doi:10.11959/j.issn.1000-436x.2015299

SDFAC：软件定义的流接入控制机制

王秀磊，张国敏，胡超，陈鸣，魏祥麟
（解放军理工大学 指挥信息系统学院，江苏 南京 210007）

摘 要：SDN 控制平面与数据平面分离的体系架构为实现细粒度的流管理以及灵活的中心化控制提供了基础。基于此，提出了一种软件定义的流接入控制机制 SDFAC。首先从流的粒度对接入控制进行建模分析，给出了实现细粒度流接入控制所需要满足的条件；其次描述了 SDFAC 的基本框架和工作原理并设计了一种支持 SDFAC 功能的流鉴别协议；最后基于原型系统验证了 SDFAC 的可行性和可用性。

关键词：接入控制机制；软件定义网络；流鉴别协议；OpenFlow；安全
中图分类号：TP393 **文献标识码**：A

SDFAC: software defined flow access control mechanism

WANG Xiu-lei, ZHANG Guo-min, HU Chao, CHEN Ming, WEI Xiang-lin
(College of Command Information System, PLA University of Science and Technology, Nanjing 210007, China)

Abstract: The software defined networking paradigm decouples control plane from data plane, offering flexible centralized control and fine grain flow management. Based on these advantages, a novel software defined access control mechanism SDFAC was proposed. Firstly, an analysis of the access control model was given from the flow granularity, and the precondition for the fine-grained access control was deduced from the model. Secondly, the framework and basic working process of the SDFAC was described. The flow authentication protocol was designed to support the function of SDFAC. Finally, the experiment results prove the feasibility and availability of SDFAC.

Key words: access control mechanism; software defined networking; flow authentication protocol; OpenFlow; security

1 引言

功能完备的接入控制机制应该具备以下 3 点：1) 防止非法用户访问网络服务；2) 为合法用户授予适当的权限，使其能够访问受保护的服务或资源；3) 防止合法用户访问未被授权的网络服务[1]。受限于 TCP/IP 网络分布式控制架构及端到端原则，当前接入控制机制存在着鉴别粒度粗、鉴别策略适应性差和部署成本高的问题[2]。

软件定义网络(SDN, software defined networking)将网络控制平面与数据平面进行解耦合[3]，集中式可编程控制的体系架构为未来园区网、企业网以及数据中心网络等重要边缘网络的安全创新提供了新的思路[4~7]。因此，本文提出了一种软件定义的接入控制机制(SDFAC, software defined flow access control)。

本文分析了现有接入控制机制的不足，指出功能完备接入控制既要保证用户身份真实，还应保证其行为可信；从流的角度对接入控制进行建模，推导出细粒度接入控制所需条件；基于 SDN 体系架构，描述了 SDFAC 框架和工作流程，设计了流鉴别协议。

2 相关工作

IEEE802.1x 是一种基于端口的网络接入方案

收稿日期：2015-10-24
基金项目：国家重点基础研究发展计划基金资助项目(“973”计划)(2012CB315806); 国家自然科学基金资助项目(61379149, 61402521); 江苏省自然科学基金资助项目(BK20140070, BK20140068); 江苏省未来网络科技计划项目(BY2013095-1-06)
Foundation Items: The National Basic Research Program of China(973 Program) (2012CB315806); The National Natural Science Foundation of China (61379149, 61402521); The Natural Science Foundation of Jiangsu Province (BK20140070, BK20140068); Jiangsu Future Network Innovation Institute Research Project on Future Networks (BY2013095-1-06)

(PNAC, port-based network access control)协议[8]，是TNC(trusted network connected)[9]的典型方案之一，其保持了 IP 网络的无连接特性，在链路层进行用户身份认证，不需要复杂的网络配置，主要功能模块都在端系统实现(接入端代理、RADIUS 服务器)，上述特点使得其在因特网中得到广泛应用。虽然 TNC 从一定程度上保证了接入主机的可信性，但是受限于当前网络体系结构，其存在如下局限性：1) 客户端/服务器模式下，它只强调了对终端的鉴别，而没有对相关服务进行评估；2) 认证粒度不足，仅仅基于端口绑定对用户身份进行鉴别，无法确保用户服务请求的可信；3) 决策信息有限且策略更新过程复杂，同时策略更新时需要同时对控制平面和数据平面功能修改，更新部署复杂性高，需要专门配置。

权限管理是接入控制的关键。基于角色的访问控制(RBAC, role-based access control)[1]通过引入角色的概念建立了分层的间接映射，有效降低了权限管理复杂度，但是其仅仅鉴别用户身份而没有对用户行为进行鉴别，同时 RBAC 采用了预先静态配置的方式为角色授权，而在用户实际使用权限的过程中并不进行监管和调整，策略配置的灵活性较差。Chakraborty 等[10]在 RBAC 基础上融入了信任度，提出了 TrustBAC 模型，根据用户身份属性与信任度动态的决定其角色，很好地克服了 RBAC 的缺点。

SANE[11]和 Ethane[12]最先研究了 SDN 在接入控制中的应用。Ethane 在 SANE 的基础上扩展了功能，将安全管理策略添加到网络管理中，扩展了控制器功能，实现更细粒度的决策，在此基础上，实现基于流的接入控制。文献[2]提出一种强的接入控制机制，该机制基于 SDN 架构进行构建，通过在控制平面将用户的名字(用户的身份，地址)以及设备与应用及端口明确的绑定，实现强制的 AAA(authentication, authorization and accounting)接入控制。文献[7]提出了一种基于流的接入控制方案 FlowNAC，该机制根据用户的等级为其分配不同的服务列表，利用 SDN 体系架构，动态地在数据平面识别数据流，在控制平面设置不同的控制粒度。文献[13]指出各个用户共享数据平面的资源，因此在使用资源时就会出现竞争，接入控制必须能够保证网络资源不被非授权用户访问。文献[14]提出了一种基于流的资源访问控制机制，通过分析数据平面流表项识别出相关流，在控制平面对不同网络流进行授权，实现对共享资源的优化。

