基于多维异构特征与反馈感知调度的SDN内生安全控制平面

王 涛,陈鸿昶
(中国人民解放军战略支援部队信息工程大学,河南郑州 450003)

摘 要: 为弥补现有研究在软件定义网络(Software-Defined Networking,SDN)控制平面未知漏洞防御方面的空白,本文提出了一种基于多维异构特征与反馈感知调度的内生安全控制平面的设计方案.该方案以"冗余、异构和动态"为切入点,通过组合执行体冗余集构建策略、多维异构元素着色策略和动态反馈感知调度策略,有效增加SDN控制平面对攻击者所呈现的执行体时空不确定性(逆转攻防不对称性).相关仿真结果表明该方案可以收敛全局执行体数目、增加执行体之间的多维异构度并降低系统全局失效率.

关键词: SDN控制平面内生安全;未知漏洞;劫持攻击;冗余集构建;多维异构特征;反馈感知调度
中图分类号: TP309.1 **文献标识码:** A **文章编号:** 0372-2112 (2021)06-1117-08
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.12263/DZXB.20200638

An SDN Endogenous Security Control Plane Based on Multi-dimensional Heterogeneous Features and Feedback-Aware Scheduling Strategy

WANG Tao, CHEN Hong-chang
(*PLA Strategic Support Force Information Engineering University, Zhengzhou, Henan* 450003, *China*)

Abstract: In order to make up for the gap of the existing research in the defense of unknown SDN (Software-Defined Networking) vulnerabilities, this paper proposes an SDN endogenous security control plane based on multi-dimensional heterogeneous features and feedback-aware scheduling strategy. This scheme effectively increases the spatiotemporal uncertainty of the execution body in the SDN control plane (reversing the asymmetry between attack and defense) by combining a redundant set construction strategy, a multi-dimensional heterogeneous element coloring strategy, and a dynamic feedback-aware scheduling strategy. The related simulation results show that the scheme can converge the number of executive bodies, increase the multi-dimensional heterogeneity index and reduce the global failure rate of the system.

Key words: SDN control plane endogenous security; unknown vulnerabilities; hijacking attack; redundant set construction; multi-dimensional heterogeneous features; feedback-aware scheduling

1 引言

软件定义网络(Software-Defined Networking,SDN)[1]彻底改变了传统网络体系架构中控制与转发紧耦合的运维形态.控制平面作为SDN架构的"神经中枢"维护着全局网络的拓扑、路由等关键状态信息,负责产生并向数据平面分发运转指令(提供数据包级别的细粒度管理).但正是由于这一分离属性,令控制平面成为攻击者的首要攻击目标.一旦攻击者通过后门或固有漏洞成功劫持控制平面,攻击者便获得了网络的实际控制权.

目前国内外针对加固SDN控制平面安全已经具有了一些研究方案[2,3],取得了一定的防御效果.但是这些"补丁式"的安全方案并未从本质上解决SDN控制平面所面临的安全威胁,各类针对SDN控制平面的漏洞依然层出不穷.究其根源,主要包括如下原因:(1)现有控制平面为了实现复杂的网络功能,其本身代码量的数量级较大,导致了漏洞的不可避免性;(2)"补丁式"安全方案具有明显的防御滞后性,只有攻击者利用其挖掘的新漏洞对控制平面成功发动攻击后,防御者才

收稿日期:2020-06-30;修回日期:2020-09-22;责任编辑:覃怀银
基金项目:国家重点研发计划课题(No.2018YFB0804004);国家自然科学基金创新研究群体项目(No.61521003);国家自然科学基金面上项目(No.62072467)

有可能意识到该漏洞的存在;(3)“补丁式”安全方案缺乏通用性,针对不同的攻击模式依赖于特有的攻击特征,故相应补丁方案仅针对特定攻击或特定漏洞有效. 综上所述,国内外就如何从根本上解决 SDN 控制平面面临的(未知)漏洞威胁仍有较大的研究空白. 为解决该问题,本文结合拟态防御相关理论将拟态基因引入到 SDN 控制平面,提出了基于多维异构特征与反馈感知调度的内生安全控制平面.

2 方案设计

2.1 内生安全控制器通用架构

内生安全控制平面基本设计原理就是利用冗余的功能等价执行体在相同输入的情况下经过一致性输出裁决来降低攻击者得到异常输出的概率. 互为异构的执行体也能极大降低共模漏洞可能导致的一致性错误输出结果,同时,在时间和空间维度动态更改功能等价执行体属性会增加执行体处理环节的不确定性,进一步增加攻击者探测成本并逆转攻防双方的不对称性. 这样内生安全控制平面就通过组合冗余、异构和动态三种属性使控制平面具有天然抗攻击基因. 其通用架构如图 1 所示.

