计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,26(3):198−204 [doi: 10.15888/j.cnki.csa.006090]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 SDN/NFV 的安全服务链自动编排部署框架①

张　奇

(北京工业大学 计算机学院, 北京 100124)

摘　要: 针对云计算等虚拟化环境的安全防护问题, 提出了一种基于 SDN/NFV 技术思想的安全服务链自动编排部署框架. 论文通过扩展 ABAC 策略模型以描述用户的安全需求, 采用优先级解决策略冲突以编排虚拟安全设备, 依据网络中虚拟安全设备实例负载与实时链路传输时延来调度网络流, 最终由 SDN 控制器生成流表下发到网络中完成流量重定向, 实现了根据安全需求自动构建安全服务链的过程. 整个框架在基于开源控制器 FloodLight 和虚拟安全设备的实验环境中实现了自动编排部署, 取得了预期效果.

关键词: 云计算; 虚拟化; 安全服务链; 策略冲突; 虚拟安全设备

引用格式: 张奇.基于 SDN/NFV 的安全服务链自动编排部署框架.计算机系统应用,2018,26(3):198–204. http://www.c-s-a.org.cn/1003-3254/6090.html

Automatic Scheduling Deployment Framework for Security Service Chain Based on SDN/NFV

ZHANG Qi

(School of Computer and Science, Beijing University of Technology, Beijing 100124, China)

Abstract: Aiming at the security protection in virtual environments such as cloud computing, an automatic scheduling deployment framework of security service chain based on SDN/NFV is proposed in this paper. The ABAC strategy model is extended to describe the security requirements of users and priorities are used to solve the policy conflicts to arrange virtualized security appliance. The load of each virtualized security appliance instance and the real-time link transmission delay are quantified to dispatch network traffic. Finally, the flow table generated by SDN controller is sent to the network to complete traffic redirection and implement the process of automatically building the security service chain according to the security requirements. The entire framework is implemented in the experimental environment to achieve the automatic scheduling deployment based on floodlight, virtualized security appliance, and it has obtained anticipatory effects.

Key words: cloud computing; virtualization; security service chain; policy conflicts; virtualized security appliance

随着云计算和软件定义网络 (Software Defined Network, SDN) 的快速发展, 如何快速的重构传统网络的安全解决方案, 进而提升网络安全防护的灵活性与效率, 成为了亟待解决的问题. 传统网络的服务链 (Service Chain, SC) 将满足特定属性的网络数据流牵引经过由多个业务功能服务节点编排组成的服务序列, 为传统网络提供了防控恶意攻击的手段. 软件定义安全 (Software Defined Security, SDS) 架构将网络安全设备的控制平面与数据平面进行了解耦, 底层抽象为安全资源池里的资源, 顶层通过软件定义的方式弹性的编排安全服务来实现灵活的安全防护. 本文结合传统网络的服务链思想与软件定义安全理念, 研究虚拟化网络环境中动态编排虚拟安全服务节点的策略, 通过扩展基于属性的访问控制策略模型[1](Attribute-Based Access Control, ABAC) 对网络中的数据流、虚拟安全设备 (Virtualized Security Appliance, VSA) 等资源进行描述, 构建网络流与虚拟安全设备等资源的映射关系, 形成针对特定网络流的个性化的安全服务链. 本文提

① 收稿时间: 2017-02-27; 修改时间: 2017-03-27; 采用时间: 2017-09-26; csa 在线出版时间: 2018-02-09

出了基于软件定义网络与网络功能虚拟化 (Network Function Virtualization, NFV) 技术思想的安全服务链自动化编排部署框架. 该框架根据策略构建安全服务链, 按照优先级明确需要使用安全资源池中的哪些类虚拟安全设备, 确定其先后顺序以编排虚拟安全设备序列 (VSA 序列), 然后选取负载与实时链路传输时延最优的虚拟安全设备实例 (VSA 实例) 加入安全服务链, 组成针对该安全需求策略的安全服务链, 并通过 SDN 网络流表机制将网络业务流依次牵引经过所需的虚拟安全设备实例进行检测防护. 论文重点讨论了安全服务链的安全需求策略描述、策略冲突、设备编排以及网络流调度问题.