3　基于 SDN 的流接入控制机制

用户访问网络服务的行为可以形式化地描述为一条流[7,14]，基于此前提，本节首先通过建模分析了流接入控制所需要满足的必要条件。

3.1　模型和定义

设任意网络管理域 D，$U_D=\{user_1, user_2, user_3 \cdots\}$ 表示用户集合，$user_i=\{uid_i, uattr_i^1, uattr_i^2, uattr_i^3, \cdots, trustvalue_i\}$，$uid_i$ 表示全局唯一标识符，$uattr_i^j$ 表示 $user_i$ 的第 j 条身份属性，$trustvalue_i$ 表示 $user_i$ 的信任度。$uattr_i^j$ 以及 $trustvalue_i$ 决定了 $user_i$ 的角色及其所能获取的网络服务。$S_D=\{service_1, service_2, service_3, \cdots, service_n\}$ 表示各种网络服务集合，$service_i$ 由服务标识 sid_i 与服务属性组成，即 $service_i=(sid_i, sattr_i^1, sattr_i^2, sattr_i^3, \cdots)$，$R_D=\{role_1, role_2, role_3, \cdots, role_n\}$ 表示系统角色集合，用户所归属的角色决定了其对系统资源的操作权限。$IP_D=\{ip_1, ip_2, ip_3 \cdots\}$ 表示 D 内所有合法的 IP 地址集合。D 中所有主机通过网络接口设备(网卡)接入网络，由于每个设备都具有唯一的 MAC 地址，因此所有接入 D 的网络设备集合可以表示为 $MAC_D=\{mac_1, mac_2, mac_3, \cdots\}$。$D$ 中所有交换设备(交换机、路由器)集合为 $SW_D=\{sw_1, sw_2, sw_3, \cdots\}$，$\forall sw_i \in SW_D$，其端口集合为 $Port_i=\{port_i^1, port_i^2, port_i^3, \cdots\}$，$D$ 内所有端口集合可以表示为 $PORT_D=\bigcup_i^{|sw_D|} Port_i=\{port_1, port_2, port_3, \cdots\}$，IP 数据分组抽象为 8 元组，$packet=(seqno, smac, dmac, sip, dip, sport, dport, appinfo)$，对应字段分别表示分组序号、源 mac、目的 mac、源 IP、目的 IP、源端口号、目的端口号以及应用层关键信息。由设备 mac_j 发出的分组集合可以表示为 $Packet_{mac_j}=\bigcup_{i=1}^{k} packet_{mac_j}^i$，$D$ 内所有主机发出的分组集合为 $Packet_D=\bigcup_{i=1}^{|MAC_D|} Packet_{mac_i}$，网络数据流表示为 7 元组 $flow_i=(smac, dmac, sip, dip, sport, dport, appinfo)$。

在给出模型分析之前，给出 2 个基本假设和 6 条定义。

假设 1　对于任意 $user_i$，其标识符 uid_i 是全局唯一且难以伪造的。

假设 2　任意 $port_i \in PORT_D$，其位置信息真实可信，任意 $port_i$ 与主机设备直接相连，不存在串联

其他设备情况。

定义 1 主机网络设备可信。$\forall mac_i \in MAC_D$，$\exists MAC\_PORT_D = \{(mac_i, port_j) | (mac_i, port_j) \in MAC_D \otimes PORT_D, \forall (mac_i, port_j), (mac_m, port_n)$，如果 $mac_i = mac_m$，则 $port_j = port_n$，且如果 $port_j = port_n$，则 $mac_i = mac_m\}$，符号“$X \otimes Y$”表示从集合 X 到集合 Y 的笛卡尔积。

定义 2 主机 IP 可信。$\forall ip_i \in IP_D$，$\exists IP\_MAC_D = \{(ip_i, mac_j) | (ip_i, mac_j) \in IP_D \otimes MAC_D, \forall (ip_i, mac_j), (ip_m, mac_n)$，如果 $ip_i = ip_m$，则 $mac_j = mac_n$，且如果 $mac_j = mac_n$，则 $ip_i = ip_m\}$。

定义 3 用户身份可信。在满足定义 1 的前提下，$\forall\ user_i \in U_D$，如果存在可信映射 $IP_UID_D = \{(ip_j, uid_i) \mid (ip_j, uid_i) \in IP_D \otimes UID_D, \forall (ip_j, uid_i) \in \{IP_D \Rightarrow UID_D\}\}$则可以保证通信用户身份可信，符号“$X \Rightarrow Y$”表示从集合 X 到集合 Y 的单射。

定义 4 可信服务映射。不同的用户会分配不同的角色。设 $\varphi(user_i)$表示网络“用户－角色”映射函数，$\rho(role_i)$表示网络“角色－服务”映射函数，由 ρ 和 φ 所确定的真实可信映射 $UID_SID_D = \{(uid_i, sid_j) | (uid_i, sid_j) \in UID_D \otimes S_D\}$称为可信服务映射。