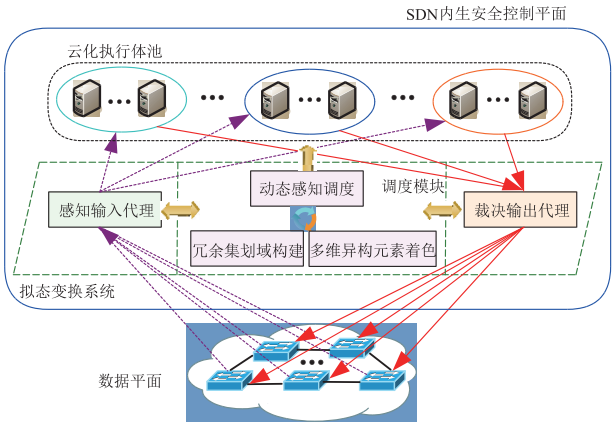


图1 内生安全控制平面通用架构图

由图 1 可知,不同于一般 SDN 控制器,内生安全控制平面本质上是一个主要包括云化执行体池和拟态变换系统的复杂组合系统. 云化执行体池是网络信息处理的实际执行体,也是拟态变换系统的直接操作对象. 值得注意的是,云化执行体池并不仅是多种类型控制器的组合,而是具有多维异构特征的抽象集合(具体介绍见 2.3 节). 另一方面,拟态变换系统是内生安全控制平面的核心组成部分,其主要包括感知输入代理、调度模块和裁决输出代理等三部分. 感知输入代理是拟态变换系统的入口,其主要负责采集数据平面状态信息、与调度模块交互反馈信息等. 调度模块主要由冗余集划域构建组件、多维异构元素着色组件及动态感知调度组件等三部分组成. 其中,冗余集划域构建组件主要负责根据数据平面冗余容错需求度对云化执行体池进行划域,使其在满足数据平面安全及性能需求的同时,提高执行体复用程度;多维异构元素着色组件可以将执行体抽象分解为多维异构元素,然后利用相似度关联的多维异构特征评价指标对执行体的异构元素进行着色,增加现有执行域内执行体之间的异构度;动态感知调度组件可以在感知输入代理与裁决输出代理产生的反馈信息基础上评估安全风险,然后以降低云化执行体池的全局失效率为目标进行执行体动态迁移,从而保证系统全局安全性能. 裁决输出代理负责接收执行体池各域的输出结果,然后按照大数判决等裁决原则过滤异常输出结果并生成一致性策略并下发到数据平面. 由于感知输入代理及裁决输出代理作为拟态防御的通用模块主要承担一些基础性操作,目前也已经较为成熟[4,5],本文不再赘述. 本章将着重介绍调度模块中具有创新性的冗余集构建策略、多维异构度量指标与着色策略、动态反馈感知调度策略.

2.2 冗余集构建策略

考虑到现实数据平面中各交换机存在的差异化安全及性能需求,将整体云化执行体集作为一个统一的冗余集不仅会造成执行体计算资源的浪费,还会造成过度冗余引入的性能波动(直接降低系统可用性). 为解决此问题,我们需要综合考虑数据平面和控制平面供需情况来制定冗余集构建策略.

模型建立过程描述:假设数据平面交换机集合为 $S=\{\mathrm{sw}_1,\mathrm{sw}_2,\cdots,\mathrm{sw}_m\}$,执行体集合为 $E=\{e_1,e_2,\cdots,e_n\}$,执行体之间的通信时延集合为 D. 为解决未知漏洞安全威胁,数据平面按照重要程度设置的冗余度需求集合为 $F=\{f_{\mathrm{sw}_1},f_{\mathrm{sw}_2},\cdots,f_{\mathrm{sw}_m}\}$,最小时延需求集合为 $T=\{t_{\mathrm{sw}_1},t_{\mathrm{sw}_2},\cdots,t_{\mathrm{sw}_m}\}$. 假设交换机 $\mathrm{sw}_i\in S$ 由运行拟态交互协议的执行体域 E_{AS_k} 管理(其中 $E_{\mathrm{AS}_k}\in E_{\mathrm{AS}}=\{E_{\mathrm{AS}_1},E_{\mathrm{AS}_2},\cdots,E_{\mathrm{AS}_l}\}$),则执行体域 E_{AS_k} 需要满足如下条件:首先,该执行体域内执行体数目需要大于 $R_{\mathrm{sw}_i}=2f_{\mathrm{sw}_i}+1$,即,

$$|E_{\mathrm{AS}_k}|\geq R_{\mathrm{sw}_i}=2f_{\mathrm{sw}_i}+1 \tag{1}$$

根据拟态防御理论[4],只有满足此数量要求才能有效实行大数判决等裁决策略,相应地,f_{sw_i} 也就是交换机 sw_i 的容错度量;其次,为满足数据平面性能需求,执行体域 E_{AS_k} 内任意两个执行体 e_p 和 e_q 的通信时延需要小于设定时延需求阈值,即,

$$D_{pq}\leqslant t_{\mathrm{sw}_i},\forall\,\mathrm{sw}_i\in S,\forall\,e_p,e_q\in E_{\mathrm{AS}_k},\forall\,E_{\mathrm{AS}_k}\in E_{\mathrm{AS}} \tag{2}$$

然后考虑到执行体计算资源有限性,我们将每个执行体容量统一表示为 $K=\mathrm{Cap}(e_i),\forall\,e_i\in E$,即执行体 e_i 最多管理 $K=\mathrm{Cap}(e_i)$ 个交换机. 假设域 E_{AS_k} 内执行体 e_p 管理的交换机集合表示为 $\phi(E_{\mathrm{AS}_k}-e_p)$,则其满足如下关系:

$$|\phi(E_{\mathrm{AS}_k}-e_p)|\leqslant \mathrm{Cap}(e_p),\forall e_p\in E_{\mathrm{AS}_k},\forall E_{\mathrm{AS}_k}\in E_{\mathrm{AS}} \quad (3)$$

最后,所有投入使用的执行体数目 n 要小于云化执行体池中可供使用的执行体总数目 N,如果将每个交换机$\mathrm{sw}_i\in S$ 的映射执行体集合表示为 ψ_{sw_i},则其相互之间的关系如式(4)所示.

$$n=\sum_{k=1}^{l}|E_{\mathrm{AS}_k}|=\left|\bigcup_{i=1}^{m}\psi_{\mathrm{sw}_i}\right|<N \quad (4)$$

根据上述模型可知,该问题本质上属于经典装箱问题[6]. 为了合理划分冗余执行体集合,本文提出了需求优先的执行体集划域算法,该算法的基本思路是通过迭代的方法令执行体域优先满足冗余需求度高的交换机,然后在当前执行体域内优先分配剩余容量小的执行体到交换机. 当然,在此过程中对于任意交换机都必须满足其冗余及时延需求,否则另外选择新的执行体域作为候选集. 其具体步骤如算法 1 所示.