1 相关工作

传统网络的服务链与物理网络拓扑紧密结合, 通过手工配置多种防护策略, 将安全设备串行到网络流路径中, 设备之间耦合度大、拓扑依赖严重, 难以满足业务快速迭代需求.

近年来, SDN 领域也有一些关于安全服务链的相关研究. 当前关于安全服务链的研究主要集中于策略构建、网络流调度等几个方面. 在安全服务链的策略构建方面, 文献[2]提出了基于 SDN/NFV 构建安全服务链的技术思想, 通过编排软件安全模块来满足未来网络安全业务需求, 但其缺少具体的安全服务构建策略. 文献[3]基于 OpenFlow 协议对网络资源进行集中的监控和管理, 通过下发交换机流指令牵引网络流到中间设备, 但并未提及应用层的策略决策机制. 文献[4]通过修改 OpenFlow 协议 QoS 字段来标识不同网络流服务需求, 提出了基于服务类型的安全服务链理论模型. 但该方案拓展了 SDN 网络的南向标准协议, 兼容性较差且需预先编排各个服务类型对应的服务链. 文献[5]提出了基于策略的网络虚拟资源管理机制, 根据不同数据流的业务需求动态的编排服务节点, 但其并未涉及应用策略更新引起的策略冲突问题.

在安全服务链的网络流调度研究方面, 文献[6]为了兼容传统硬件安全设备, 在 SDN 网络中部署传统硬件安全设备来实现安全防护目标, 其主要思想是通过 SDN 的全局拓扑感知与网络流调度能力将业务流依次牵引到各种实体安全设备中进行安全防护. 但该方案拓扑依赖严重、且需手工部署安全设备. 文献[7]提出了基于链路时延的贪婪算法, 通过路由跳数来评估链路时延并根据流量规模来部署安全设备从而有效牵引数据流. 但其并未综合考虑实时网络链路状态以及网络流动态多变特性且缺乏对安全设备负载的有效评估.

2 安全服务链自动编排部署框架

要在网络中实现安全防护解决方案的自动化部署, 根据上层用户的安全业务需求, 自动构建安全服务链策略, 动态编排虚拟安全设备, 快速调度网络数据流到特定的虚拟安全设备序列实现安全防护的目标, 如何有效的实现策略冲突处理与网络流调度优化是两个关键难题. 本文设计的安全服务链自动部署框架包括以下几个部分: 策略冲突决策节点、网络流调度节点、SDN 控制器, 具体如图 1 所示.

2.1 策略冲突决策节点

策略冲突决策节点通过开放北向接口, 接收用户安全业务需求, 如为 Web 服务器提供安全防护, 则需为其构建一条由入侵检测 WAF 和流量清洗 ADS 设备组成的安全服务链策略, 并将访问数据流依次牵引经过上述策略的 VSA 序列进行过滤. 因细粒度的访问控制及网络流属性复杂, 描述不同安全业务需求的策略可能涉及相同的网络流, 使得先后下发的策略之间存在动作冲突. 策略冲突决策节点通过优先级完成特定属性网络流的 VSA 序列的正确编排, 保证网络中不存在冲突, 并将策略决策结果发送给网络流调度节点.

2.2 网络流调度节点

网络流调度节点主要负责管理安全资源池的 VSA 实例, 监控它们的负载信息 (CPU 使用率、内存使用率), 配置它们的防护策略, 同时根据策略决策结果解析 VSA 序列中安全设备类型并结合实时链路状态信息, 评估最优的网络流调度路由, 选取合适的 VSA 实例, 并将这些信息加入策略中, 发送交给 SDN 控制器处理完成流量重定向.