定义 5 数据分组可信。分组可信分为 2 部分。首先数据分组所携带的源 IP 地址与其发送者所使用的 IP 地址一致且真实有效，形成主机发送数据分组集合与用户身份标识集合可信映射，即$\exists PACKET_UID_D = \{(Packet_{mac_i}, uid_i) | Packet_{mac_i} \in Packet_D, uid_i \in UID_D\}, \forall packet_j, (packet_j \rightarrow srcip, uid_i) \in IP_UID_D$。其次分组所对应的服务与系统定义的服务映射一致，即网络中的任意分组 $packet(seqno, smac, dmac, sip, dip, sport, dport, appinfo)$一定属于某类服务，设 $\delta(packet)$表示分组到服务的映射函数，$(Packet_{mac_i}, \varphi^{-1}(\rho^{-1}(\delta(Packet_{mac_i}))) \in PACKET_UID_D$。

定义 6 数据流可信。网络中任意一条流 $flow_i$ 的源 IP 地址、源 MAC 地址和源端口与实际发送该流的主机网卡设备 MAC、端口和 IP 信息一致。即 $\forall flow_i$，$\exists (flow_i \rightarrow srcmac, port_j) \in MAC\_PORT_D$，$(flow_i \rightarrow srcip, uid_i) \in IP\_UID_D$。

3.2 形式化分析

设存在可信端口集合 $PORT_D$，可信映射 MAC_PORT_D、IP_MAC_D、UID_SID_D 以及 $UID\_MAC\_IP\_PORT_D$，下列形式化分析，证明上述前提条件是实现完备接入控制的必要条件。

定理 1 保证 D 内所有分组可信，则能够保证 D 内数据流可信。

证明 网络 D 中任意一条流 $flow_i$ 是若干有序分组组成的序列，即 $flow_i = [packet_i^1, packet_i^2, \cdots, packet_i^n]$，$packet_i^j = (j, flow_i \rightarrow smac, flow_i \rightarrow dmac, flow_i \rightarrow sip, flow_i \rightarrow dip, flow_i \rightarrow sport, flow_i \rightarrow dport)$。如果能够确保 $packet_i^j$ 真实可信，那么相应的 $flow_i \rightarrow smac, flow_i \rightarrow dmac, flow_i \rightarrow sip, flow_i \rightarrow dip, flow_i \rightarrow sport, flow_i \rightarrow dport$ 字段分别可信，从而确保了 $flow_i$ 的真实性。

定理 2 可信映射 MAC_PORT_D 保证了网卡设备的真实可信。

证明 1) 充分性。由于$\forall port_i \in PORT_D$ 真实可信，同时 MAC_PORT_D 保证了网卡设备与交换设备端口真实一一映射信息，即$\forall mac_j \in MAC_D$，使得式(1)成立。

$$\begin{cases} f(mac_j, MAC\_PORT_D) \rightarrow port_j = port_i \\ f(port_i, MAC\_PORT_D) \rightarrow mac_i = mac_j \end{cases} \tag{1}$$

因此集合 MAC_D 真实可信，其中，$f(a, B)$表示在多元组集合 B 中查找含有 a 元素的元组项。

2)必要性。若主机网卡设备与交换机端口不存在一一映射，即仅$\exists f(port_i, MAC_PORT_D) \rightarrow mac_i = mac_j$，则只能证明集合$\{f(port_i, MAC_PORT_D) \rightarrow mac_i\}$可信，因为$\{f(port_i, MAC_PORT_D) \rightarrow mac_i\} \subseteq MAC_D$，故无法证明全部主机可信；同理，若仅存在 $f(mac_j, MAC\_PORT_D) \rightarrow port_j = port_i$，存在多台主机接入同一端口的情况，形成网卡设备集合到端口集合的单射关系，如式(2)所示。

$$f(mac_j, MAC\_PORT_D) \rightarrow port_j = f(mac_i, MAC\_PORT_D) \rightarrow port_i (mac_j \neq mac_i) \tag{2}$$

即无法通过位置信息鉴别主机。

定理 3 可信映射 IP_MAC_D 和 MAC_PORT_D 保证了网络管理域 D 内 IP 地址的可信。

证明 1) 充分性。定理 2 已经证明，可信的映射集合 MAC_PORT_D 保证了网卡设备真实可信，IP_MAC_D 保证了 IP 地址的可信，即对$\forall ip_i$，式(3)成立。

$$\begin{cases} f(ip_i, IP\_MAC_D) \rightarrow mac_i = mac_j \\ f(mac_j, IP\_MAC_D) \rightarrow ip_j = ip_i \end{cases} \tag{3}$$

2) 必要性。

若仅$\exists f(ip_i, IP_MAC_D) \rightarrow mac_i = mac_j$，则只能

说明任意一个 IP 地址都与一个接入网络设备映射，有可能出现式(2)多个IP地址对应一个MAC地址的情况，在式(4)中，同一个 MAC 地址映射下的多个 IP 地址可以相互伪造，存在安全威胁。

$$f(ip_i, IP\_MAC_D) \rightarrow mac_i = f(ip_j, IP\_MAC_D) \rightarrow mac_j \ (ip_i \neq ip_j) \tag{4}$$

同理，若仅存在 $f(mac_j, IP\_MAC_D) \rightarrow ip_j$，则只能说明$\{f(mac_j, IP_MAC_D) \rightarrow ip_j\}$可信，由于$\{f(mac_j, IP_MAC_D) \rightarrow ip_j\} \subseteq IP_D$，不能保证所有 IP 地址可信。

