计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210500071

基于多阶段博弈的虚拟化蜜罐动态部署机制

高雅卓　刘亚群　张国敏　邢长友　王秀磊
陆军工程大学指挥控制工程学院　南京 210007
(gyz9396@163.com)

摘　要　作为一种重要的欺骗防御手段,蜜罐对于增强网络主动防御能力具有重要意义,但现有蜜罐大多采用静态部署方法,难以高效应对攻击者的策略性探测攻击等行为。为此,将完全信息静态博弈与马尔可夫决策过程相结合,提出了一种基于多阶段随机博弈的虚拟化蜜罐动态部署机制 HoneyVDep。HoneyVDep 结合攻防双方多阶段持续对抗的特点,以资源约束下防御方的综合收益最大化为目标,建立了基于多阶段随机博弈对抗的蜜罐部署优化模型,并实现了基于 Q_Learning 的求解算法,以快速应对攻击者的策略性探测攻击行为。最后,基于软件定义网络和虚拟化容器实现了一个可扩展的原型系统对 HoneyVDep 进行验证,实验结果表明,HoneyVDep 能够根据攻击者的攻击行为特征,有效生成蜜罐部署策略,提升对攻击者的诱捕率,减少部署成本。

关键词:虚拟化蜜罐;多阶段博弈;深度强化学习;软件定义网络;容器
中图法分类号　TP393.00

Multi-stage Game Based Dynamic Deployment Mechanism of Virtualized Honeypots

GAO Ya-zhuo,LIU Ya-qun,ZHANG Guo-min,XING Chang-you and WANG Xiu-lei
College of Command & Control Engineering,Army Engineering University of PLA,Nanjing 210007,China

Abstract　As an important deception defense method,honeypot is of great significance to enhance the network active defense capability. However,most of the existing honeypots adopt the static deployment method,which is difficult to deal with the strategic attacks effectively. Therefore,by combining the complete information static game with Markov decision process,we propose a multi-stage stochastic game based dynamic deployment mechanism HoneyVDep. By taking the resource constrained maximum comprehensive gain of the defensive side as the goal,HoneyVDep establishes a multi-stage random game based honeypot deployment optimization model. Besides,we also implement a Q_Learning based solution algorithm,which can deal with the attacker's strategic detection attack behavior quickly. Finally,based on software defined network and virtualization containers,we implement an extensible prototype system. The experimental results show that HoneyVDep can effectively generate honeypot deployment strategy according to the characteristics of the attacker's attack behavior,improve the trapping rate of the attackers,and reduce the deployment cost.

Keywords　Virtual honeypot,Multi stage game,Deep reinforcement learning,Software defined network,Container

1　引言

随着互联网技术的蓬勃发展,网络安全问题已经受到业内的广泛关注。传统的防御手段如防火墙、IDS 和 IPS 等只能被动地对攻击者进行检测阻断,而攻防双方信息的不对称性使得这些被动防御技术很难应对如今复杂多样的网络攻击手段。为了改善这种攻防不对称的现象,网络欺骗防御思想应运而生。面向攻击诱捕的网络欺骗防御思想最早可以追溯到 1989 年[1],传统的攻击诱捕技术通常使用模拟程序部署的低交互蜜罐或普通虚拟机实现的简单高交互蜜罐,虽然具备一定的诱捕能力,但是总体而言部署功能单一、结构僵化、灵活性也较差[2],很难适应当今复杂多变的 APT 攻击和零日漏洞攻击。

为了提高蜜罐系统的自适应性,目前已经有许多学者致力于研究如何通过蜜罐系统与攻击者进行博弈。但另一方面,蜜罐的部署策略同样会对其诱捕效果产生重要影响,主要体现在两个方面:如何在有限的成本等约束下部署最佳数量和类型的蜜罐;如何根据攻击者的攻击行为动态调整蜜罐的部署策略。

对此,本文提出了一种基于多阶段博弈的虚拟化蜜罐动

到稿日期:2021-05-12　返修日期:2021-08-12
基金项目:国家自然科学基金项目(61379149, 61772271);江苏省自然科学基金青年基金(SBK2020043435)
This work was supported by the Natural Science Foundation of China(61379149,61772271) and Natural Science Foundation of Jiangsu Province (SBK2020043435).
通信作者:邢长友(changyouxing@126.com)

态部署机制，其基本思想是：防御方持续观察攻击者的行为特征，并构建智能博弈的决策模型，通过博弈学习形成智能诱捕决策。针对诱捕策略的高效部署问题，利用容器、SDN等技术建立一种松耦合、可扩展的诱捕环境架构，可以根据智能诱捕决策的结果自适应生成和调整诱捕环境。本文的主要贡献有：

(1)针对蜜罐的动态部署决策问题，结合完全信息博弈和马尔可夫决策过程，建立了能够刻画多阶段网络攻防过程的多阶段随机博弈模型，并利用 Q_Learning 算法求解防御方的动作行为，在此基础上建立了基于多阶段博弈的虚拟化蜜罐动态部署机制 HoneyVDep。