算法 1　需求优先的执行体集划域算法

Input: S R T E D
Output: E_{AS} ψ_{sw}
1 :Sort S based on R in descending order
(If R is the same, it is arranged in ascending order based on T.)
2:$k=0$;$E_{\mathrm{AS}_k}\leftarrow\phi$
3:for $\mathrm{sw}_i\in S$ do
4:　if $E_{\mathrm{AS}_k}=\phi$ or $\mathrm{FreeNum}(E_{\mathrm{AS}_k})<R(\mathrm{sw}_i)$ then
5:　$w\leftarrow\mathrm{solve}\left(\sum_{j=i}^{w}R(\mathrm{sw}_j)\leqslant K\cdot R(\mathrm{sw}_i)\wedge\sum_{j=i}^{w+1}R(\mathrm{sw}_j)>K\cdot R(\mathrm{sw}_i)\right)$
6:　$T_{\min}=\min\left(\bigcup_{\mathrm{num}=i}^{i+w-1}T_{\mathrm{sw}[\mathrm{num}]}\right)$
7:　$E_{\mathrm{AS}_k}\leftarrow\mathrm{SearchNewSet}(E\backslash E_{\mathrm{AS}_k},\phi,R(\mathrm{sw}_i),T_{\min},++k)$
8:　end if
9:　Sort E_{AS_k} based on Free Cap in ascending order
10:　for $e_p\in E_{\mathrm{AS}_k}$ do
11:　　if $\mathrm{FreeCap}(e_p)>0$ then
12:　　$\psi_{\mathrm{sw}_i}\leftarrow e_p$
13:　　$R(\mathrm{sw}_i)=R(\mathrm{sw}_i)-1$
14:　　$\mathrm{FreeCap}(e_p)=\mathrm{FreeCap}(e_p)-1$
15:　　if $R(\mathrm{sw}_i)==0$ then
16:　　break
17:　　end if
18:　　end if
19:　end for
20:end for
21:function $\mathrm{SearchNewSet}(E_{\mathrm{rest}},E_{\mathrm{now}},R,T_{\min},k)$
22: if $R==0$ then
23:　return E_{now}
24: end if
25: for $e_i\in E_{\mathrm{rest}}$ do
26:　if $D(e_i,E_{\mathrm{rest}})\leqslant T_{\min}$ and $\mathrm{FreeCap}(e_i)$ then
27:　　$E_{\mathrm{AS}_k}\leftarrow\mathrm{SearchNewSet}(E_{\mathrm{rest}}-\{e_i\},E_{\mathrm{now}}+\{e_i\},R-1,T_{\min},k)$
28:　　if $|E_{\mathrm{AS}_k}|==R$ then
29:　　return E_{AS_k}
30:　　end if
31:　end for
32:end function

2.3　多维异构度量指标与异构元素着色策略

为了赋予执行体域异构属性,本节设计了相似度关联的多维异构特征评价指标,并以该指标为参考提出了基于随机种子预配置的功能等价执行体异构元素着色算法,从而在当前随机种子下增加每个执行体域内的综合异构度.

相似度关联的多维异构特征评价指标:本文提出的内生安全控制平面与传统控制器方案有所不同,作为一种广义控制平面的概念,它的异构考量维度不能仅仅包括控制器类型,而应该从信息系统组成维度全方位衡量. 本节假设域内执行体在系统维度可细化为控制器单元、操作系统单元和处理器单元. 下面首先定义域内执行体的多维异构特征.

假设执行体域 $E_{\mathrm{AS}_k},k\in[1,L]$ 内的执行体 $e_{\mathrm{AS}_k}(p)$,$p\in[1,|E_{\mathrm{AS}_k}|]$ 多维异构特征集合表示为:

$$Q_{e_{\mathrm{AS}_k}(p)}=\{\boldsymbol{Q}_1^{e_{\mathrm{AS}_k}(p)},\boldsymbol{Q}_2^{e_{\mathrm{AS}_k}(p)},\cdots,\boldsymbol{Q}_n^{e_{\mathrm{AS}_k}(p)}\},\forall n\in[1,\mathrm{DimNum}] \quad (5)$$

其中,DimNum 表示维度层数,比如域内执行体在信息系统组成维度可细化为控制器单元(controller)、操作系统单元(OS)和处理器单元(CPU),那么其维度层数即为 3,n 则映射第 n 维度所表示的具体物理含义. 多维异构特征内的第 n 维异构特征向量 $\boldsymbol{Q}_n^{e_{\mathrm{AS}_k}(p)}$ 由对应维度内的具体异构元素所决定,比如第 n 维对应控制器类型维度,异构元素共包括{ODL, ONOS, Floodlight, Ryu, NOX},假设执行体 $e_{\mathrm{AS}_k}(p)$ 在控制器类型维度赋予了 ODL 元素,则其特征向量为 $\boldsymbol{Q}_n^{e_{\mathrm{AS}_k}(p)}=[1,0,0,0,0]$. 同理,在其他异构维度下(如操作系统维度{Win, CentoS, Ubuntu, RedHat, …}、处理器维度{Intel, AMD, IBM, Loongson, …}等)也可根据实际情况表示(异构元素集合为 Γ). 值得注意的是,本模型维度层数及对应维度内异构元素数目均可根据具体情况弹性变化,保证了模型的通用性.

为了增加各执行体域的异构程度,我们首先需要在多维异构特征的基础上按照从异构元素到执行体再到执行体域的逻辑顺序衡量其异构指标. 具体而言,域内不同执行体对应维度的异构元素相似度指标如下式所示:

$$\mathrm{similarity}(p.\,\mathrm{Dim}(i),q.\,\mathrm{Dim}(i))=\boldsymbol{Q}_i^{e_{\mathrm{AS}_k}(p)}\boldsymbol{\Theta}_i[\boldsymbol{Q}_i^{e_{\mathrm{AS}_k}(q)}]^{\mathrm{T}} \quad (6)$$

其中,$\mathrm{similarity}(p.\,\mathrm{Dim}(i),q.\,\mathrm{Dim}(i))$ 表示执行体域 E_{AS_k} 内的执行体 p 和 q 在维度 i 上的异构元素之间的相似度,$\boldsymbol{\Theta}_i$ 是在维度 i 上的异构元素相似状态转移矩