定理 4　$UID_MAC_IP_PORT_D$与 UID_SID_D保证了数据分组的可鉴别与可控。

证明　由定理 2 和定理 3 可知 MAC_PORT_D 与 IP_MAC_D 确保了 D 内 MAC、IP 的真实性设 $MAC\_IP\_PORT_D$ 表示真实一一映射关系集合，$UID\_MAC\_IP\_PORT_D$ 表示用户标识与 $MAC\_IP\_PORT_D$ 中元素的映射关系。如果使得式(5)成立，则保证了用户的真实性。

$$\begin{cases} f(uid_i, UID\_MAC\_IP\_PORT_D) \rightarrow \\ (ip, mac, port)_i = (ip, mac, port)_i \\ f((ip, mac, port)_j, \\ UID\_MAC\_IP\_PORT_D \rightarrow uid_j = uid_i \end{cases} \tag{5}$$

若仅$\exists$ $f(uid_i, UID_IP_MAC_PORT_D) \rightarrow (ip, mac, port)_i = (ip, mac, port)_j$则只能说明任意一个用户都与一个 IP 地址、网卡设备、接入位置映射，但是可能存在着多个用户同时对应一个接入位置映射的情况，即 $f(uid_i, UID\_MAC\_IP\_PORT_D) \rightarrow (ip, mac, port)_i = f(uid_j, UID\_MAC\_IP\_PORT_D) \rightarrow (ip, mac, port)_j$。若仅存在 $f((ip, mac, port)_j, UID\_MAC\_IP\_PORT_D) \rightarrow uid_j$，则只能说明$\{f((ip, mac, port)_j, UID_MAC_IP_PORT_D) \rightarrow uid_j\} \subseteq UID_D$是可信的，即不能保证全部用户的可信性。

设 $packet_i^1$ 表示 $flow_i$ 的第 1 个分组，网络利用 $\delta(packet_i^1)$计算分组归属服务，由于 D 保证了 UID_SID_D 的私密性，若该分组为虚假分组，则 $(Packet_{mac_i}, \varphi^{-1}(\rho^{-1}(\delta(Packet_{mac_i}))))\in PACKET\_UID_D$，从而对用户服务请求进行了认证。

通过上述的证明可以看到，如果网络管理员能够获取管理域 D 内所有交换设备的真实端口信息以及可信的映射关系：MAC_PORT_D、IP_MAC_D、UID_SID_D、$UID\_MAC\_IP\_PORT_D$，则可以实现完备的接入控制。

3.3　SDFAC 架构及工作原理

基于模型本文提出了一种新型的软件定义接入控制机制 SDFAC，如图 1 所示。

系统主要功能模块位于控制器，包括流鉴别应用程序、IP 地址管理软件以及设备管理及拓扑发现模块。其中，流鉴别应用程序基于流鉴别协议(FAP, flow authentication protocol)与端主机的鉴别代理进行通信；IP 管理应用通过调用 DHCP 等应用，负责管理域内 IP 地址；设备管理和拓扑发现应用通过调用南向接口相关功能获得底层物理设备的端口、名称、队列等配置信息及数据平面拓扑信息。权威认证服务器(AAS, authority authentication server)通过安全通道(如 SSL)与控制器进行信息交互。网络管理员通过策略配置接口为认证服务器设置相应的服务权限配置策略，根据底层用户的身份信息和信

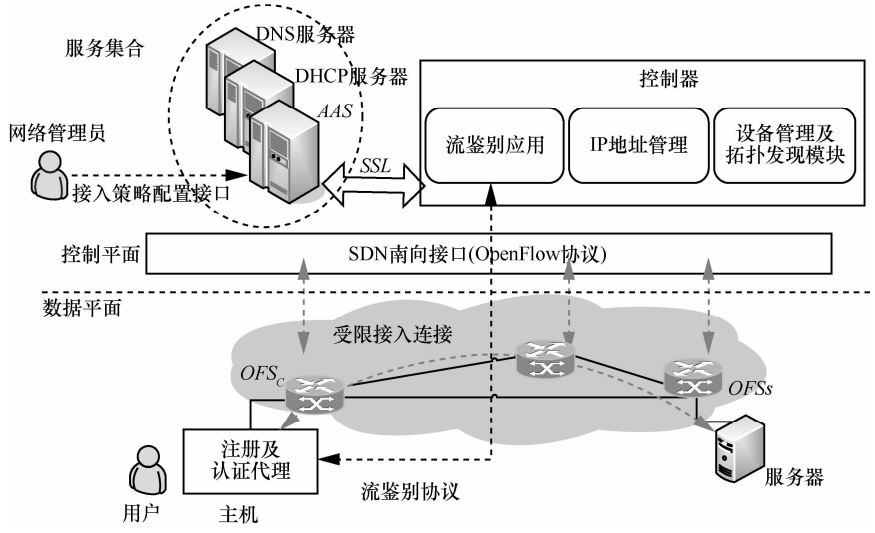


图 1　SDFAC 总体架构

誉值为其分配相应角色。

对于任意的用户，在 SDFAC 架构下访问服务需要经历 2 个基本阶段。

step1 用户注册。对于系统中用户 $user_A$，必须在 *AAS* 中进行预注册，由 *AAS* 为其生成全局唯一且难以伪造的身份标识符 uid_A。注册时，$user_A$ 提供必要的身份信息以及自身公钥，由 *AAS* 为其生成公钥证书并将相应信息存储在数据库中，同时 *AAS* 能够根据用户身份属性和信誉值为其分配相应角色[9,10]，从而限定 $user_A$ 能够访问的网络服务。