(2)结合 SDN 和容器技术，建立了一种松耦合、可扩展的虚拟化诱捕架构，并实现了基于 HoneyVDep 的蜜罐部署策略，实验结果表明 HoneyVDep 能够在学习攻击者行为模式的基础上进行高效部署决策，提高系统的攻击诱捕效率。

本文第 2 节介绍了相关工作；第 3 节对诱捕架构中的智能诱捕决策问题进行了建模分析，并给出了基于 Q_Learning 的求解方法；第 4 节讨论了可扩展的动态虚拟化诱捕架构设计，并基于前述研究成果构建了原型系统，然后进行了实验测试；最后总结全文，并对后续研究进行了展望。

2 相关工作

“蜜罐(Honeypot)”被定义为“是一种价值在于被扫描、攻击和攻陷的安全资源[3]”，其主要思想是通过诱导攻击者对蜜罐发起攻击，来欺骗攻击者与蜜罐进行交互，以达到检测并分析攻击行为的目的。由于传统的蜜罐诱捕系统存在功能单一、交互能力不足等问题，难以应对繁杂多样的攻击手段，因此近年来的诱捕系统越来越倾向于与博弈论和机器学习等方法相结合来体现智能诱捕的思想。

在与机器学习结合方面，Karnel[4] 提出了一种基于机器学习聚类思想的算法，用于在诱捕点中辨认攻击者，并将结果用于配置后期的防御策略；另一种更加常见的方法是使用强化学习的思想进行智能诱捕，这种将强化学习加入诱捕系统的思想最早来自于 2011 年提出的机制 Heliza[5]；Pauna 等将 Cowrie 蜜罐与 DQN 思想[6] 相结合，实现了一种自适应的智能 SSH 蜜罐；除了 Heliza 的建模方法，SMDP[7] 提出将马尔可夫决策过程的方法应用于攻击诱捕中，把连续时间过程转化为等效的离散决策模型，并使用强化学习对该模型进行训练，最后得到了规避风险、提高成本效益和时间效益的最优策略。

在与博弈论相结合的方面，Boumkheld 等[8] 将攻防间的交互建模成不完全信息动态博弈，主要利用博弈论帮助防御者对不同级别的防御系统进行选择，推导出完美贝叶斯纳什均衡解决方案，求解不同参数下防御方和攻击者的最佳动作参数；Aliou[9] 利用博弈论对蜜罐的分配进行决策，减小攻击者发现蜜罐的概率；Attiah 等[10] 对攻防双方进行了多层次的策略建模，双方都根据策略的成本、潜在的攻击收益或损害以及预测对手策略的有效性来调整自己的策略，最终得到混合策略纳什均衡；Ahmed 等[11] 还将博弈论与攻击图相结合，判断在已知当前攻击者位置的情况下接下来多跳诱捕点的分配策略。然而，这些博弈论与蜜罐系统的结合大多只停留在单阶段博弈，但是蜜罐系统内部的网络攻防博弈具有多阶段特征，因此与马尔可夫决策过程结合的随机博弈[12] 相对更加符合网络攻防的场景。

除了单独使用博弈论或机器学习，还可以将两者共同结合到网络攻防的建模和求解中，Zhang 等[13] 针对攻击者进入真实系统内部后的攻防过程建立了一个不完全信息随机博弈的模型，并使用 Q_Learning 算法进行求解，但该博弈决策重点关注于攻击者入侵到网络内部后的诱捕动作。相对而言，本文更加关注针对攻击探测阶段蜜罐部署策略的研究，且基于 Q_Learning[14] 给出了最终的部署建议。

另外，随着网络技术的发展，传统的攻击诱捕架构开始与新兴技术相结合，在架构设计和实现上展现了许多新思路，如使用比虚拟机更加轻量级的容器技术[15-16] 来部署蜜罐和使用 SDN[17] 技术对网络流量进行定向传输等。文献[18]针对 IoT 中特定的僵尸网络漏洞，使用 Docker 开发了一种模拟 UPnP 路由设备的中高交互蜜罐，该蜜罐系统在 2019 年已经稳定运行大半年，并提供了许多日志材料；Wang 等[19] 提出了一种基于多阶段攻击的 SDN 动态蜜罐 SDHG，使用不完全信息动态博弈对不同阶段的攻防策略进行建模，并证明了模型的可行性，最后使用 Docker 容器对原型系统进行了实现；在流量重定向的过程中，由于 TCP 连接在会话转移之后需要重新建立，连接中断的异常可能会被高级的攻击者发觉，因此，为了进一步降低被攻击者发现的几率，HoneyDOC[20] 提出了一种与 SDN 技术结合的 TCP 重放机制，可以将攻击者流量进行无缝迁移，迁移之后无需重新建立连接。

3 智能诱捕决策

3.1 概述

攻击者在对网络发起攻击之前会采用多种手段对目标系统进行探测，获取在线主机的情况和地址，随后根据自身具备的攻击手段针对某个特定的主机进行攻击，而防御方在检测到攻击者的行为后，也将根据自身具备的诱捕技术对诱捕方案进行调整。因此，如何根据所观察到的攻击者行为，动态调整诱捕环境的部署策略是一个需要解决的问题。本文重点针对这一问题，综合博弈模型和轻量级容器架构，建立一个支持动态部署的攻击诱捕机制，通过与攻击者进行多阶段持续博弈来形成诱捕策略，并维护一个基于轻量级虚拟化容器的蜜罐库，根据诱捕策略快速部署和调整蜜罐的部署场景，从而增强对攻击者的诱捕能力。