阵,其表达式如式(7)所示:

$$\boldsymbol{\Theta}_i=\begin{pmatrix}1 & s_{12}^i & \cdots & s_{1\text{TypeNum}(\text{Dim}(i))}\\ s_{21}^i & \ddots & & \vdots\\ \vdots & & & \\ s_{\text{TypeNum}(\text{Dim}(i))1}^i & \cdots & & 1\end{pmatrix} \tag{7}$$

其中,s_{mn}^i表示在维度i内的异构元素m和n的相似度(其值域在[0,1]之间),TypeNum(Dim(i))则表示维度i内的异构元素总数. 异构元素相似度越小,s_{mn}^i值越小;如果异构元素完全一致,则$s_{mn}^i=1$(如矩阵对角线元素所示). 考虑到域内执行体是由多维异构元素组成,我们继续定义域内执行体之间的相似度表达式如下所示:

$$\begin{aligned}\text{similarity}(p,q)&=\sum_{i=1}^{\text{DimNum}}(\lambda_i\text{similarity}(p.\text{Dim}(i),q.\text{Dim}(i)))\\&=\sum_{i=1}^{\text{DimNum}}(\lambda_i\boldsymbol{Q}_i^{e_{\text{AS}_k}(p)}\boldsymbol{\Theta}_i[\boldsymbol{Q}_i^{e_{\text{AS}_k}(q)}]^{\text{T}})\end{aligned} \tag{8}$$

其中,λ_i表示第i维异构特征权重系数,其物理意义是域内执行体的第i维异构特征对系统安全产生影响的重要程度. λ_i值越高代表第i维异构特征对系统安全来说越重要,且满足域内执行体各维度的异构特征权重系数的总和为1,即:

$$\sum_{i=1}^{\text{DimNum}}\lambda_i=1 \tag{9}$$

基于上述分析可得对于执行体域层面的相似度关联的多维异构特征评价指标为:

$$\begin{aligned}\text{similarity}(E_{\text{AS}_k})&=\frac{1}{C_{|E_{\text{AS}_k}|}^2}\sum_{p=1}^{|E_{\text{AS}_k}|-1}\sum_{q=p+1}^{|E_{\text{AS}_k}|}\text{similarity}(p,q)\\&=\frac{1}{C_{|E_{\text{AS}_k}|}^2}\sum_{p=1}^{|E_{\text{AS}_k}|-1}\sum_{q=p+1}^{|E_{\text{AS}_k}|}\sum_{i=1}^{\text{DimNum}}\\&\quad\cdot(\lambda_i\boldsymbol{Q}_i^{e_{\text{AS}_k}(p)}\boldsymbol{\Theta}_i[\boldsymbol{Q}_i^{e_{\text{AS}_k}(q)}]^{\text{T}})\end{aligned} \tag{10}$$

通过如上分析可知,为了使现有执行体域综合异构度量最大,就需要令其相似度关联的多维异构特征评价指标最小化. 如果以该指标最小化为准则利用穷举的求解方式对域内执行体进行异构元素着色,一方面会使冗余度相同的执行体域的着色结果产生很强的倾向性(这会降低控制平面执行体中元素的综合不确定性),另一方面穷举方式在异构元素较多或维度层级较深时会显著增加求解难度. 为了解决上述问题,本节提出了随机种子预配置的功能等价执行体异构元素着色算法. 其本质思想是根据预配置的随机种子确定初始的执行体着色方案,然后以该着色方案为参照对后续执行体选取评价指标最优的着色组合,这样就能在提高着色结果不确定性的同时保证着色结果在当前随机种子下最优化. 基于随机种子预配置的异构元素着色算法具体步骤如算法2所示.

算法2 基于随机种子预配置的异构元素着色算法

Input: E_{AS} λ $\boldsymbol{\Theta}$ Γ
Output: $Q_{E_{\text{AS}_k}}$
1: for each $E_{\text{AS}_k}\in E_{\text{AS}}$ do
2: if $|E_{\text{AS}_k}|\leqslant\min\{\text{TypeNum}[\text{Dim}(i)]\}$, $i\in[1,\text{DimNum}]$ then
3: for each $e_p\in E_{\text{AS}_k}$ do
4: if $p==1$ then
5: $\Gamma_{kp}\leftarrow\Gamma$
6: for $i=1$ to DimNum do
7: $Q_{e_p}.\text{Dim}(i)\leftarrow$Randomly pick element from $\Gamma_{kp}(\text{Dim}(i))$
8: $\Gamma_{kp}(\text{Dim}(i))\leftarrow\Gamma_{kp}(\text{Dim}(i))-Q_{e_p}.\text{Dim}(i)$
9: end for
10: $\Lambda_k\leftarrow$Exhaustive search coloring results based on Γ_{kp}
11: else if $p==2$ then
12: $Y_k\leftarrow$Pick elements from Λ_k
Y_k need to satisfy the constraints Rule
Rule: $\text{Num}[Y_k(i)]=|E_{\text{AS}_k}|-1,\forall i\in[1,|Y_k|]$
$Y_k(i).[c(x).\text{Dim}(j)]\cap Y_k(i).[c(y).\text{Dim}(j)]=\phi$
$\forall i\in[1,|Y_k|],x\neq y\in[1,|E_{\text{AS}_k}|-1],j\in[1,\text{DimNum}]$
13: target$\leftarrow\arg\min_{i\in[1,|Y_k|]}\text{similarity}(Q_{e_1}\cup Y_k(i))$
14: $Q_{E_{\text{AS}_k}}\leftarrow\{Q_{e_1}\cup Y_k(\text{target})\}$
15: break
16: end if
17: end for
18: end for
19: else
20: for each $e_p\in E_{\text{AS}_k}$ do
21: $\Lambda_k\leftarrow$Exhaustive search coloring results based on Γ
22: if $p==1$ then
23: $Q_{e_p}\leftarrow$Randomly pick one element from Λ_k
24: $\Omega_k\leftarrow$Pick elements from Λ_k if $\text{similarity}(Q_{e_1}\cup\Lambda_k(i))\leqslant s_{\text{lim}}$
25: $Y_k\leftarrow$Pick elements from Ω_k
Y_k need to satisfy the constraints Rule
Rule: $\text{Num}[Y_k(i)]=|E_{\text{AS}_k}|-1,\forall i\in[1,|Y_k|]$
$\text{similarity}(Y_k(i).c(x),Y_k(i).c(y))\leqslant s_{\text{lim}}$
$\forall i\in[1,|Y_k|],x\neq y\in[1,|E_{\text{AS}_k}|-1]$
26: target$\leftarrow\arg\min_{i\in[1,|Y_k|]}\text{similarity}(Q_{e_1}\cup Y_k(i))$
27: $Q_{E_{\text{AS}_k}}\leftarrow\{Q_{e_1}\cup Y_k(\text{target})\}$
28: break
29: end if
30: end for
31: end for