$user_A$ 的在线注册过程包含 5 个基本过程，如图 2 所示。

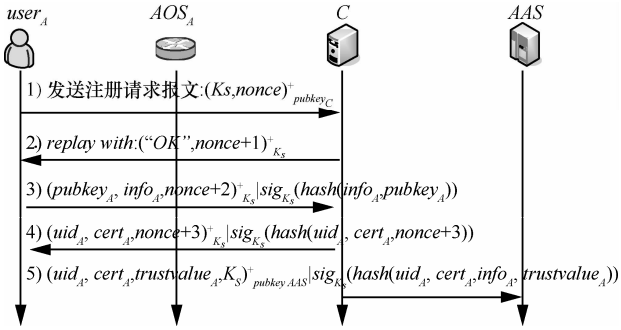


图 2 用户在线注册过程

1) $user_A$ 在终端 $host_A$ 运行注册程序，发送注册请求至接入 OpenFlow[15]交换机(AOS, access openflow switch)，注册请求报文包含由控制器 *C* 公钥 $pubkey_C$ 加密的用户临时会话对称密钥 K_s 以及防止重放攻击的不重数 *nonce*: $(K_s, nonce)^{+}_{pubkey_C}$。

2) 由于 AOS_A 无法查找到转发流表项，注册请求将被发送至 *C*。*C* 利用私钥解密注册请求报文，获取对称密钥 K_s，向 $user_A$ 发送注册成功报文：$(\text{“}OK\text{”}, nonce+1)^{+}_{K_S}$，通知用户已经成功获取对称密钥。

3) $user_A$ 向 *C* 发送自身身份信息 $info_A$、公钥 $pubkey_A$ 进行注册。为保证消息的完整性，对消息散列值进行签名：$(pubkey_A, info_A, nonce+2)^{+}_{K_S}|sig_{K_S}(hash(info_A, pubkey_A))$。

4) *C* 利用协商好的对称密钥解密$(pubkey_A, info_A, nonce+2)^{+}_{K_s}$，利用 $pubkey_A$ 及 $info_A$ 计算身份标识符 uid_A[16]及公钥 $cert_A$，将$(uid_A, cert_A, nonce+1)^{+}_{K_S}|sig_{K_S}(hash(uid_A, cert_A, nonce+3))$返回 $user_A$。

5) *C* 利用文献[10]的方法计算用户信誉值 $trustvalue_A$，结合 uid_A，$cert_A$ 与 $info_A$ 发送至 *AAS*：$(uid_A, cert_A, info_A, trustvalue_A, K_s)^{+}_{pubkey\,AAS}|sig_{K_S}(hash(uid_A, cert_A, info_A, trustvalue_A, K_s))$，*AAS* 解密并将相关信息存储至数据库。

step2 流鉴别。基于 step1 的注册信息，控制器完成用户身份、主机信息以及服务信息的鉴别。该阶段分为 2 个子过程，用户身份鉴别与服务鉴别。如图 3 所示，该阶段共分为 6 个基本过程。

1) $user_A$ 向控制器发起认证请求。请求分组包含用户注册过程中所获取 uid_A 以及鉴别用户身份信息 $info_A$。客户端程序将 $uid_A|(uid_A, info_A)^{+}_{pubkey_C}$ 发送至 *C*。

2) *C* 成功接收来自 AOS_A 端口 $Port_A$ 的认证请求报文，为了防止恶意用户利用认证请求报文发起恶意 DDoS 攻击，将在控制器安装默认流表，并在 *T*

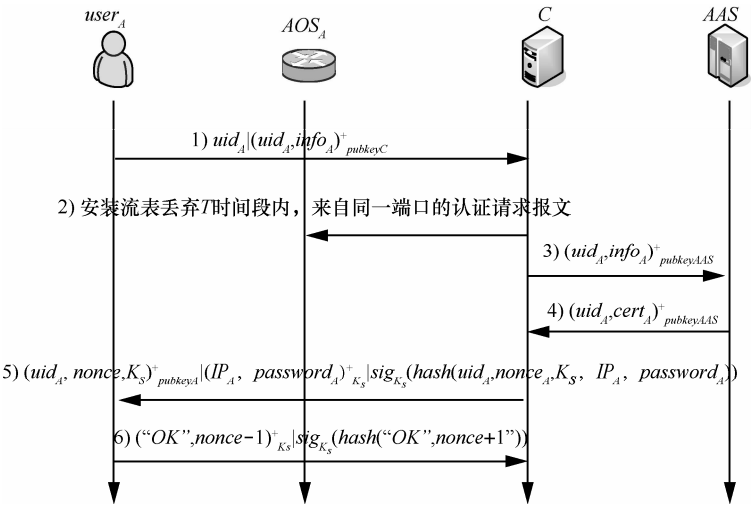


图 3 用户身份鉴别

时间段内丢弃来自端口 $Port_A$ 的认证报文。

3) C 利用私钥解密 $(uid_A, info_A)^{+}_{pubkey_C}$，获取 uid_A，通过对比判断该消息的真实性。若通过验证则通过安全信道向 AAS 发送报文 $(uid_A, info_A)^{+}_{pubkey_{AAS}}$ 获取 $user_A$ 的公钥信息。

4) AAS 解密 $(uid_A, info_A)^{+}_{pubkey_{AAS}}$，利用 uid_A 和 $info_A$ 在数据库中查找注册信息，获取 $user_A$ 的公钥证书 $cert_A$，并将该信息反馈给 C。

5) C 获得 $cert_A$ 后，调用 DHCP 服务为 $host_A$ 分配临时 IP 地址，同时为其分配对称密钥 K_s 和登录口令 $password$。利用对称密钥 K_s 进行加密 $(IP_A, password_A)^{+}_{K_s}$，利用用户公钥 $pubkey_A$ 进行加密 $(uid_A, nonce, K_s)^{+}_{pubkey_A}$，同时为了确保信息的完整性利用 $pubkey_A$ 对重要信息进行散列值签名 $sig_{K_S}(hash(uid_A, nonce, K_s, IP_A, password_A))$，$C$ 将上述信息返回给 $user_A$。同时 C 会建立 uid_A、接入端口 $Port_A$、$host_A$ 标识的可信映射，并临时存储在数据库。