2.3 SDN 控制器

SDN 控制器对网络资源进行集中的控制与管理, 具备全局网络拓扑监控与流指令下发能力. 当收到网络流调度节点的路由选择结果, SDN 控制器对其进行解析、生成流指令并下发给交换机, 将网络流重定向并依次牵引经过对应的 VSA 实例进行安全防护处理, 完成整个安全服务链框架的自动化部署. 相反, SDN 控制器也可根据上述网络流调度路由等信息, 删除之前已下发的交换机流指令.

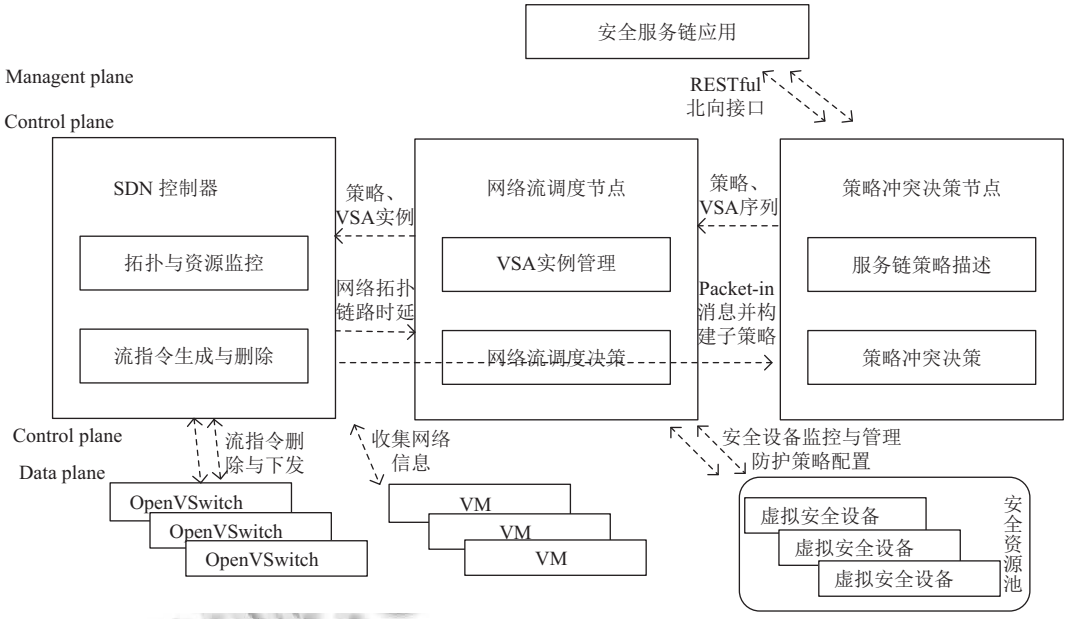


图 1 安全服务链自动编排部署框架

3 安全服务链实现机制

3.1 安全服务链策略描述

安全需求的描述是自动编排和部署安全服务链的依据和前提. 基于属性的访问控制模型提供了复杂信息系统中的动态的权限分配和更加细粒度的访问控制[1]. 本文通过扩展 ABAC 策略模型对 SDN 网络中的上层应用的安全需求进行描述, 构建安全服务链策略, 每条策略均可解析为一条安全服务链. 策略的主体、客体、权限以及 VSA 优先级可表示为 *Policy*(*Subject*, *Object*, *Permission*, *Priority*), 策略语法树如图 2 所示.

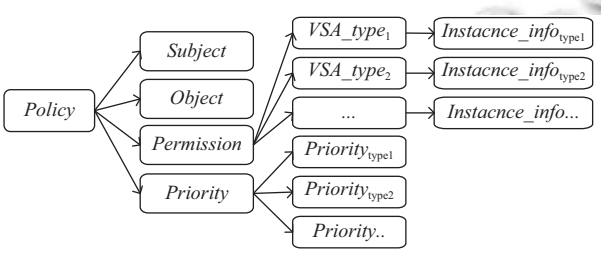


图 2 策略语法树

(1) *Subject*: 策略的执行者, 包括策略冲突决策节点、网络流调度节点或 SDN 控制器.