2.4 动态反馈感知调度策略

通过冗余集构建策略和异构元素着色策略,本文所构建的广义控制平面(执行体域)已经具备冗余和异构属性. 但是,目前本文所构建的广义控制平面仍具有典型静态属性,执行体域映射关系及其异构元素均相对固定. 如果攻击者利用无限时间资源优势(攻防不对称性)对其进行探测,就有可能利用相关信息针对性突破冗余异构防线. 为了解决该问题,本节首先对仅具备

冗余和异构属性的执行体域建立失效概率模型,然后以失效概率最小化为目标对执行体域引入动态反馈感知调度策略.

为了合理量化攻击者探测次数与执行体可靠性的规律,本节假设两者满足 $p=e^{-\alpha n}$ 的关系. 其中,p 表示执行体可靠性(即 $1-p$ 表示攻击者的成功概率),n 为攻击者探测次数,α 为调节系数. 通过分析上述关系式可知,其满足"当探测次数越多,执行体可靠性越低,攻击者成功概率也就越高"的规律. 基于上述分析可构建如下执行体域失效概率模型.

目标函数:

$$\min \mathrm{FR}=\sum_{\mathrm{AS}_k\in \mathrm{AS}}\ \sum_{E_s\in E_{\mathrm{AS}_k}(e_{\mathrm{fail}})\geqslant\lceil(|E_{\mathrm{AS}_k}|+1)/2\rceil}\ \bigcup_{e_p,e_q\in E_s} B_f(e_p,e_q) \tag{11}$$

约束条件:

$$B_f(e_p,e_q)=\max\left\{\begin{array}{l}\bigcup\limits_{i\in\{i|E_{\mathrm{AS}_i}(e_p).\,\mathrm{Dim}(i)=E_{\mathrm{AS}_i}(e_q).\,\mathrm{Dim}(i)\}}\max(1-p_{E_{\mathrm{AS}_i}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]},\\ \qquad 1-p_{E_{\mathrm{AS}_i}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]}),\\ \bigcup\limits_{j\in\{j|E_{\mathrm{AS}_k}(e_p).\,\mathrm{Dim}(j)\neq E_{\mathrm{AS}_k}(e_q).\,\mathrm{Dim}(j)\}}\max((1-p_{E_{\mathrm{AS}_k}(e_p).\,\mathrm{type}[\mathrm{Dim}(j)]})\cdot\\ \qquad (1-p_{E_{\mathrm{AS}_i}(e_p).\,\mathrm{type}[\mathrm{Dim}(j)]}))\end{array}\right\} \tag{12}$$

$$p_{E_{\mathrm{AS}_i}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]}=e^{-\alpha N_{E_{\mathrm{AS}_k}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]}} \tag{13}$$

$$N_{E_{\mathrm{AS}_i}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]}=\sum_{k=1}^{|E_{\mathrm{AS}}|} n_{E_{\mathrm{AS}_i}}(\mathrm{ht}_{i,\arg(\mathrm{type}[\mathrm{Dim}(i)])}) \tag{14}$$

$$D_{pq}\leqslant t_{\mathrm{sw}_i},\ \forall e_p,e_q\in E_{\mathrm{AS}_k},\ \forall \mathrm{sw}_i\in\phi(E_{\mathrm{AS}_k}) \tag{15}$$

$$\sum_{k=1}^{l}|E_{\mathrm{AS}_k}|=\left|\bigcup_{i=1}^{m}\psi_{\mathrm{sw}_i}\right| \tag{16}$$

在上述模型目标函数(11)中,FR 表示执行体域全局失效率,$\mathrm{AS}_k(e_{\mathrm{fail}})\geqslant\lceil(|E_{\mathrm{AS}_k}|+1)/2\rceil$ 表示执行体域 E_{AS_k} 中失效执行体数目 e_{fail} 大于 $\lceil(|E_{\mathrm{AS}_k}|+1)/2\rceil$ 时的所有情况集合,$B_f(e_p,e_q)$ 则表示执行体域 E_{AS_k} 中执行体 e_p,e_q 同时失效概率(元失效概率). 元失效概率表达式如式(12)所示,根据两执行体对应维度的异构元素类型(异构元素类型相同的维度由 i 表示,异构元素类型不同的维度由 j 表示)确定连乘或共享最大概率形式. $E_{\mathrm{AS}_k}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]$ 表示执行体域 E_{AS_k} 中执行体 e_p 的第 i 维异构特征对应的异构元素类型,该类型的异构元素失效概率与其全局探测次数 N 的关系如式(13)所示. 对应类型的异构元素全局探测次数表达式如式(14)所示,$\arg(\mathrm{type}[\mathrm{Dim}(i)])$ 具体表示在维度 i 下异构元素类型为 $E_{\mathrm{AS}_k}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]$ 的序号,$\mathrm{ht}_{i,\arg(\mathrm{type}[\mathrm{Dim}(i)])}$ 则作为维度 i 下异构元素类型为 $E_{\mathrm{AS}_k}(e_p).\,\mathrm{type}[\mathrm{Dim}(i)]$ 的全局通用符号,$n_{E_{\mathrm{AS}_i}}$ 代表各执行体域下该类型的异构元素局部探测次数. 除此之外,该模型仍满足控制时延及执行体数量要求(详见 2.2 节),如式(15)和(16)所示.

