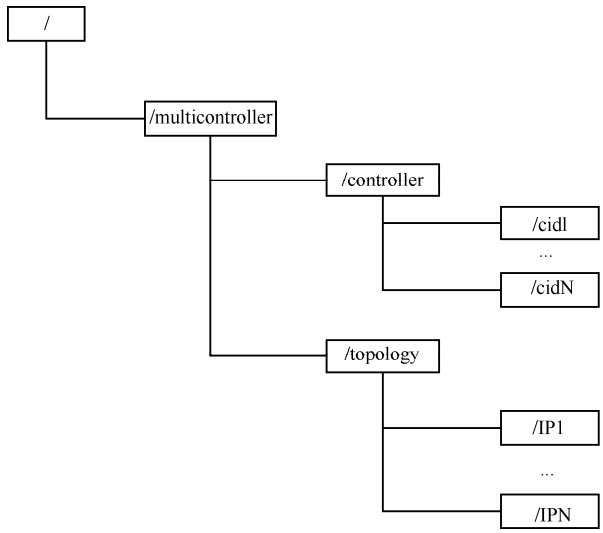

图 8 ZNode 节点关系

ZNode“/controller”及其子节点用于记录每个控制器的信息。“/cid1”至“/cidN”节点代表网络内的不同控制器，每个节点代表一台控制器。

ZNode“/topology”节点用于存储全局网络拓扑信息以及交换机连接信息，这 2 种信息是多控制器之间需要共享的信息。子节点“/IP”标识了不同的控制器，该节点下的数据代表一个控制器掌握的拓扑信息和设备连接信息。当网络中的拓扑或交换机连接状态发生变化时，主控制器首先负责将更新后的信息存储至每一个“/IP”节点内，Zookeeper 通过 Watcher 监听“/IP”节点内的数据变化，并通过 Watch 事件通知从控制器。随后从控制器访问 Zookeeper 服务器中自身对应的“IP”节点，获取新的全网拓扑信息以及交换机连接信息。上述方法能够使多控制器中的网络拓扑信息数据保持一致，并且当网络更新后，多控制器的数据仍处于一致状态。

2.4.3 容灾机制实现过程

当出现单点失效情况时，主控制器发送同步数

据包至所有从控制器，将自身单点失效的消息通知从控制器。随后集群内部所有控制器根据选举机制选出新的主控制器。选举出的主控制器将自己在交换机中的角色由 SLAVE 切换为 MASTER，其余控制器将角色设置为 SLAVE。此时选举出的主控制器成功接替原主控制器的工作，控制交换机中的数据转发。上述容灾机制实现了主从控制器之间的切换，解决了控制器单点失效问题。

3　方案整体设计

本文方案的主要流程如图 9 所示。

1) 当用户接入该 SDN 时，属性签名标识生成模块根据入网用户的身份属性，生成用户的属性集合。

2) 属性签名标识认证中心生成系统的公开参数与主密钥。

3) 属性签名验证模块生成访问控制树 T，并将 T 发送给认证中心。认证中心根据 T 生成 T 的公开参数。

4) 属性签名标识生成模块将用户的属性集合发给认证中心。认证中心根据属性集合为用户生成用户私钥。

5) 认证中心将用户私钥发送给属性签名标识生成模块。

6) 属性签名标识生成模块以用户用于签名的属性集、T 的公开参数和数据包传输的消息 m 为输入，计算出属性签名值。然后将属性签名值以及用户的签名属性放入数据包的 Option 字段中。