(2) *Object*: 满足特定属性的网络流, 为符合 OpenFlow 协议包头域的 12 元组属性域.

(3) *Permission*: 网络流重定向操作, 牵引网络流依次经过该策略对应的 VSA 有序序列, 且 VSA 序列中同种类型的安全设备均只出现一次. *VSA_type*是安全资源池中安全设备类型, 多个类型的安全设备组成 VSA 序列. *Instance_Info*为该类型的 VSA 实例相关信息, 如该实例的接入交换机、端口等.

(4) *Priority*: 用于解决策略动作权限的冲突. 用户可对各类型 VSA 分别设置优先级, 通过优先级编排 VSA 序列, 按照 VSA 优先级从大到小顺序依次牵引网络流, 保证策略正确执行. $priority_{type}$为该类型 VSA 对应的优先级, 数值越大表明优先级越高.

3.2 策略冲突决策算法

为了保证所有安全服务链策略正确实施, 使得网络流正确的经过安全服务链策略对应的 VSA 序列进行防护, 需要建立完备的策略冲突决策机制. 为此, 采用优先级对安全服务链策略中的 VSA 序列进行动态的、合理的编排以解决策略冲突.

策略冲突决策节点为用户的安全业务需求构建安全服务链策略后, 首先由 SDN 控制器检索网络中所有交换机上当前正在执行的包含有新构建的安全服务链策略中 12 元组对应的网络流的流规则, 将其删除以触发 Packet-in 消息. 然后, 策略冲突决策节点监听所有的 Packet-in 消息并依次为这些消息描述的所有网络流重构相应的安全服务链策略（该策略由用户下发的安全服务链策略重构所得, 经由网络流调度节点、SDN

控制器依次处理并转换为流规则后即删除），以此解决策略冲突, 下文将介绍具体过程.

(1) Packet-in 消息触发

在 SDN 网络中, 网络流一般由多个数据包组成, 当没有该网络流的转发流规则时, 交换机会将该网络流的首个数据包封装为 Packet-in 消息并发送给 SDN 控制器请求下发流规则, 后续数据包则按此流规则进行转发. 因此, 为了保证安全服务链策略的正确实施, 每当安全服务链框架为用户的安全业务需求构建一条新的安全服务链策略时, 会检索网络中所有交换机的流规则, 根据 OpenFlow 协议[8]的 12 元组匹配域结构来判断流规则包头域的 12 元组匹配域是否包含于策略的客体属性域中, 若是, 则说明网络中已经有针对该 12 元组的网络流的流规则在执行, 那么将这些流规则删除以触发 Packet-in 消息.

(2) 安全服务链策略重构

策略冲突决策节点通过扩展 SDN 控制器以监听并解析交换机上传的 Packet-in 消息, 并依次将这些 Packet-in 消息封装的数据包信息与用户下发的所有安全服务链策略逐条进行检测, 判断该数据包的 12 元组匹配域是否包含于这些策略的客体属性域中; 若该数据包并未包含于某条安全服务链策略中, 则直接交给 SDN 控制器的默认转发模块 Forwarding 处理, 该模块会为该数据包建立转发路由以使该数据包从属的网络流到达目的主机; 否则, 则为该数据包重构其安全服务链策略, 具体过程如下:

1) 若该数据包包含于某一条安全服务链策略中, 则直接为该数据包重构相应的安全服务链策略, 策略客体属性域为该数据包的 12 元组匹配域信息, 动作权限与 VSA 优先级为匹配上的安全服务链策略的 VSA 序列与 VSA 优先级信息.

在安全服务链框架中, 为了保证虚拟安全设备的有序编排, 同一个策略内不允许存在数值相等的优先级; 为了保证安全服务链策略的重构, 在用户下发安全服务链策略时检测该策略是否与已有策略存在冲突. 若存在, 则检测新加入策略是否与已有策略存在相等数值的优先级, 若是则不执行该策略并反馈给用户; 在策略重构过程中, 若多条安全服务链策略的 VSA 序列中存在同一种类 VSA 且它们的优先级数值不同, 则以最新安全服务链策略的 VSA 优先级覆盖其他安全服务链策略的 VSA 优先级, 以满足用户更新 VSA 优先级的业务需求.