图 1 给出了基本的模型架构，其中在 $t=k$ 时刻，网络中共有 3 台真实业务服务器(S_1，S_2，S_3)，其 IP 地址分别为 IP_1，IP_w，IP_n，并部署了 3 个不同类型的蜜罐(H_1，H_2，H_3)，其 IP 地址分别为 IP_2，IP_{w+1} 以及 IP_{w+2}。攻击者按照一定的策略对网络进行攻击，防御方利用入侵检测系统、蜜罐诱捕结果等手段检测攻击行为，并根据攻击行为调整真实业务服务器和蜜罐的部署，在 $t=k+1$ 时刻，3 台真实业务服务器的 IP 地址分别为 IP_1，IP_2 和 IP_n，而 4 个蜜罐的地址分别为 IP_w，IP_{w+1}，IP_{w+2} 以及 IP_{n+1}，其中对应类型 3 的真实业务服务器有两个蜜罐。

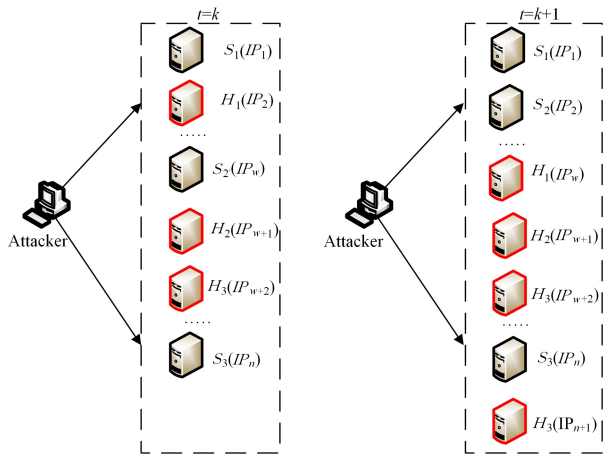


图 1 诱捕系统部署情况变化

Fig. 1 Changes in deployment of decoy system

在诱捕系统内部,防御方可以调整蜜罐的类型、数量和地址,其中蜜罐的类型与攻击者的攻击手段相匹配,通过动态改变蜜罐的部署策略,掩藏真实主机的位置和类型,诱导攻击者对蜜罐发起攻击。

在建模之前,首先对模型进行如下假设:

(1)地址空间、攻击者和蜜罐的类型有限,即攻防双方可操作的动作空间有限;

(2)攻防双方都是理性的,会选择能够将自己的利益最大化的动作;

(3)尽管防御方无法获得攻击者的全部信息,但在模型训练和模型应用的过程中,网络的威胁程度以及对攻击者的吸引力不会发生剧烈的变化。

3.2 攻防博弈模型

定义 1 完全信息随机博弈模型为(N,S,A,D,γ,π,R)。

$N=\{attacker,defender\}$为参与博弈的局中人,$attacker$表示攻击者,$defender$表示防御方。

$S=\{\boldsymbol{s}_1,\boldsymbol{s}_2,\cdots,\boldsymbol{s}_n\}$为随机博弈状态集合,每个博弈状态代表一个网络状态,可以使用一个向量表示,即$\boldsymbol{s}_k=(t_1,t_2,\cdots,t_m)$,$m$为节点的地址空间。

$A=\{\boldsymbol{A}_1,\boldsymbol{A}_2,\cdots,\boldsymbol{A}_n\}$为攻击者的动作空间,其中每个动作代表一个动作序列,可以用向量表示,即$\boldsymbol{A}_k=(x_1,x_2,\cdots,x_m)$,$m$为节点的地址空间。

$D=\{\boldsymbol{D}_1,\boldsymbol{D}_2,\cdots,\boldsymbol{D}_n\}$为防御方的动作空间,其中每个动作代表一个动作序列,可以用向量表示,即$\boldsymbol{D}_k=(y_1,y_2,\cdots,y_m)$,$m$为节点的地址空间。

$\gamma\in(0,1)$表示防御方蜜罐被发现的概率,当一个蜜罐内部存在多个攻击会话时,该蜜罐被发现的概率也会相应增加,因此在网络威胁程度较大时,每个蜜罐都应有γ的概率被发现,这里的γ随蜜罐内部攻击者的数量变化而变化。另外,如果攻击者的能力在蜜罐诱骗能力之上,则该类型的蜜罐会有一个固定的概率被发现。

π为策略函数,定义$\pi_a:S\to A$为攻击者在每个状态下选择的混合策略;$\pi_a(\boldsymbol{s}_i)$表示攻击者在状态$\boldsymbol{s}_i$下的混合策略;$\pi_d:S\to D$为防御方在每个状态下选择的混合策略;$\pi_d(\boldsymbol{s}_i)$表示防御方在状态$\boldsymbol{s}_i$下的混合策略。