7) P4 转发设备解析数据包中的属性签名标识。采样控制模块根据设置的检测因子 θ 来判断当前数据包是否为采样检测数据包，若是，执行 8a)；否则执行 8b)。

8a) 待检测数据包被加载至数据包镜像模块，该模块将待检测数据包复制，并将镜像数据包发送至 P4 转发设备出端口，原数据包执行 8b)。

8b) P4 转发设备转发数据包至 OpenFlow 交换机，而 OpenFlow 交换机通过匹配流规则执行相应的数据包转发动作。

9) P4 转发设备将其出端口中的待检测数据包发送至控制器，并由控制器的属性签名验证模块接收。

10) 属性签名验证模块对用户的属性以及签名值进行验证。若验证通过，说明该数据包是有效数据包。若验证不通过，将数据包转发给异常告警模块。

11) 异常告警模块收到无效数据包后，根据 Ryu 控制器事件机制，向数据转发处理模块发送告警事件 EventWarning。

12) 数据转发处理模块监听到 EventWarning 事件后，下发相应的流规则给 OpenFlow 交换机，令其对于检测异常的数据流进行拦截。

13) 如果出现控制器单点失效情况时，根据控

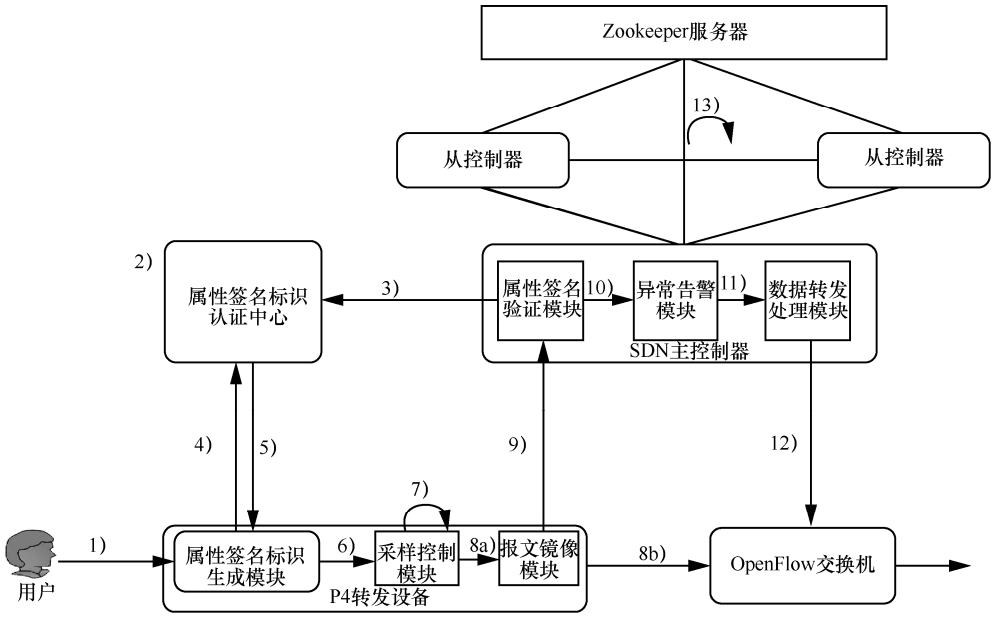

图 9　本文方案的主要流程

制器容灾机制使用从控制器接替当前主控制器的工作。

4 仿真实验及评估

4.1 实验设置

本实验采用一台 Windows Server2012 服务器作为属性签名标识认证中心，一台 Windows Server2012 服务器作为 Zookeeper 服务器，2 台 Windows 主机作为终端，一台虚拟机上安装 3 台 Ryu 控制器，另一台虚拟机安装一个 Bmv2 软件交换机以及 2 个 OpenFlow 软件交换机。控制器与交换机在 Mininet 环境下实现网络互联，服务器、终端以及虚拟机通过网线互联。图 10 显示了实验环境的拓扑结构。本实验主要测试该机制的检测准确率、数据包转发时延以及系统性能。

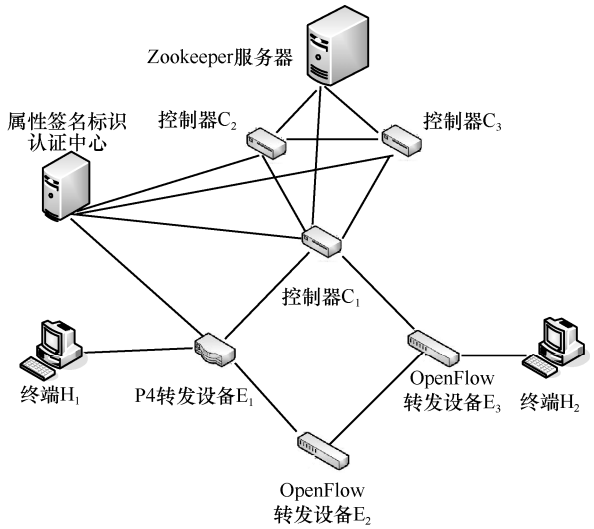

图 10 实验环境的拓扑结构

4.2 检测准确率

在终端 H_1 上使用发包脚本对网络进行模拟攻击，分别以 0.1 的概率发送 2 种数据流，一种数据流中伪造了属性签名验证字段，另一种数据流中篡改了数据包中的数据字段。不同数据流中数据包数量为 10 000 个，攻击各进行 100 次，实验结果取平均值，并以漏报率作为衡量系统检测准确率的实验指标。检测因子 θ 分别设置为 9、7、4。若数据包验证检测系统检测到异常数据包并下发流表，则记录为检测成功；若未检测到，则记录为漏报。不同检测因子下的漏报率如图 11 所示。