2) 若该数据包同时包含于多条安全服务链策略中, 则将这些策略加入策略集合 *policylist* 中, 并以 *policylist* 中的所有策略为依据, 为该数据包重构相应的安全服务链策略 $policy_n$, 具体如图 3 所示.

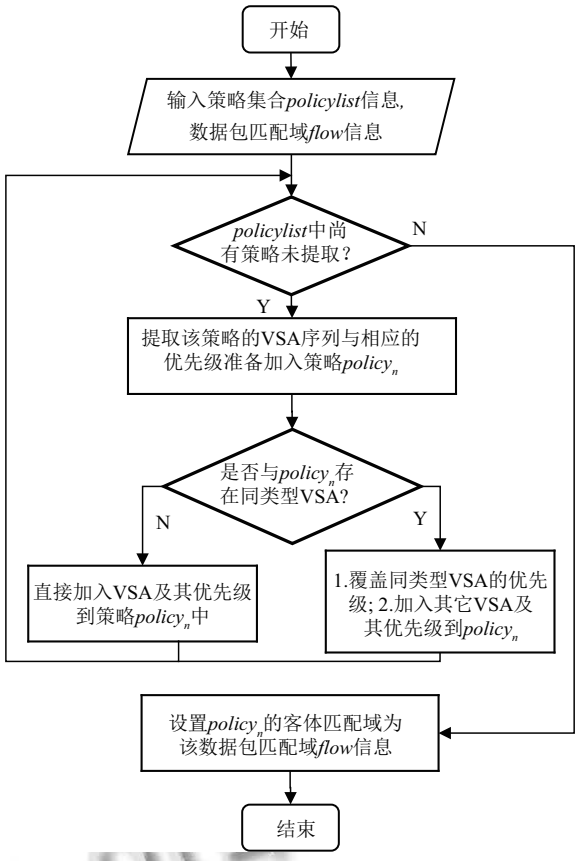


图 3 安全服务链策略重构

3.3 网络流调度算法

为了提高虚拟安全设备 VSA 的利用率和网络流调度效率, 进而提升其承载物理服务器的利用率. 本文采用 Qemu 或 Kvm 等虚拟化技术实现虚拟安全设备的弹性部署, 并根据安全设备需求、VSA 实例负载、实时链路状态, 动态的选取 VSA 实例完成安全服务链的网络流调度路由选择优化.

3.3.1 安全资源池

SDN 网络拓扑图可表示为$G(V,E)$, 其中 E 为与交换机相连接边的集合, $V=S\cup H\cup D$, S 为交换机集合, H 为主机集合, D 为虚拟安全设备集合 (安全资源池), 那么安全资源池具体如下:

(1) 安全资源池中有多个类型的安全设备, 可提供

多种不同安全能力, 如: 入侵检测 ADS、网页防护 WAF、防火墙 FW 等. 具备 p 种类型安全设备的安全资源池表示为: $D=\{D_1,D_2,D_3,\cdots,D_p\}$, 其中, p 为正整数.

(2) 同一类型安全设备可存在多台 VSA 实例 $D_p=\{d_{p1},d_{p2},\cdots,d_{pq}\}$, 其中, q 为正整数, 指实例个数. 且任意一个实例对象d_{pq}的内存利用率和 CPU 利用率的空闲程度分别为:

$$idle_{m(d_{pq})}=(thr_{m(d_{pq})}-uti_{m(d_{pq})})/thr_{m(d_{pq})} \quad (1)$$

$$idle_{c(d_{pq})}=(thr_{c(d_{pq})}-uti_{c(d_{pq})})/thr_{c(d_{pq})} \quad (2)$$

其中, uti, thr 分别表示当前利用率与利用率阈值, 阈值 thr 由 VSA 实例任务需求而定.