6) $user$ 利用自身私钥解密 $(uid_A, nonce, K_s)^{+}_{pubkey_A}$ 得到 K_s，利用 K_s 解密 $(IP_A, password_A)^{+}_{K_S}$ 得到 IP_A 和 $password_A$，重新计算 $sig_{K_S}(hash(uid_A, nonce, K_s, IP_A, password_A))$ 检验数据完整性。若通过鉴别，将 IP_A 配置为主机 IP 地址并存储临时口令 $password_A$。

在完成用户身份鉴别和主机配置的基础上，图 4 给出了流鉴别的基本流程。

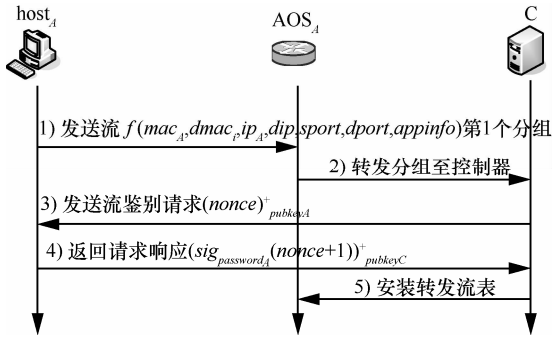


图 4　流鉴别流程

1) $user_A$ 经 $host_A$ 发起的服务请求流 f (mac_A, $dmac$, ip_A, dip, $sport$, $dport$, $appinfo$)的第一个分组发送至 AOS_A。

2) AOS_A 将 f (mac_A, $dmac$, ip_A, dip, $sport$, $dport$, $appinfo$)信息转发至控制器 C。

3) C 解析报文得到 mac_A, ip_A, AOS_A 标识符、$appinfo$ 以及该分组的入端口 $Port_A$, C 通过（mac_A, IP_A, AOS_A, $Port_A$）查询数据库获取与其关联的 uid_A、公钥证书 $cert_A$。C 向 $user_A$ 发起流鉴别挑战请求报文 $(nonce)^{+}_{pubkey_A}$。

4) 运行在 $host_A$ 上的鉴别代理接收挑战，利用自身私钥解密，得到随机数。随后，利用之前身份鉴别所协商的临时密钥 $password_A$ 对($nonce$+1)签名，并将签名信息利用控制器公钥加密 $(sig_{passwordA}(nonce+1))^{+}_{pubkey_C}$ 并返回 C。

5) C 利用自身私钥解密($sig_{password_A}$(nonce + 1))$^{+}_{pubkey_C}$，基于分组的返回端口查找 $PORT\_UID$ 映射获取对应的临时密钥 $password_A$ 对($nonce$ + 1)重新进行计算，若匹配，则下发相应流表，否则决绝连接。

3.4　流鉴别协设计

通过对目前广泛使用的 802.1x 协议进行修改，设计了一种支持上述过程的流鉴别协议 FAP(flow authentication protocol)。

1) 流鉴别协议帧格式，如图 5 所示。

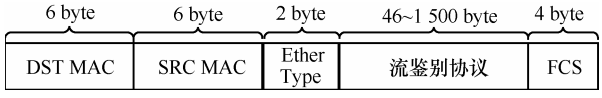


图 5　流鉴别协议帧结构

标准的以太网帧必须具备目的MAC地址，DST MAC 和以太网帧类型，Ether Type，在流鉴别协议运行时，主机客户端程序无法确定目的主机 MAC 地址，因此采用缺省组播地址 01-80-C2-00-00-03，帧类型区别于 802.1x 定义为 0X888F。流鉴别协议承载于帧数据部分。由于目前以太网帧数据最大长度设置为 1518 byte，因此鉴别协议数据最大长度为 1 500 byte。

2) 流鉴别协议格式，如图 6 所示。

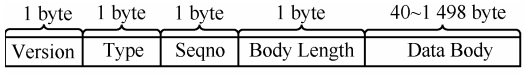


图 6　流鉴别协议格式

该协议的每个字段的语义如下。

Version：指明当前协议的版本号。

Type：类型字段。SDFAC 工作过程包含了主机

鉴别、用户身份鉴别以及流鉴别 3 个阶段，该字段主要用来说明当前数据帧所处的阶段：00 表示注册，01 表示用户身份鉴别，02 表示流鉴别。

Seqno：序号字段。每个阶段由多次有序交互的分组构成，该字段表明了当前分组的序号，通过该字段可以防止分组的丢失和失序。

Body Length：数据字段长度。该字段用来指明后续数据字段的长度。

Data Body：数据字段。根据不同的鉴别阶段，承载相应的数据。

4 性能评估

为了实际验证 SDFAC 的可行性和可用性，本文搭建了其原型系统，如图 7 所示，并基于该系统进行了测试。

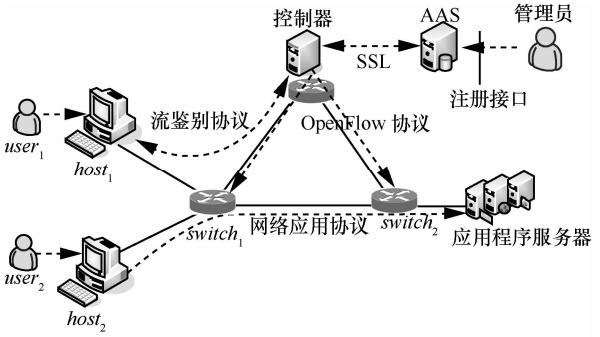


图 7 SDFAC 原型系统

该实验环境主要由 2 台主机、1 台应用服务器、1 台 POX [注1]控制器、3 台 OpenFlow 交换机和 1 台权威认证服务器组成。其中，OpenFlow 交换机由运行 Open Vswitch [注2](v.1.7.1)软件的主机充当，POX 控制器通过 OpenFlow 协议与交换机交互。POX 控制器上运行了由 Python 语言开发的鉴别程序；$host_1$ 和 $host_2$ 分别运行了由 C++语言开发的鉴别代理；应用服务器上运行了典型的网络服务，包括 Web 服务、E-mail 服务、FTP 等。