根据图 11 可知，对于相同的检测因子伪造验证字段与篡改数据字段的漏报率差别不大，原因是系统通过验证数据包中的属性签名来识别异常数据包，2 种攻击方式都能使属性签名验证失败。随着 θ 的减小，漏报率降低，这说明采样比越高，系统安全性越高。但同时较高的采样比会增加 CPU 的使用率，从而增加系统的开销。实验发现，现在检测因子分别设为 9、7、4 的情况下，控制器 CPU 平均使用率分别为 7.8%，10.3%以及 16.2%。综合考虑系统的安全性能以及 CPU 的利用率，检测率设置为 7 时，可以获得很好的检测准确率，且不需要较高的系统开销。

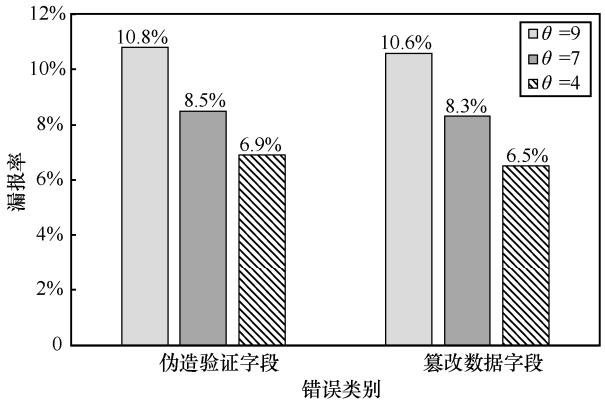

图 11 不同检测因子下的漏报率

4.3 数据包转发时延

通过编写自定义发包脚本程序模拟用户主机的行为，区分数据包携带属性签名标识（检测因子设为 7）与不携带属性签名标识 2 种情况，分别发送 10 000 个数据包进行测试。通过比较 2 种情况下的转发时延，分析添加属性签名标识后对交换机的转发行为的影响。采用 Wireshark 工具在该网络的出入口进行抓包，并计算数据包转发时延。本实验分别统计 20 次转发时延（其中每次的时延是由 10 次重复实验得出的平均值），结果如图 12 所示。

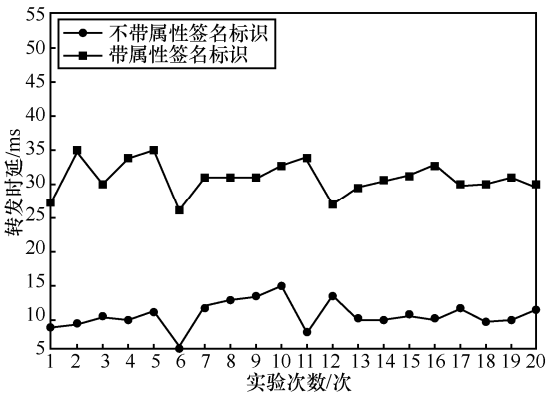

图 12 数据包转发时延

如图 12 可看出，正常情况下的转发时延的平均值为 10.75 ms，在加入属性签名标识后平均值为 30.95 ms，其平均转发时延增加了 20.20 ms，通过分析可知在本原型系统中，由于数据包中携带了属性签名标识（40 B），使传输的数据包要比正常的数据包大，在加之部分数据包还要作为采样检测数据包进行属性签名验证，而签名验证时又要与认证中心进行数据互传，导致该系统的转发时延增大，在增大的时延中用于属性签名以及签名验证的时间约占整个时延的 65.2%，所以签名算法的复杂度直接决定了转发时延。由于该方案在仿真环境下的平均转发时延为 30.95 ms，这表示控制器平均每秒处理 33 个数据流。根据 Stanford University 提供的实验数据[24]，一个拥有 300 台用户主机的网络每秒处理的数据流请求约在 30～40。因此该机制虽然由于引入属性签名验证而增大了转发时延，但是仍能满足一般中小型网络的通信需求。