算法 3　基于历史探测信息的反馈感知调度算法

```
1:In each time slot
2:Receive feedback information from input and output proxies
3:if the elapsed time since the last scheduling t_elapse ≥ T_threshold or
    the output of E_AS_k(e_p) is abnormal then
4:   Reset redundant sets and heterogeneous elements by algorithms 2,3
5:   Reset historical probe information statistics
6:else
7:      if E_AS_k(e_p), ∀k ∈ [1, |E_AS|], ∀p ∈ [1, |E_AS_k|] is
          detected then
8:       Update n_E_AS_k(e_p), n_E_AS_k(e_p.type[Dim(j)]), ∀j ∈ [1,
          DimNum], E_total
9:   end if
10:  for each E_AS_k ∈ E_AS do
11:    for each e_p ∈ E_AS_k do
12:      if n_E_AS_k(e_p) ≥ Threshold_e then
12:        for j = 1 to DimNum do
13:        e_p.type[Dim(j)] ← DynamicScheduling(E_total, e_p.type[Dim
             (j)])
14:        end for
15:        n_E_AS_k(e_p) ← 0
16:      else if n_E_AS_k(e_p.type[Dim(j)]) ≥ Threshold_ht_j,arg(e_p.type[Dim(j)]) then
17:        e_p.type[Dim(j)] ← DynamicScheduling(E_total, e_p.type[Dim
             (j)])
18:      end if
19:    end for
20:  end for
21:end if
22:function DynamicScheduling(E_total, e_p.type[Dim(j)])
23:  x ← arg min_x {E_total[Dim(j)].ht_j,x}
24:  if ht_j,x ≠ e_p.type[Dim(j)] then
25:      return ht_j,x
26:    else
27:      y ← arg min_y {E_total[Dim(j)] \ {ht_j,x}.ht_j,y}
28:      return ht_j,y
29:  end if
30:end function
```

动态感知调度模块可以在感知输入代理及裁决输出代理的反馈信息基础上综合处理执行体域的历史探测信息,并根据相应策略动态调度域内执行体异构元素,降低执行体域的全局失效概率. 其中,历史探测信息具体内容如表 1 所示,其主要包括两部分:一部分统计各个执行体域 E_{AS_i} 在所有维度下 $\mathrm{Dim}(j),j\in[1,\mathrm{DimNum}]$ 异构元素 $\mathrm{ht}_{j,\mathrm{TypeNum}(\mathrm{Dim}(j))}$ 的被探测次数 n 及其全局累计次数 E_{total};另一部分则统计各执行体域内 E_{AS_i} 的所有执行体 $e_p,p\in[1,|E_{\mathrm{AS}_i}|]$ 的被探测次数. 基于历史探测信息的反馈感知调度算法将结合上述全局与局部指标信息,采用实时与定期相结合的策略进行动态调度

更新,其具体步骤如算法 3 所示.

表 1 历史探测信息统计

E_{AS_i}	$\mathrm{Dim}(j), j\in[1,\mathrm{DimNum}]$	E
	$\mathrm{ht}_{j,1}\cdots\mathrm{ht}_{j,\mathrm{TypeNum}(\mathrm{Dim}(j))}$	$e_{p\in[1,\lvert E_{\mathrm{AS}_i}\rvert]}$
1	$n_{E_{\mathrm{AS}_1}}(\mathrm{ht}_{j,1})\quad\cdots\quad n_{E_{\mathrm{AS}_1}}(\mathrm{ht}_{j,\mathrm{TypeNum}(\mathrm{Dim}(j))})$	$n_{E_{\mathrm{AS}_1}}(e_p)$
…	…	…
k	$n_{E_{\mathrm{AS}_k}}(\mathrm{ht}_{j,1})\cdots n_{E_{\mathrm{AS}_k}}(\mathrm{ht}_{j,\mathrm{TypeNum}(\mathrm{Dim}(j))})$	$n_{E_{\mathrm{AS}_k}}(e_p)$

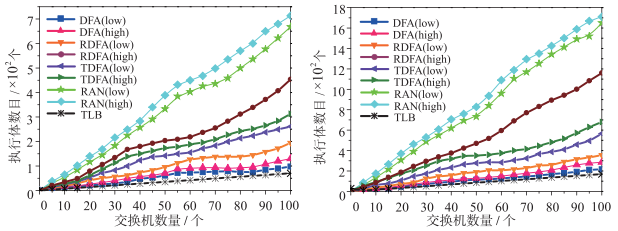
3 方案评估

为了评估内生安全控制平面有效性,本节分别就冗余集构建策略、多维异构元素着色策略和动态反馈感知调度策略等三部分内容进行实验仿真.

3.1 冗余集构建策略评估

为评估冗余集构建策略有效性,我们在服务器(配置 Intel(R) Xeon(R) CPU E5-2600 V4,主频 2.1GHz,16GB 内存,ubuntu-16.04 系统)上利用 python 2.7 networkx 库生成随机执行体网络. 其中,$N=5000$,执行体间时延 D_{ij} 设置按照高(high)、低(low)时延场景分别在区间[50ms,2000ms]和[50ms,200ms]内均匀随机分布,执行体容量 K 设置为 10;交换机时延需求 t_{SW_m} 设置在区间[50ms,2000ms]内均匀随机分布,冗余需求度 f_{SW_m} 设置同样按照高、低需求场景分别在区间[1,5]和[6,10]内均匀随机分布. 此外,为了横向对比需求优先的冗余集构建算法(DFA)有效性,本小节引入如下算法作为对照:(1)冗余需求优先算法(RDFA),相较于 DFA 算法,该算法仅优先考虑冗余需求度指标,并优先将剩余容量小的执行体分配给交换机;(2)时延需求优先算法(TDFA),与 DFA 算法不同,该算法仅以时延约束为首要考虑条件,未将冗余需求度纳入考量约束范围;(3)随机冗余集构建算法(RAN),该算法按照随机原则将满足条件的执行体与交换机建立映射;(4)理论下界分析法(TLB)[7],该算法主要根据交换机规模、交换机冗余需求与执行体负载约束等指标确定全局执行体数的理论下界值. 为了比较上述策略有效性,本节对上述策略产生的全局执行体数进行对比. 基于上述算法的冗余集构建策略在具有不同交换机规模(交换机规模[0,100])的仿真环境中产生的全局执行体数变化曲线如图 2 所示(每一结果均是对相同规模下的 20 个网络实例的平均值).