实验 1 SDFAC 的可行性

实验 1 的过程分为 4 个阶段。1) $user_1$ 和 $user_2$ 向 AAS 注册其身份和可用服务。控制器从 AAS 获取自身公钥证书、注册用户公钥证书及其角色信息；$host_1$ 和 $host_2$ 保存相应的私钥文件、公钥证书及控制器公钥证书。2) 实验开始时，设定 $user_1$ 服务列表包含 Web 服务和电子邮件服务，而 $user_2$ 服务列表为空，并且 $user_1$ 和 $user_2$ 尝试使用 Web 服务。3) 实验 60 s 后，网络管理员登陆到 AAS 数据库中为 $user_1$ 添加 Web 服务访问权限，实验 120 s 后网络管理员在 AAS 数据库中取消 $user_2$ 的 Web 服务权限，同时控制器删除数据平面 $user_2$ 中的 Web 服务转发流表项。4) 实验 140 s 后，$user_1$ 启动第 2 个服务电子邮件。

在实验过程中，$host_1$ 和 $host_2$ 访问存储在应用服务器中大小为 1 GB 的文件。$host_1$ 和 $host_2$ 的密钥长度为 1 024 bit。利用 Tcpdump [注3]等工具测量了每个主机上的 Web 和 E-mail 应用的的平均吞吐量变化情况，实验结果如图 8 所示。

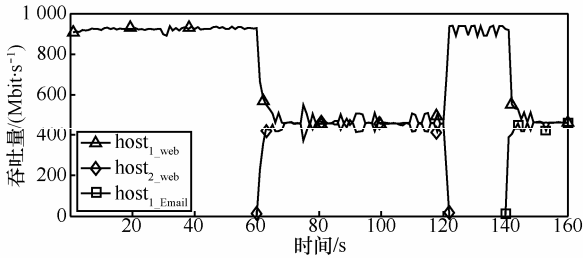


图 8 对不同用户不同服务请求的接入控制过程

测量结果及分析。从图 8 可见，Web 请求触发的流鉴别过程能够使 $host_1$ 顺利通过身份和流鉴别，从而使 $host_1$ 的 Web 服务能够得以执行。另一方面，尽管 $user_2$ 的身份信息存控制器中，但由于其服务列表为空，因此其虽然能够顺利通过身份鉴别，而无法通过服务鉴别，从而使 $host_2$ 的 Web 服务无法执行。到第 60 s，增加了 $user_2$ 的 Web 服务后，其就能够通过流鉴别。140 s 时，$user_1$ 启动 E-mail 服务，由于该过程不再需要对身份属性进行鉴别，因此其吞吐量相对于 $user_2$ 启动 E-mail 服务能够更快（具体分析见实验 2）的达到链路带宽的一半。$user_1$ 与 $user_2$ 访问 Web 服务过程中实体不需要任何附加操作就完成了鉴别过程，因此该实验同时证明了 SDFAC 具有很好的透明性。

实验 2 SDFAC 可用性分析

在实验 1 的基础上，本文在 $host_1$ 上运行 *Wireshark* 程序记录连接发起到第一个分组 ACK 返回的时延。设在 SDFAC 机制下，从发送首个分组到接收到该分组 ACK 的时延记为 T_{SDFAC}，

注1 About POX, http://www.noxrepo.org/pox/about-pox

注2 OpenVswitch, http://openvswitch.org/

注3 Tcpdump, http: //www.tcpdump.org.

而不采用 SDFAC 机制时的时延记为 $T_{\text{NO-SDFAC}}$。重复测量 50 次，图 9 给出两者每次的测量数据及其平均值。

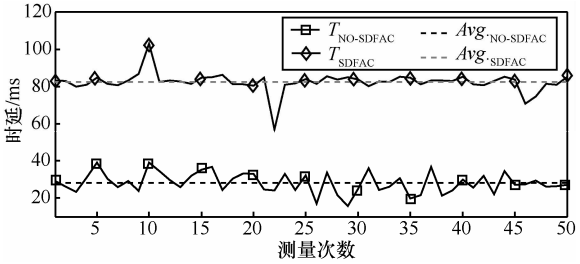


图 9　SDFAC 和 NO-SDFAC 通信建立时延对比

测量结果及分析。从图 9 可以看到，增加了 FAP 机制后，流建立时延平均增加了 52.3 ms。T_{SDFAC} 主要由以下 3 种时延组成：1) $host_1$ 应用的第 1 个分组到达控制器的时延；2) 控制器进行流鉴别的时延；3) 控制器下发流表的时延，分组 ACK 到达 $host_1$。测量发现在组成 T_{SDFAC} 的 3 种时延中，时延 2)占据了总时延的 93%。因此在流鉴别过程中主要采用了公钥密码机制，因此加/解密算法的复杂度将直接决定了其时延。图 10 显示了公钥密码机制中 2 种主要的密钥强度， 即密钥长度为 1 024 bit 以及 2 048 bit 下 SDFAC 的处理时延。