4.4　网络吞吐量开销

在本实验中，首先设定不同的检测因子 θ（θ 值设为 9、7、4、3、1），然后分别测量吞吐量开销，并与正常情况进行对比。网络吞吐量对比如图 13 所示。

由图 13 可以看出，与未使用属性签名标识机制的正常网络吞吐量相比，在 θ 为 9 时，吞吐量下降 7.7%，随着 θ 的减少，吞吐量随之下降，当 θ=1 时，吞吐量降低了约 29.8%左右。因此可以看出网络吞吐量与 θ 成正比，θ 越大，网络吞吐量越大。

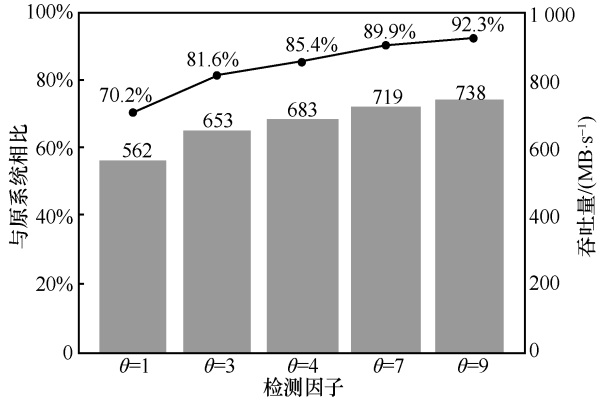

图 13　网络吞吐量对比

4.5　扩展实验

本节实验增加了数据平面上的用户节点，通过这种方式来测试当网络规模扩大时数据包转发时延以及网络吞吐量的变化情况。由于实验条件有限，系统共添加了 32 台主机作为用户主机，负责向网络中发送数据包。本文分别测试用户主机数为 1、2、4、8、16、32 时的数据包转发时延和网络吞吐量，扩展实验拓扑如图 14 所示。

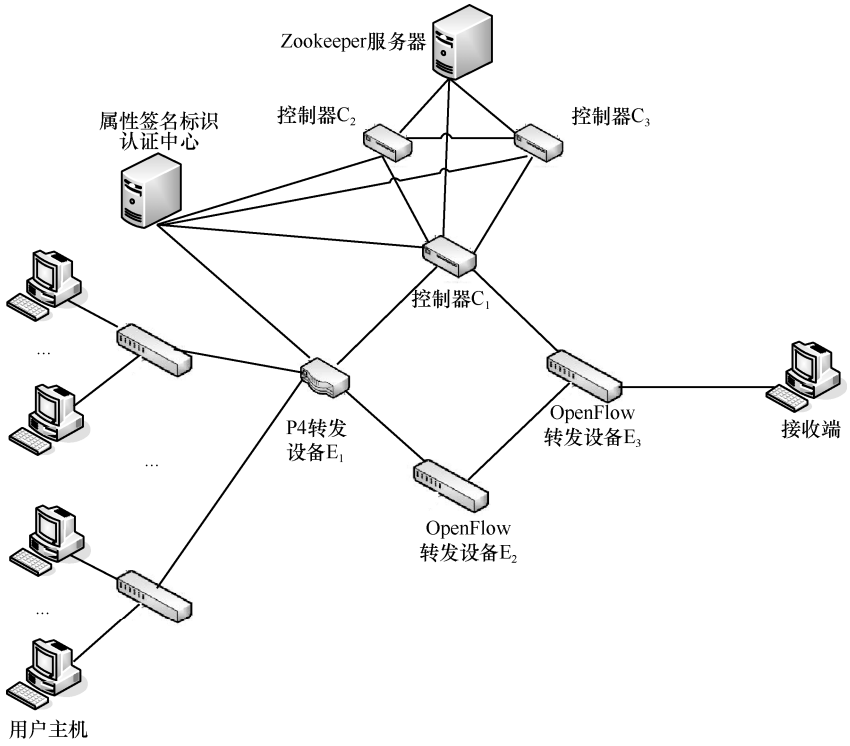

图 14　扩展实验拓扑

首先，本文在每个用户主机上发送 10 000 个数据包，并设定检测因子 θ 为 7，测量在不同的用户主机数量条件下该网络中数据包转发时延，并与未添加属性签名标识的网络的数据包转发时延进行对比，结果如图 15 所示。随着网络中主机数量的增多，转发时延不断增大。而随着用户数量的增多，添加了属性签名标识的网络的数据包转发时延的增长率要更高，但是仍属于较为平缓的增长，处于可接受范围内。