分析图 2 可知,在不同冗余集构建策略下,基于 TLB 算法产生的全局执行体数是最小的,DFA 算法产生的全局执行体数与 TLB 最为接近,而 RAN 的全局执行体数是最大的. 这是由于 DFA 算法能够综合考虑冗余度、时延和容量约束,使其在交换机规模增加时既能满足渐变扩容需求也能控制执行体数目;而 RAN 在任何交换机规模下都采用随机映射原则,不能充分提高



(a) $f_{\mathrm{SW}m}$在[1,5]均匀随机分布 (b) $f_{\mathrm{SW}m}$在[6,10]均匀随机分布

图2 不同冗余集构建策略下全局执行体数对比图

执行体利用率而导致全局执行体数目发散. 相较于其他策略,RDFA(low)在低时延环境下的全局执行体数与 DFA 最相近,这是由于冗余需求优先的算法在时延约束宽松的条件下能够提高执行体利用效率;但是当时延约束收紧时,RDFA(high)全局执行体数会随着执行体域边界选取难度增大而显著上升,这会导致其性能明显劣于 TDFA. 综上所述,DFA 能够在不同冗余需求度、不同交换机规模下控制全局执行体数目而且算法稳定性较高.

3.2 多维异构元素着色策略评估

为了定量评估该策略有效性,本节假定执行体在信息系统组成维度可细化为控制器单元、操作系统单元和处理器单元. 其中,控制器异构元素包括{ODL,ONOS,Floodlight,Ryu,NOX},操作系统异构元素包括{Win7,Win10,CentoS,Ubuntu,RedHat},处理器元素包括{Intel,AMD,IBM,Loongson,ARM}. 上述维度对应的异构特征权重系数设置为 $\lambda=\{\lambda_1,\lambda_2,\lambda_3\}=\{0.6,0.3,0.1\}$,算法相似性阈值设置为 $s_{\mathrm{lim}}=0.5$,$\boldsymbol{\Theta}_i$ 矩阵内元素根据 Beta 分布 $\mathrm{Be}(\alpha,\beta)$ 循环转置填充,参数 α,β 在区间[1,10]内随机确定. 在上述仿真环境下,本节测试多维异构元素着色策略、基于传染病模型的着色策略 HSIS[8] 和随机着色策略在不同执行域数目 $|E_{\mathrm{AS}}|\in[1,10]$ 和域内执行体数目 $|E_{\mathrm{AS}_k}|\in[3,19]$ 下的全局相似度指标(每种组合取三次测试平均值). 上述三种着色策略的全局相似度指标结果如图 3 所示.

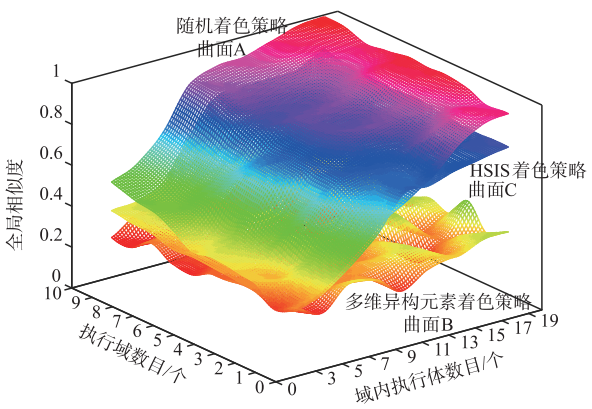


图3 不同着色策略下全局相似度指标宏观对比图

分析图 3 可知,对于随机着色策略而言,当执行域数目固定时,全局相似度指标会随着域内执行体数目增加而增加. 这是由于随机着色策略具有较大随机性,在域内执行体数较小时($|E_{AS_k}|\leqslant 5$)时,其全局相似度指标可能会与多维异构元素着色策略接近(0.3 左右),但是在域内执行体数目较大时($|E_{AS_k}|\geqslant 13$),其全局相似度指标逐渐接近 1,异构性能显著降低. 对于 HSIS 着色策略而言,由于其仅将控制器类型作为异构属性的切入点,对操作系统、处理器等维度采用默认随机着色原则,故其全局相似度变化趋势与随机着色策略相仿,但是由于 HSIS 着色策略考虑了域内控制器类型的异构化程度,因此,其全局相似度变化范围(0.25 ~ 0.75)要小于随机着色策略下全局相似度的变化范围(0.3 ~ 0.95). 而对于多维异构元素着色策略,其全局相似度指标不会随域内执行体数增加而急剧增加,仅会在可控范围波动(0.2 ~ 0.4). 这是由于该策略综合考虑域内执行体数、异构元素间相似度等多种要素保证全局相似度指标在不同随机种子下达到最优(由于随机种子不确定性,全局相似度指标会略有波动). 当然,受限于异构元素有限性,当域内执行体数很大时,该策略全局相似度指标也会略有上升,但总体仍维持在较低水平.