$R=(R_a,R_d)$是蜜罐攻防双方的奖励,如果$R_a=-R_d$,则表示该博弈为零和博弈。R的取值与攻防双方的动作和当前的状态有关,$R_a(\boldsymbol{s}_k,\boldsymbol{A}_k,\boldsymbol{D}_k)$表示在$\boldsymbol{s}_k$状态下,攻击者采取动作$\boldsymbol{A}_k$、防御方采取动作$\boldsymbol{D}_k$时攻击者的得失;$R_d(\boldsymbol{s}_k,\boldsymbol{A}_k,\boldsymbol{D}_k)$表示在$\boldsymbol{s}_k$状态下,攻击者采取动作$\boldsymbol{A}_k$、防御方采取动作$\boldsymbol{D}_k$时防御方的得失。

(1)攻防策略

在博弈状态$\boldsymbol{s}_k$下,攻击者或防御方采取动作的规则被称为该状态下的攻防策略,此攻防策略是一个混合策略,表示在此状态下,攻击者和防御方采取每个不同动作的概率分布,如$\pi_a(\boldsymbol{s}_i)=(\sigma_{a1}(\boldsymbol{s}_i,\boldsymbol{A}_1),\sigma_{a2}(\boldsymbol{s}_i,\boldsymbol{A}_2),\cdots,\sigma_{an}(\boldsymbol{s}_i,\boldsymbol{A}_n))$,$\sigma_{ai}(\boldsymbol{s}_i,\boldsymbol{A}_1)$表示在$\boldsymbol{s}_i$状态下,攻击者选择动作$\boldsymbol{A}_1$的概率,且所有动作概率之和为1;同理,$\pi_d(\boldsymbol{s}_i)=(\sigma_{d1}(\boldsymbol{s}_i,\boldsymbol{D}_1),\sigma_{d2}(\boldsymbol{s}_i,\boldsymbol{D}_2),\cdots,\sigma_{dn}(\boldsymbol{s}_i,\boldsymbol{D}_n))$,$\sigma_{di}(\boldsymbol{s}_i,\boldsymbol{D}_1)$表示在$\boldsymbol{s}_i$状态下,防御方选择动作$\boldsymbol{D}_1$的概率,且所有动作概率之和为1。

攻防策略是攻防双方选择动作的规则,而攻防动作则是特定的动作,每个动作都是针对某一个地址空间的操作,如攻击者的动作可以设为$\boldsymbol{A}_k=(x_1,x_2,\cdots,x_m)$,表示攻击者对整个地址段的操作,其中$x_i=\{none,attack_1,attack_2,\cdots,attack_n\}$表示攻击者对地址$i$的操作,$none$表示不进行攻击,$attack_i$表示使用$i$类型的攻击;同理,防御方的动作同样可以设为$\boldsymbol{D}_k=(y_1,y_2,\cdots,y_m)$,$y_i$表示对第$i$个地址的操作,其中$y_i=\{none,host_1,host_2,\cdots,host_l,honeypot_1,honeypot_2,\cdots,honeypot_n\}$。需要注意的是,如果该地址已经布置了蜜罐,则防御方执行$none$操作后,该地址仍保持以前的蜜罐类型;如果执行了$honeypot_i$操作,则表示将该地址的蜜罐类型转换为$honeypot_i$,如果执行了$host_i$操作,则表示将该地址转换为主机$host_i$。

这里的$attack_i$与$host_i$和$honeypot_i$成对应关系,如果攻击者针对一个已经部署了$honeypot_i$的地址进行$attack_i$操作,则该攻击者将被捕获,地址状态变为$capture$;如果攻击者针对一个已经部署了$host_i$的地址进行$attack_i$操作,则攻击者攻击成功。

(2)攻防随机博弈状态

博弈状态指当前蜜罐系统内部所有蜜罐的配置状态,包括每个蜜罐的配置和网络状态,蜜罐的配置包括蜜罐的地址、种类以及资源的配置情况,蜜罐的网络状态主要包括每个蜜罐与其他蜜罐的连接状态以及与攻击者的交互情况。

博弈开始之前,防御方会首先初始化一些蜜罐,将这些蜜罐部署在真实主机的附近,攻击者只能检测到在线的主机以及这些主机的类型,但是并不知道哪些是蜜罐,哪些是真实主机,攻击者根据策略选择部分地址进行攻击,一段时间后,防御方会对未捕获蜜罐的主机地址进行变换,这就是一个博弈阶段,此阶段反复多次后,形成多阶段的随机博弈过程。

博弈状态可以用一个序列来表示,如$\boldsymbol{s}_k=(t_1,t_2,\cdots,t_m)$,$m$表示整个地址空间,$t_i$表示地址$i$的部署状态,对于$t_i$有:

$$t_i=\{none,host_1,host_2,\cdots,host_l,honeypot_1,honeypot_2,\cdots,honeypot_n,capture\}$$

其中,$none$表示未进行部署,$host_i$表示部署的是第i种类型的真实主机,$honeypot_i$表示部署的是第i种类型的蜜罐,$cap-$

ture 表示已经捕获到攻击者。

如图 2 所示，依据博弈模型建立一个时间片的攻防博弈树，在状态$\boldsymbol{s}_{k-1}$下，防御方将根据自己的策略，以一定的概率选定某个动作$\boldsymbol{D}_i$，攻击者并不知道防御方的策略，同样根据自己的判断选择某个动作$\boldsymbol{A}_i$，经过双方的动作以后，攻防双方都将得到一个自己的阶段性奖励，之后进入下一个状态。