其次，区分添加属性签名标识与不添加属性签名标识 2 种情况分别测量在不同的用户主机数量情况下的网络吞吐量，如图 16 所示。可以看出，随着用户主机数量的增多，网络吞吐量下降，虽然添加了属性签名标识的网络吞吐量下降率较高，但总体来说下降幅度不大，仍处于可接受范围内。

综上所述，扩展网络规模会给网络的吞吐量以及数据包转发时延带来一定的影响，但通过实验结果的分析可知对于添加了数据转发验证机制的网络影响有限，网络开销并非成倍增长，仍处于可接受范围内。

4.6 机制对比

将本文的转发验证方案与已经提出的一些转发验证方案[8,10,18]进行对比，如表 1 所示。

本文采用基于用户属性的签名方案，而其他方案均是基于用户身份的，相较而言，本文方案更加细粒度地划分了用户身份特征，具有更强的灵活性和匿名性。本文方案与方案 3 采用了 P4 转发设备通过自定义匹配字段来完成对数据流的精确采样，相比于其他方案采用的 OpenFlow 匹配字段，能够更细粒度地对数据包进行控制。

在验证设备方面，本文方案和方案 1、方案 3 均采用了控制器作为验证设备，这样会增加控制器的负载，造成控制器单点失效等问题。本文方案通过构建多控制器架构有效避免了该问题。方案 2 使用了交换机作为验证设备，虽然能减轻控制器负担，但对交换机的安全性要求较高。

在验证开销方面，方案 1～方案 3 均通过验证 Hmac 消息摘要的方式来对数据流进行验证。而本方面采用属性签名技术，涉及双线性对运算等耗时计算，所以在时间开销上要大于其他 3 种方案。

在转发时延方面，方案 1 转发时延为 33.17 ms，方案 2 转发时延为 33.65 ms，方案 3 转发时延为 0.83 ms，本文方案的转发时延为 30.95 ms。可以看出，本文方案虽然验证开销较大，但整体转发时延并不比方案 1 和方案 2 大。这是由于本文方案采用采样检测的方法，对于未被采样的数据转发设备进行正常的匹配转发，且本文方案主要的时延来自验证开销，密钥传输次数较少，通信开销较低，故转发时延方面与方案 1 和方案 2 无较大差距。

在实现功能方面，4 种方案都可以实现检测伪造、篡改数据包的功能，而由于本文方案将用户的身份属性与用户发出的数据包进行了绑定，一旦发现用户在网络中的违规行为，可以利用属性签名标识追踪其真实身份，因此相比其他方案，实现了用户身份的可追踪性。

最后，本文分析了所有方案存在的局限性。本签名方案采用属性签名方法，所采用的通信方式是一对多，验证者只需验证被验证者是否满足访问控制结构即可。而其他方案采用基于身份的签名验证，采用的通信方式是一对一，为保证安全性采用“一次一密”的方式，这需要进行较频繁的密钥传输，增大了密钥通信的时间，此外还给密钥管理增加了难度。此外，方案 1 与方案 3 均使用单控制器架构，并使用控制器作为验证设备，这样会增大控制器的开销，容易出

表 1 不同方案特点比较

方案	签名方式及采样粒度	验证设备及开销	转发时延	实现功能	局限性分析
方案 1（文献[8]）	基于身份，OpenFlow 匹配字段	控制器，0.15 ms	33.17 ms（3 层树形结构，最多经 5 台交换机）	定位并检测伪造、篡改数据包	控制器单点失效、密钥通信开销大
方案 2（文献[10]）	基于身份，OpenFlow 匹配字段	交换机，时延未知	33.65 ms（4 层树形结构，最多经 7 台交换机）	检测伪造、篡改数据包	需要改变交换机内部构成，密钥通信开销大
方案 3（文献[18]）	基于身份，自定义匹配字段	控制器，0.19 ms	0.83 ms（自定义结构，3 台 OpenFlow 交换机和一台 P4 交换机）	检测伪造、篡改数据包	控制器单点失效、密钥通信开销大
本文方案	基于属性，自定义匹配字段	控制器，约 20.2 ms	30.95 ms（自定义结构，2 台 OpenFlow 交换机和一台 P4 交换机）	检测伪造、篡改数据包，用户身份可追踪	计算开销较大