(3) 若某 VSA 实例的 CPU 或者内存利用率的空闲程度小于零, 则认为该实例已过载, 需要初始化新的实例并接入到网络中.

3.3.2 链路状态度量

安全服务链牵引网络流经过特定顺序的安全服务节点, 需要考虑链路传输时延对数据传输性能的影响. 安全服务链中时延分为两类:链路传输时延、服务节点处理时延. 服务节点处理时延为安全业务节点时延之和, 指数据包进入虚拟安全设备后, 对数据包进行相应的检测与分析, 由于同一条服务链的内部虚拟安全设备顺序、类型固定, 且时延差异不大, 本文忽略不计. 对$\forall e\in E$, 随机测量 t 次时延取其均值, 则链路时延可表示为:

$$Delay(e)=\sum_{i=1}^{t}delay(e)/t \quad (3)$$

3.3.3 安全服务链解析

安全服务链通过策略来描述, 每条策略可解析为一条安全服务链. 如当防御 DDoS 攻击时, 用户需要构建策略调度网络流依次经过 Web 防护设备 WAF、流量清洗设备 ADS 形成细粒度的安全防护, 该策略对应的安全服务链$ServiceChain$可表示为:

$$ServiceChain=\{(Src,D_1,D_2,Dst)\},D_1\in D,D_2\in D \quad (4)$$

且可分解为$Path_{Src,D1},Path_{D1,D2},Path_{D2,Dst}$三个$Path$子单元. 那么$\forall ServiceChain$有:

$$Path=[Path_{Src,D1},Path_{D1,D2},\cdots,Path_{Dn,Dst}] \quad (5)$$

其中, $Path$为当前安全服务链的安全设备类型调度路由, 按照 VSA 优先级从大到小顺序生成路由, 相邻虚拟安全设备之间均有调度路由$Path_{Di,Dj}$.

网络流调度节点将策略$policy_n$解析为相应的安全服务链, 通过网络流调度算法选取合适的 VSA 实例存入$policy_n$中并更新策略执行者为网络流调度节点, 然后将该安全服务链策略发送给 SDN 控制器. SDN 控制器接收并解析该策略生成相应的网络流调度路由 $path=[path_{Src,d1},path_{d1,d2},\cdots,path_{dn,Dst}]$, 并分解其为若干个 子单元, 依次生成流指令并下发交换机完成流量重定向, 网络流匹配域为策略$Object_{policy_{nj}}$属性域信息.

算法 1. 网络流调度算法

输入: $G(V,E)$, $policy_n$

输出: $policy_n$ with VSA Instances' Info

1. Find $List_{Dp}$&$Path(5)$ for $ServiceChain(4)$ from $policy_n$ //提取策略 $policy_n$ 所需 VSA 类型
2. Find $List_{dpq}$ for $List_{Dp}$ in $G(V,E)$ //获取所需类型 VSA 所有实例
3. For each D_p in $List_{Dp}$
4. For each d_{pq} in $List_{dpq}$
5. compute $idle_{m(d_{pq})}(1)$&$idle_{c(d_{pq})}(2)$ //计算每个实例负载
6. Quick Sorting to sort $idle_{m(d_{pq})}$*$idle_{c(d_{pq})}$ //对各类型的实例负载空闲程度排序
7. select the top k VSAs of each type and input to $List_{path}$ //获得$pow(k,length_{path}-1)$种实例组合选择, 每种组合对应一条网络流调度路由 $path$ 并存入集合 $List_{path}$ 中
8. If $num_{dpq}<k$ then start new VSA //若该类型 VSA 实例空闲个数num_{dpq}小于 k 个, 则实例化新 VSA 接入网络
9. End for
10. End for
11. For all $path$ in $List_{path}$
12. compute $Delay(e)(3)$ //采用最短路径优先算法对每种网络流调度路由计算其传输性能:时延消耗
13. select the min delay of all $path$ //最小时延的作为最优数据流调度路由和最佳 VSA 实例组合
14. write VSA Instances' Info into $policy_n$ //将最优路由上的实例信息写入 $policy_n$
15. End for
16. Send $policy_n$ to SDN Controller