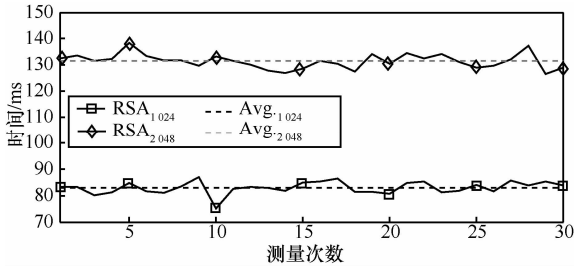


图 10　1 024 bit 密钥与 2 048 bit 密钥下 T_{SDFAC} 的对比

通过图 10 的测量可以看到，2 048 bit 长度密钥相对于 1 024 bit 鉴别过程平均时延从 55 ms 增加到了 102 ms，增加了约 1 倍，但是仍在可行通信时延的范围内，因此本文所设计的 SDAM 机制具有很好的可行性。

5　结束语

本文提出了一种基于 SDN 的流接入控制机制 SDFAC，该机制通过为合法用户授予相应的服务访问的权限，实现了对用户访问行为的控制。SDN 体系架构基于流的细粒度管理以及动态、灵活、可编程的控制机制为实现 SDFAC 提供了良好的基础。基于 SDFAC 原型系统的实验和测量已经说明了其可用性，但是其基于角色的管理机制以及流鉴别协议还需要进一步研究和完善。

参考文献：

[1] SANDHU R S, COYNE E J, FEINSTEIN H L, *et al*. Role-based access control models[J]. IEEE Computer, 1996, 29(2):38-47.

[2] DANGOVAS V, KULIESIUS F. SDN-driven authentication and access control system[J]. Society of Digital Information & Wireless Communication, 2014.

[3] AHMAD I, NAMAL S, YLIANTTILA M, *et al*. Security in software defined networks: a survey[J]. IEEE Communications Survey & Tutorials, 2015, 99: 1-30.

[4] YOON C H, PARK T J, LEE S G, *et al*. Enabling security functions with SDN: a feasibility study[J]. Computer Networks, 2015, (85): 19-35.

[5] HU Z Y, WANG M G, YAN X Q, *et al*. A comprehensive security architecture for SDN[A]. Proceedings of the 18th International Conference on Intelligence in Next Generation Networks[C]. Paris, France, 2015.30-37.

[6] KERPEZ K J, CIOFFI J M, GINIS G, *et al*. Software-defined access networks[J]. IEEE Communication Magazine, 2014, 52(9):152-159.

[7] MATIAS J, GARAY J, MENDIOLA A, *et al*. Flow NAC: flow-based network access control[A]. Proceedings of 2014 3rd European Workshop on Software Defined Networks[C]. Budapest, 2014. 79-84.

[8] IEEE Std. 802.1X-2010, Port-Based Network Access Control, [EB/OL]. http://www. ieee802.org /1/ pages/802.1x-2010.html.

[9] Trusted computing group. trusted network connect architecture for Interoperability, specification version 1.5[EB/OL]. http://www.trusted decomputing group .org/tnc/, 2012.

[10] CHAKRABORTY S, RAY I. TrustBAC-integrating trust relationships into the RBAC model for access control in open system[A]. Proceedings of the 11th ACM symposium on Access control models and technologies[C]. New York, USA, 2006.49-58.

[11] CASADO M, GARFINKEL T, AKELLA A, *et al*. SANE: a protection architecture for enterprise networks[A]. Proceedings of USENIX Security Symposium[C]. 2006. 1-12.

[12] CASADO M, FREEDMAN M J, PETTIT J, *et al*. Ethane: taking control of the enterprise[J]. ACM SIGCOMM Computer Communication Review, 2007,37(4):1-12.

[13] ZHENG R B, YANG W L, ZHOU J. Future access architecture: software-defined access networking[A]. Proceedings of IEEE the 11th Consumer Communications and Networking Conference[C]. Las Vegas, NV, 2014.881-886.

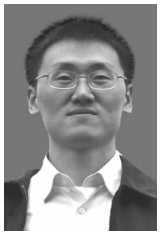
[14] KLAEDTKE F, KARAME G O, BIFULCO R, *et al*. Towards an access control scheme for accessing flow in SDN[A]. Proceedings of the 1st IEEE Conference on Network Softwarization[C]. London, 2015.1-6.

[15] MCKEOWN N, ANDERSON T, BALAKRISHNAN H, *et al*. Openflow: enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008 38: 69-74.

[16] MAZIERES D, KAMINSKY M, KAASHOEK M F, *et al*. Separating key management from file system security[J]. ACM SIGOPS Operating System Review, 1999,33(5):124-139.

作者简介：

王秀磊（1988-），男，山东邹城人，解放军理工大学博士生，主要研究方向为软件定义网络、网络安全、网络测量和性能评价。



张国敏（1979-），男，山东济南人，博士，解放军理工大学讲师，主要研究方向为网络管理、分布式计算等。

胡超（1984-），男，江西吉安人，博士，解放军理工大学讲师，主要研究方向为分布式计算、未来网络等。

陈鸣（1956-），男，江苏无锡人，博士，解放军理工大学教授、博士生导师，主要研究方向为网络测量、网络管理和网络体系结构等。

魏祥麟（1985-），男，安徽砀山人，博士，解放军理工大学工程师，主要研究方向为数据中心网络、无线网络安全、对等网络等。