现单点失效问题。而方案 2 将验证功能放在传统 OpenFlow 交换机上，实际上要在 OpenFlow 交换机上开发安全模块，不仅开发难度较大，可行性未知，而且很难维护和添加新功能。本文方案的不足在于，由于引入属性签名技术导致计算开销增大。

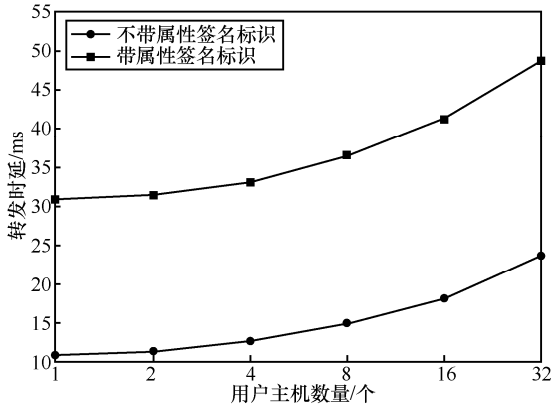

图 15　不同主机数量下的数据包转发时延

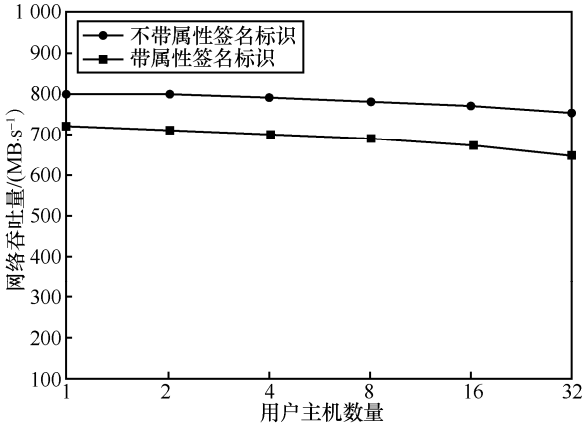

图 16　不同主机数量下的网络吞吐量

5　结束语

针对 SDN 中数据包缺乏有效的转发验证以及数据流缺乏精确控制方法的问题，本文提出一种基于属性签名标识的数据包转发验证方案。采用根据用户的属性集合进行签名，并在控制器上进行验证的方式，实现了数据包的安全转发验证功能。在 SDN 中利用 P4 转发设备，生成并解析属性签名标识，实现了对数据流的更细粒度的精准控制以及采样等目的。本文还设计了一种多控制器架构，解决了控制器单点失效问题。最后，在基于 Ryu 控制器和 Mininet 的实验环境中对该方案进行了实现。结果表明该方案能够有效检测出数据流被恶意篡改、伪造等异常攻击行为，虽引入了相对较高的转发时延，但仍处于可接受范围之内。针对验证功能给控制器带来的性能问题，未来的工作将研究在 P4 交换机上开发验证模块，将验证功能放置到交换机中进行，从而进一步提升控制器的性能。

参考文献：

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

[2] NUNES B A A, MENDONCA M, NGUYEN X N, et al. A survey of software-defined networking: past, present, and future of programmable networks[J]. IEEE Communications Surveys & Tutorials, 2014, 16(3): 1617-1634.

[3] 王蒙蒙, 刘建伟, 陈杰, 等. 软件定义网络: 安全模型、机制及研究进展[J]. 软件学报, 2016, 27(4): 969-992.
WANG M M, LIU J W, CHEN J, et al. Software defined networking: security model, threats and mechanism[J]. Journal of Software, 2016, 27(4): 969-992.

[4] GAO S, LI Z C, XIAO B, et al. Security threats in the data plane of software-defined networks[J]. IEEE Network, 2018, 32(4): 108-113.

[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.

[6] RANA D S, DHONDIYAL S A, CHAMOLI S K. Software defined networking (SDN) challenges, issues and solution[J]. International Journal of Computer Sciences and Engineering, 2019, 7(1): 884-889.

[7] GUPTA B B, PEREZ G M, AGRAWAL D P, et al. Handbook of computer networks and cyber security[M]. Cham: Springer International Publishing, 2020.

[8] 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.
WANG S Y, LI Q, ZHANG Y. LPV: lightweight packet forwarding verification in SDN[J]. Chinese Journal of Computers, 2019, 42(1): 176-189.