4 实验设计和结果分析

4.1 实验环境

本文的实验环境搭建在 64 位 Ubuntu 服务器上. 通过扩展开源控制器 FloodLight 实现策略冲突决策与网络流调度机制, 并启动两台虚拟交换机 (OpenV Switch, OVS) 连接至控制器, 将安全设备 Web 应用防火墙 WAF(串联部署模式)、流量清洗设备 ADS(旁路部署模式) 和流量生成器 Tester 部署至网络中. 通过流量生成器模拟访问者 (正常数据流与攻击数据流, IP 地址

为 3.3.3.1) 与被访问者 (Web 服务器, IP 地址为 6.6.6.5), 实验环境如图 4 所示.

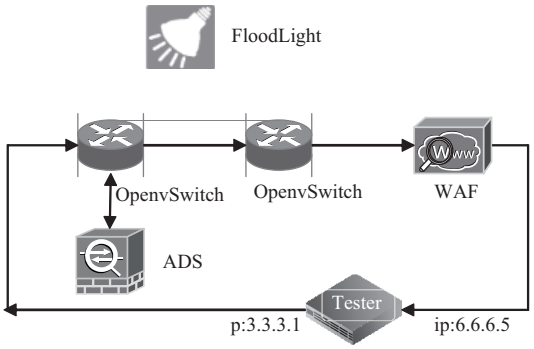


图 4 实验环境

4.2 实验过程

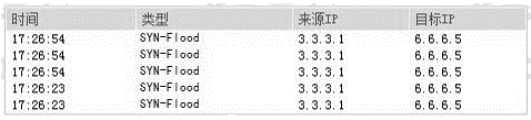
基于上述环境, 本文通过安全服务链框架开放的北向接口实现并测试了抗分布式拒绝服务攻击 (Distributed Denial of Service, DDoS) 实验, 其应用层的安全需求为: 对部署在 6.6.6.5 上的 Web 服务器进行安全防护. 用户构建了两条安全服务链, 策略描述为: 策略 1 将访问数据流重定向到 WAF 检测异常数据流, VSA 优先级为 10; 策略 2 将访问数据流重定向到 ADS 清洗异常数据流, VSA 优先级为 20. 可分别表示为: 策略 1 (策略冲突决策节点, 访问数据流, WAF, 10); 策略 2(策略冲突决策节点, 访问数据流, ADS, 20).

如图 4 所示, 策略 1 被自动编排部署后, 牵引访问数据流至 WAF 做异常检测. 当在访问 Web 服务器的数据流中混入 DDoS 攻击数据流 (SYN_FLOOD) 后, 用户检测 WAF 日志并发现异常流攻击时, 如图 5, 触发策略 2 下发. 因策略 2 的 VSA 优先级大于策略 1, 执行策略冲突决策算法生成子策略 (策略冲突决策节点, 访问数据流, ADS→WFA, 20→10), 来替代已下发策略 1 和新加入策略 2. 网络流调度算法解析编排结果, 选取合适的 VSA 实例, 配置其防护策略, 更新策略执行者, 写入 ADS 和 WAF 信息, 然后发送给 SDN 控制器生成流指令并下发交换机, 先牵引数据流至 ADS 清洗攻击流如图 6, 并将正常数据流回注网络继续发送给 WAF 检测, 此时并未发现异常攻击流.

为了衡量该框架的时间性能, 本文进一步统计了控制器 FloodLight 在收到各安全服务链策略与 OpenFlow 流表下发完成之间的时间差, 每条策略均重复多次, 取其均值. 具体如表 1 所示.