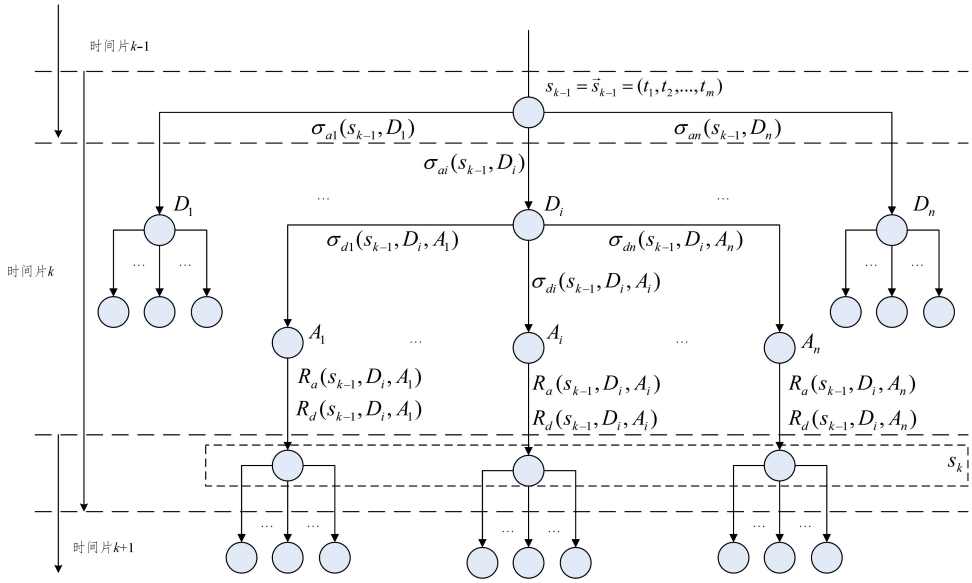


图 2　多阶段博弈中每个阶段的攻防博弈树

Fig. 2　Game tree of each stage in multistage game

(3)量化策略收益

定义 2　防御方收益 *DG* 指防御方使用正确的蜜罐配置，并吸引到针对该类型的攻击者后获得的收益。因此，可以将 *DG* 定义为 $DG=\sum_{i=1}^{m} dg_{i\{t_i=x_i\cap t_i\in honeypot\}}$，$m$ 表示地址空间，$dg_{i\{t_i=x_i\cap t_i\in honeypot\}}$ 表示当在地址 i 处攻击者选择了与蜜罐对应的攻击方式时，防御方捕获到攻击者的收益。

定义 3　防御方成本 *DC* 指防御方部署蜜罐的成本，配置不同类型的蜜罐所消耗的资源情况不同。因此，可以将 *DC* 定义为 $DC=\sum_{i=1}^{m} dc_{i\{y_i\neq none\}}$，$m$ 表示地址空间，$dc_{i\{y_i\neq none\}}$ 表示防御方在地址 i 处需要进行部署，dc_i 表示在该位置部署蜜罐的成本。

定义 4　防御方被攻击后的损失 *LOSS* 指当攻击者攻击到正常主机时，防御方受到的损失 $LOSS=\sum_{i=1}^{m} loss_{i\{t_i=x_i\cap t_i\in host\}}$，$m$ 表示地址空间，$loss_{i\{t_i=x_i\cap t_i\in host\}}$ 表示当攻击者攻击成功后防御方的损失。

定义 5　防御方奖励R_d指防御方在一个博弈阶段结束后获得的总收益。根据前文可以将其定义为：

$$R_d=DG-DC-LOSS$$

$$=\sum_{i=1}^{m}(dg_{i\{t_i=x_i\cap t_i\in honeypot\}}-dc_{i\{y_i\neq none\}}-loss_{i\{t_i=x_i\cap t_i\in host\}})$$

定义 6　攻击者收益 *AG* 指攻击者在成功对主机进行攻击后的收益，即：

$$AG=LOSS=\sum_{i=1}^{m} loss_{i\{t_i=x_i\cap t_i\in host\}}$$

定义 7　攻击者成本 *AC* 指攻击者进行攻击的成本，执行不同手段的攻击所消耗的资源情况不同。因此，可以将 *AC* 定义为 $AC=\sum_{i=1}^{m} ac_{i\{x_i\neq none\}}$，$m$ 表示地址空间，$ac_{i\{x_i\neq none\}}$ 表示攻击者对地址 i 处发起攻击，ac_i 表示在该位置进行攻击的成本。

定义 8　攻击者奖励R_a指攻击者在一个博弈阶段结束后获得的总收益。根据前文可以将其定义为：

$$R_a=AG-AC$$

$$=\sum_{i=1}^{m}(loss_{i\{t_i=x_i\cap t_i\in host\}}-ac_{i\{x_i\neq none\}})$$

3.3　HoneyVDep 求解算法

Q-Learning 算法中 Q 的计算与状态 $\boldsymbol{s}$、智能体选择的动作和智能体的奖励有关。针对防御方而言，在状态$\boldsymbol{s}_k$的下，防御方选择动作$\boldsymbol{D}_k$，Q 的计算式如下：

$$Q(\boldsymbol{s}_k,\boldsymbol{D}_k)=(1-\alpha)\cdot Q(\boldsymbol{s}_k,\boldsymbol{D}_k)+\alpha(R+\gamma\cdot\max_d Q(\boldsymbol{s}_{k+1},\boldsymbol{D}_{k+1}))$$