[9] SASAKI T, PAPPAS C, LEE T, et al. SDNsec: forwarding accountability for the SDN data plane[C]//2016 25th International Conference on Computer Communication and Networks. Piscataway: IEEE Press, 2016: 1-10.

[10] 秦晰, 唐国栋, 常朝稳, 等. 软件定义网络中基于密码标识的数据包转发验证机制[J]. 电子与信息学报, 2018, 40(9): 2042-2049.
QIN X, TANG G D, CHANG C W, 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.

[11] 冯登国, 陈成. 属性密码学研究[J]. 密码学报, 2014, 1(1): 1-12.
FENG D G, CHEN C. Research on attribute-based cryptography[J]. Journal of Cryptologic Research, 2014, 1(1): 1-12.

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

[13] BOSSHART P, GIBB G, KIM H S, et al. Forwarding metamorphosis: fast programmable match-action processing in hardware for SDN[C]//The ACM SIGCOMM 2013 Conference on SIGCOMM. New York: ACM Press, 2013: 99-110.

[14] 祝现威, 常朝稳, 朱智强, 等. 基于身份属性的 SDN 控制转发方法[J].

通信学报, 2019, 40(11): 1-18.
ZHU X W, CHANG C W, ZHU Z Q, et al. SDN control and forwarding method based on identity attribute[J]. Journal on Communications, 2019, 40(11): 1-18.

[15] KHADER D. Attribute based group signatures[J]. IACR Cryptology ePrint Archive, 2007, 2007: 159.

[16] 陈剑锋.基于属性签名方案的研究[D]. 广州: 中山大学, 2010.
CHEN J F. Research on attribute-based signatures[D]. Guangzhou: Sun Yat-Sen University, 2010.

[17] GOYAL V, PANDEY O, SAHAI A, et al. Attribute-based encryption for fine-grained access control of encrypted data[C]//The 13th ACM conference on Computer and Communications Security. New York: ACM Press, 2006: 89-98.

[18] 左志斌, 常朝稳, 祝现威. 一种基于数据平面可编程的软件定义网络数据包转发验证机制[J]. 电子与信息学报, 2020, 42(5): 1110-1117.
ZUO Z B, CHANG C W, ZHU X W. A software-defined networking packet forwarding verification mechanism based on programmable data plane[J]. Journal of Electronics & Information Technology, 2020, 42(5): 1110-1117.

[19] 林耘森箫, 毕军, 周禹, 等. 基于 P4 的可编程数据平面研究及其应用[J]. 计算机学报, 2019, 42(11): 2539-2560.
LIN Y, BI J, ZHOU Y, et al. Research and applications of programmable data plane based on P4[J]. Chinese Journal of Computers, 2019, 42(11): 2539-2560.

[20] YAZICI V, SUNAY M O, ERCAN A O. Controlling a software-defined network via distributed controllers[J]. arXiv Preprint, arXiv:1401.7651, 2014.

[21] 田心宁. 基于 Zookeeper 的 SDN 多控制器架构的研究与实现[D]. 兰州: 兰州大学, 2016.
TIAN X N. Design and implementation of multiple SDN controllers via zookeeper[D]. Lanzhou: Lanzhou University, 2016.

[22] HUNT P, KONAR M, JUNQUEIRA F P, et al. Zookeeper: wait-free coordination for internet-scale systems[C]//USENIX Annual Technical Conference. Berkeley: USENIX Association, 2010: 9.

[23] 陈世强. 基于多控制器的 SDN 一致性机制研究[D]. 北京: 北京理工大学, 2016.
CHEN S Q. Research of consistency mechanism based on multi controllers in software-defined network[D]. Beijing: Beijing Institute of Technology, 2016.

[24] CASADO M, FREEDMAN M J, PETTIT J, et al. Ethane: taking control of the enterprise[C]//The 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. New York: ACM Press, 2007: 27-31.

[作者简介]

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。

金建树（1992− ），男，辽宁锦州人，信息工程大学硕士生，主要研究方向为 SDN 安全、网络安全。

韩培胜（1978− ），男，河北黄骅人，博士，信息工程大学教授，主要研究方向为网络安全、可信计算。

祝现威（1991− ），男，河南虞城人，信息工程大学博士生，主要研究方向为 SDN 安全、网络安全、云计算安全。