3.3　动态反馈感知调度策略评估

为了衡量不同调度策略对全局失效概率的影响,本节在执行域数目及域内执行体数目参数为 $|E_{AS}|=5$, $|E_{AS_k}|=11$ 下,对比动态反馈感知调度策略与以下三种调度策略的全局失效率指标:(1)静态策略,在静态策略下,各执行体域内的执行体异构元素保持不变;(2)随机调度策略,在该策略下,随机选择各执行体域内的执行体异构元素进行随机替换;(3)博弈调度策略(Mcad)[9],在该策略下,系统根据博弈过程指导控制器类型调度. 攻击频率满足参数为 10 的泊松分布. 动态反馈感知调度策略中全局更新周期的阈值 $T_{threshold}$ 设置为 25min,执行体探测次数的阈值$Threshold_e$ 统一设置为 100,异构元素探测次数阈值$Threshold_{ht}$ 统一设置为 300. 在上述仿真环境下,三种策略的全局失效率如图 4 所示.

分析图 4 可知,随着时间的推移和攻击次数不断积累,静态策略的全局失效率急剧上升,在 $t=20$min 时系统全局失效率已接近 0.9;对随机调度策略而言,由于引入随机挑选执行体进行异构元素替换,增加了执行体域不确定性,故其全局失效率指标上升趋势比静态策略缓慢. 当然,随机替换作为一种盲目的操作,只是利用随机概率事件轻微延缓了失效率的增长变化过程,并没有实质优化系统失效率,当 $t=80$min 时随机调度策略下的全局失效率已上升至 0.8,此时系统面临严重安全风险;对于博弈调度策略而言,由于系统根据博弈结果而概率调整多维异构特征中的控制器类型,调度过程具有一定针对性,因此,其全局失效率指标相对于随机策略上升趋势明显变慢,直至 $t=100$min 时系统失效率达到 0.2 左右,证明该策略具有一定安全效果;当然,相比上述三种策略,动态反馈感知调度策略能够基于历史探测信息以全局失效率最小化为目标进行多维度异构元素替换调度,其调度策略针对性更强,故随着时间的变化其全局失效率上升最为缓慢. 以时间间隔 25min 为一个评价窗口,其窗口内平均失效率增长低于 0.1,远低于其他三种策略. 此外,动态反馈感知调度策略配合全局更新周期能够周期性使攻击信息积累清零,全局失效率均从 0 开始增长,进一步提升了系统安全性. 综上所述,动态反馈感知调度策略能够降低全局失效概率,逆转攻击者时间优势造成的不对称性.

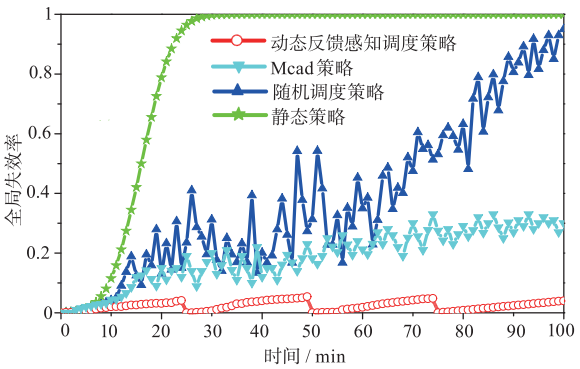


图4　不同调度策略下全局失效率对比图

4　结束语

为解决 SDN 控制平面未知漏洞威胁,本文提出了一种基于多维异构特征与反馈感知调度的内生安全控制平面的设计方案,该方案分别以拟态基因的冗余、异构和动态属性为切入点协同提升系统内生安全性. 相关仿真结果证明了该方案可以收敛全局执行体数目、增加执行体之间的多维异构度并降低系统全局失效率.

参考文献

[1] Mckeown N, et al. Openflow: enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008, 38(2): 69 – 74.

[2] Ahmad I, Namal S, Ylianttila M, et al. Security in software defined networks: A survey[J]. IEEE Communications Surveys & Tutorials, 2015, 17(4): 2317 – 2343.

[3] Scott S, Natarajan S, et al. A survey of security in software defined networks[J]. IEEE Communications Surveys & Tutorials, 2016, 18(1): 623 – 654.

[4] 邬江兴. 网络空间拟态防御研究[J]. 信息安全学报, 2016, 1(4): 1 – 10.
WU Jiang-xing. Research on cyber mimic defense[J]. Jour-

nal of Cyber Security,2016,1(4):1 – 10. (in Chinese)
[5] 王禛鹏,扈红超,程国振. 一种基于拟态安全防御的 DNS 框架设计[J]. 电子学报,2017,45(11):2705 – 2714.
WANG Zhen-peng, HU Hong-chao, CHENG Guo-zhen. A DNS architecture based on mimic security defense [J]. Acta Electronica Sinica,2017,45(11):2705 – 2714. (in Chinese)
[6] Martello S,Toth P. Lower bounds and reduction procedures for the bin packing problem[J]. Discrete Applied Mathematics,1990,28(1):59 – 70.
[7] Xie J,Guo D,Zhu X,et al. Minimal fault-tolerant coverage of controllers in IaaS datacenters[J]. IEEE Transactions on Services Computing,2017,36(3):1 – 14.
[8] Wang Z,Hu H,Zhang C. On achieving SDN controller diversity for improved network security using coloring algorithm[A]. Proceedings of the IEEE International Conference on Computer and Communications[C]. Chengdu:IEEE,2017. 1 – 5.
[9] Qi C,Wu J,Chen H,et al. Game-theoretic analysis for security of various software-defined networking (SDN) architectures[A]. Proceedings of the IEEE Vehicular Technology Conference[C]. Sydney:IEEE,2017. 1 – 5.

作者简介

王 涛 男,1993 年生于山东临朐. 现为中国人民解放军战略支援部队信息工程大学博士研究生. 主要研究方向为 SDN 安全.
E-mail:wangtaogenuine@163.com

陈鸿昶 男,1964 年生于河南新密. 现为中国人民解放军战略支援部队信息工程大学博士研究生教授、博士生导师. 主要研究方向为未来网络体系结构、人工智能等.
E-mail:chc@ndsc.com.cn