根据 Q_Learning 算法的架构，对博弈过程进行求解，HoneyVDep 算法如算法 1 所示。

算法 1　HoneyVDep 求解算法

输入：博弈模型，累计奖励折扣因子 γ，学习效率 α

输出：防御动作与攻击动作的对应关系(D↔A)

1. 初始化攻击者动作空间 $A=\{A_1,A_2,\cdots,A_n\}$
2. 初始化防御方动作空间 $D=\{D_1,D_2,\cdots,D_n\}$
3. For A_i in A：
4.　初始化网络状态 $s=s_0$；
5.　For episode=1，M：
6.　　While s!=s_{end}：
7.　　防御方利用 ε-greedy 策略选取防御动作D_t或者直接选择 $D_t=\max_d Q(s_t,D;\omega)$；
8.　　防御方执行动作D_t，获得奖励 $r_t=R_d(s_t,D_t)$；
9.　　$Q(s_t,\boldsymbol{D}_t)=(1-\alpha)\cdot Q(s_t,\boldsymbol{D}_t)+\alpha(R+\gamma\cdot\max_d Q(s_{t+1},\boldsymbol{D}_{t+1}))$
10.　　进入下一状态 $s=s_{t+1}$；
11.　End for
12. End for

4 原型系统实验及结果分析

4.1 原型系统实现

为了验证 HoneyVDep 的性能，本文基于 SDN 和容器技术实现了相应的原型系统，其主要分为 3 个部分，代理服务器作为一个 SDN 控制器，对所有流经交换机的出入站流量进行检测，根据检测结果对诱捕环境和流量转发进行决策下发；诱捕环境中主要有两类容器服务器，分别维护蜜罐容器和真实业务容器，每个服务器上都可以运行多个容器或服务。系统架构如图 3 所示。

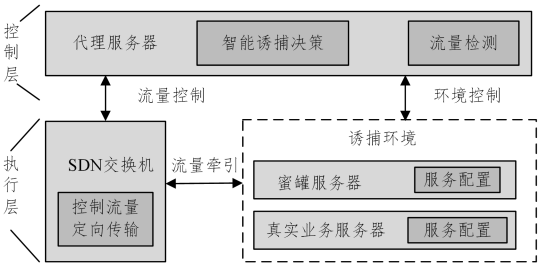


图 3 原型系统架构图

Fig. 3 Prototype system architecture

代理服务器使用 Centos 系统，CPU 为 4 核 4 线程、i5-3470、3.2GHz，内存为 4GB，硬盘为 512GB；诱捕环境内部的服务器使用 Ubuntu 系统，CPU 为 10 核 20 线程、i9-10900X、3.7GHz，内存为 64GB，硬盘为 1T。蜜罐和真实系统的容器分别部署在诱捕环境内部的两台服务器上，SDN 交换机采用 Pica8 P3297 交换机。

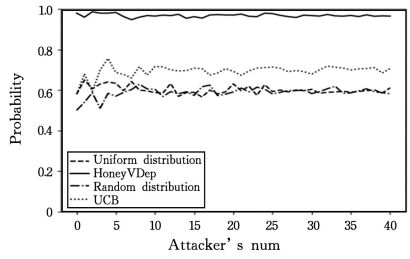
4.2 性能评估分析

为了提高攻击效率，攻击者往往遵循一定的策略来发现攻击目标。如本地偏好策略中，攻击者以较大的概率选择其所在子网空间进行探测，发现攻击目标后，下一个攻击点将主要在初次攻击地址的附近进行选择[21]。本部分实验将主要针对本地策略攻击模式进行，学习攻击者前期的行为模式并对其之后的行为进行预测，博弈主要包括两种情况：若攻击者进入蜜罐，则该次攻击行为被捕获，博弈终止，防御方获得收益；若攻击者攻击到真实主机，则攻击方得到收益，防御方受到损失，随后攻击者可以根据自身策略继续选择下一个攻击目标，开始新的博弈过程。