本地时间	事件类型	风险级别	服务器IP：端口	动作
2016-06-14 17:26:34	SYN_FLOOD攻击	▲	6.6.6.5:0	进入防护状态
2016-06-14 17:26:34	SYN_FLOOD攻击	▲	6.6.6.5:999	进入防护状态
2016-06-14 17:26:34	SYN_FLOOD攻击	▲	6.6.6.5:998	进入防护状态
2016-06-14 17:26:34	SYN_FLOOD攻击	▲	6.6.6.5:997	进入防护状态

图 5 WAF 检测攻击流记录



时间	类型	来源IP	目标IP
17:26:54	SYN-Flood	3.3.3.1	6.6.6.5
17:26:54	SYN-Flood	3.3.3.1	6.6.6.5
17:26:54	SYN-Flood	3.3.3.1	6.6.6.5
17:26:23	SYN-Flood	3.3.3.1	6.6.6.5
17:26:23	SYN-Flood	3.3.3.1	6.6.6.5

图 6 ADS 清洗攻击流记录

表 1 安全服务链的时间开销

策略编号	1	2	平均
执行时间 (ms)	92	108	100

实验结果表明了安全服务链自动编排部署框架能准确地进行策略相关性检测、处理策略冲突、选取 VSA 实例、配置防护策略、生成并下发交换机流表, 实现虚拟化环境中安全服务链的自动化编排和部署. 同时, 该框架具有毫秒级的响应速度, 虽然策略 2 下发时与已有策略 1 存在冲突, 冲突检测与处理增加了时间消耗但并不明显, 在实际网络中用户的安全服务链策略应该避免过多复杂关联的出现, 且与传统网络的人工手动部署安全设备与配置防护策略相比, 该框架具备良好的时间性能, 大大的缩减了人工部署与运维成本, 提高了虚拟化网络中安全防护的效率.

5 结论

本文设计了策略冲突决策算法和网络流调度算法, 通过优先级解决策略冲突以保证 VSA 序列的正确编排, 并将 VSA 实例负载和实时链路传输时延作为网络流调度依据, 提出了基于 SDN/NFV 的弹性的、动态的安全服务链自动编排部署框架. 在云计算等虚拟化网络中, 安全服务链框架既可部署于网络边界位置作为边界网关, 实现“南北向”网络流防护, 也可部署在租户内部 VM 之间, 实现“东西向”网络流防护, 均可灵活、快速地按照用户应用的安全需求提供深度安全防护, 同时提高虚拟安全设备和底层基础设施的效率.

参考文献

1 Yuan E, Tong J. Attributed based access control (ABAC) for web services. Proceedings of 2015 IEEE International

Conference on Web Services. Orlando, FL, USA. 2005. 569.
2 Lee W, Choi YH, Kim N. Study on virtual service chain for secure software-defined networking. The International Conference on Control and Automation. 2013. 177–180.
3 Gushchin A, Walid A, Tang A. Scalable routing in SDN-enabled networks with consolidated middleboxes. Proceedings of the 2015 ACM SIGCOMM Workshop on Hot Topics in Middleboxes and Network Function Virtualization. London, UK. 2015. 55–60.
4 Martini B, Paganelli F, Mohammed AA, *et al*. SDN controller for context-aware data delivery in dynamic service chaining. Proceedings 1st IEEE Conference on Network Softwarization (NetSoft). London, UK. 2015. 1–5.
5 Giotis K, Kryftis Y, Maglaris V. Policy-based orchestration of NFV services in software-defined networks. Proceedings of the 1st IEEE Conference on Network Softwarization. London, UK. 2015. 1–5.
6 Qazi ZA, Tu CC, Chiang L, *et al*. SIMPLE-fying middlebox policy enforcement using SDN. Proceedings of the ACM SIGCOMM 2013. New York, NY, USA. 2013. 27–38.
7 Zhang Y, Beheshti N, Beliveau L, *et al*. StEERING: A software-defined networking for inline service chaining. Proceedings of the 21st IEEE International Conference on Network Protocols. Goettingen, Germany. 2013. 1–10.
8 McKeown N, Anderson T, Balakrishnan H, *et al*. OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM Computer Communication Review, 2014, 38(2): 69–74.