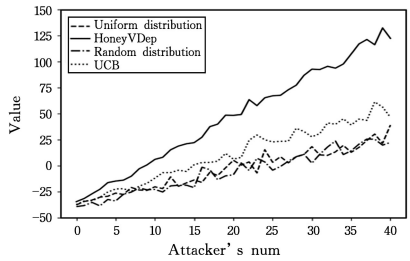
场景 1 智能决策效果评估

本部分实验主要分析在蜜罐数量固定的情况下，防御方能否在线学习攻击者的攻击行为，并根据学习结果动态调整蜜罐部署策略。实验过程中，攻击者采用本地优先的攻击策略，即攻击者随机对目标网络空间进行扫描，若发现节点则对其进行攻击，当成功攻击一个目标节点后，以更高的概率在该节点周围选择新的目标节点。实验过程中将蜜罐数量固定为 40，主机数量固定为 20，这主要是考虑了容器总数与地址空间的比例，攻击者的数量作为变量由 0 递增至 40，选择了 FTP，SQL 和 Web 这 3 种服务类型的蜜罐，根据不同类型蜜罐的镜像大小、CPU 占比和内存占比，设置部署蜜罐的成本分别为 0.5，1.5，1。根据 CVSS 的漏洞评分规则，综合漏洞的影响度和攻击复杂度，分别设置攻击者攻陷真实服务器的防御方损失为 5，15，10，以及捕获到攻击者后的收益为 3，5，4。另外，对文献[22]提出的 UCB 算法进行部分改进，对比分析其学习攻击者行为并部署蜜罐的能力。此外，还实现了蜜罐部署类型和位置均随机选择的随机部署策略，以及将蜜罐类型和部署位置均匀排列的均匀部署策略。最后，将本文算法得出的策略与这 3 种策略进行对比。

图 4(a)给出了不同蜜罐部署策略下的诱捕率，诱捕率指被捕获的攻击者数量占所有攻击者数量的百分比，图中横坐标为攻击者的数量，纵坐标为诱捕率，实线表示使用本文算法得到的策略，可以看出 HoneyVDep 得到的策略诱捕率在 4 种策略中是最高的。图 4(b)给出了对 4 种情况的收益值进行评估的结果，横坐标为攻击者的数量，纵坐标为防御方的平均收益值，实线表示使用 HoneyVDep 算法得到的策略，可以看出 HoneyVDep 得到的策略收益值同样是最高的。综上可以发现，HoneyVDep 智能决策的结果具有较高的诱捕能力。



(a)诱捕率对比



(b)收益值对比

图 4 HoneyVDep 算法的效果评估

Fig. 4 Effect evaluation of HoneyVDep algorithm

场景 2 蜜罐最佳部署数量评估

接下来将对蜜罐的最佳部署数量进行测试。由于网络的威胁程度不会发生剧烈的变化，因此攻击者的数量在一段时间内也不会急剧地增加或减少，故可以根据网络运行一段时间内攻击者的数量大致估算出模型需要应对的攻击者数量。

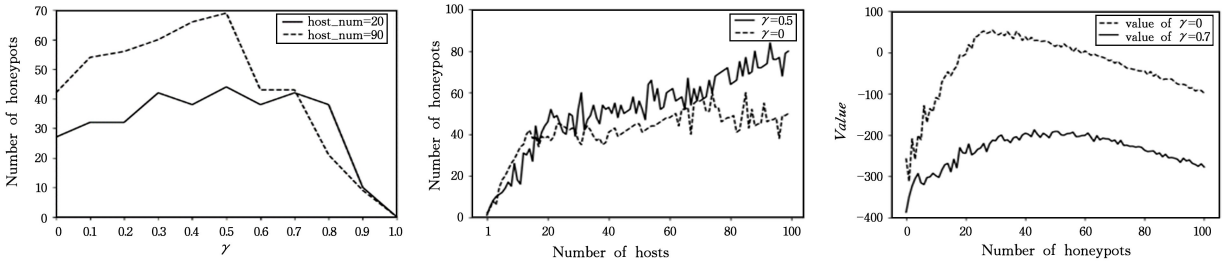
考虑到蜜罐可能会存在一定的概率被攻击者发现，因此首先分析不同的蜜罐被发现概率对蜜罐部署策略的影响。在实验中分别考虑了网络中真实主机数量较多(90 台主机)和较少(20 台主机)两种情况，攻击者的数量固定为 50。图 5(a)给出了在不同的被发现概率下蜜罐最佳部署数量的情况。可以发现，无论真实主机数量较多还是较少，随着被发现概率的增加，蜜罐最佳部署数量均呈现先上升再下降的趋势，在被发现概率为 0.5 左右时达到最大值。其原因在于，初始情况下被发现概率较低时，为了达到更好的诱捕效果，需要增加蜜罐部署的数量，但随着被发现概率的增加，部署的大量蜜罐均会被攻击者发现，带来的收益难以抵消增加部署的成本，在极端情况下，当被发现概率为 1 时，所有的蜜罐都

会被发现，不会产生任何收益，这时最佳策略就是不再部署任何蜜罐。

图5(b)给出了蜜罐部署数量随需要保护的主机数量变化而变化的情况。根据图5(a)中蜜罐数量随被发现概率的变化情况，这里设置了两种典型的概率，即 $\gamma=0$ 和 $\gamma=0.5$，如图5(b)所示，前者代表了蜜罐不会被发现的情况，后者代表了需要部署最大数量蜜罐的情况，实验中将攻击者数量固定为50，攻击者采用本地优先策略进行攻击目标的选择，根据部署成本和收益的综合情况来计算需要在网络中部署的最佳蜜罐数量，横坐标表示真实主机的数量，纵坐标表示部署蜜罐的数量。

从单条线的斜率变化可以看出，蜜罐数量相对于主机数量的比例随着主机数量的增加而减小，在主机数量小于20时斜率为2左右，主机数量大于20后斜率逐渐降低；从两条线的整体结果可以看出，当蜜罐被发现概率增加且主机数量大于20后，蜜罐数量的增加量有明显的减少，说明被发现概率会影响蜜罐的部署数量。

图5(c)给出了收益值随蜜罐数量的变化情况，主机数量固定为20，攻击者数量固定为50，可以看到收益值的整体趋势是先上升再下降，说明前期的收益主要是通过诱捕攻击者得到的，后期的损失则主要来自部署蜜罐的成本；在增加蜜罐的被发现概率后，整体的收益值有所下降。



(a)蜜罐数量随被发现概率的变化 (b)蜜罐数量与主机数量的关系 (c)蜜罐数量与收益值的关系

图5 蜜罐数量评估

Fig. 5 Honeypot quantity evaluation

结束语 作为一种欺骗防御的手段，蜜罐能够有效增加攻击者的攻击难度，提升对核心资产的保护能力。然而，当前研究者的关注焦点主要在于如何提升蜜罐的逼真度等方面，对于蜜罐的部署策略缺乏深入的分析。事实上，部署策略的优劣将直接影响蜜罐的防护效果。对此，本文结合攻击者的攻击目标选择行为，研究了相应的蜜罐动态部署策略，并建立了基于多阶段博弈的部署模型HoneyVDep，给出了Q_Learning求解算法。此外，为了增强蜜罐部署调整的灵活性，还提出了基于SDN和虚拟化容器的原型系统模型，支持根据策略高效调整部署。实验结果显示，HoneyVDep能够在不断学习攻击者行为的基础上，形成优化的部署策略，提升对攻击者的诱捕率，降低部署成本。

参 考 文 献

[1] STOLL C. The cuckoo's egg: Tracking a spy through the maze of computer espionage [M]. London: The Bodley Head Ltd, 1989.

[2] SHI L, LI Y, MA M. Latest Research Progress of Honepot Technolog[J]. Journal of Electronics & Information Technology, 2019, 41(2): 498-508.

[3] SPITZNER L. Honeypots: Tracking hackers [M]. Addison-Wesley Reading, 2003.

[4] KAMEL N E, EDDABBAH M, LMOUMEN Y, et al. A smart agent design for cyber security based on honeypot and machine learning[J]. Security and Communication Networks, 2020, 9(8): 1-9.

[5] WAGENER G, STATE R, DULAUNOY A, et al. Heliza: Talking dirty to the attackers[J]. Journal in Computer Virology, 2011, 7: 221-232.

[6] PAUNA A, IACOB A C, BICA I. Qrassh-a self-adaptive ssh honeypot driven by q-learning[C]// 2018 International Conference on Communications (COMM). 2018: 441-446.

[7] HUANG L, ZHU Q. Adaptive Honeypot Engagement Through Reinforcement Learning of Semi-Markov Decision Processes [C]// Decision and Game Theory for Security (GameSec 2019). 2019: 196-216.

[8] BOUMKHELD N, PANDA S, RASS S, et al. Honeypot type selection games for smart grid networks[C]// Conference on Decision & Game Theory for Security. Vienna, Austria: Springer International Publishing, 2019: 85-96.

[9] SARR A B, ANWAR A H, KAMHOUA C, et al. Software diversity for cyber deception[C]// IEEE Global Communications Conference. 2020: 1-6.

[10] ATTIAH A, CHATTERJEE M, ZOU C C. A game theoretic approach to model cyber attack and defense strategies[C]// 2018 IEEE International Conference on Communications (ICC). 2018: 1-7.

[11] ANWAR A H, KAMHOUA C A, LESLIE N. Honeypot allocation over attack graphs in cyber deception games[C]// ICNC, USA. IEEE, 2020.

[12] FILAR J, VRIEZE K. Competitive markov decision processes [M]. Competitive Markov Decision Processes, 1996.

[13] ZHANG H, YANG J, ZHANG C. Defense decision-making method based on incomplete information stochastic game and Q-learning[J]. Journal on Cmmunications, 2018, 39(8): 56-68.

[14] WATKINS C J C H, DAYAN P. Technical note: Q-learning[J]. Machine Learning, 1992, 8(3/4): 279-292.

[15] SOLTESZ S, PÖTZL H, FIUCZYNSKI M E, et al. Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors[J]. ACM SIGOPS Operating Systems Review, 2007, 41: 275-287.

[16] MERKEL D. Docker: Lightweight linux containers for consistent development and deployment [J]. Linux Journal, 2014.

https://dl.acm.org/doi/10.5555/2600239.2600241.
[17] NICK F, JENNIFER R, ELLEN Z. The road to SDN: An intellectual history of programmable networks [C]// ACM SIGCOMM Computer Communication Review. 2014: 87-98.
[18] ZHANG W, ZHANG B, ZHOU Y, et al. An iot honeynet based on multiport honeypots for capturing iot attacks[J]. IEEE Internet of Things Journal, 2020, 7(5): 3991-3999.
[19] WANG J, YANG H, FAN C. A SDN Dynamic Honeypot with Multi-phase Attack Response [J]. Netinfo Security, 2021, 21(1): 27-40.
[20] FAN W, DU Z, SMITH-CREASEY M, et al. Honeydoc: An efficient honeypot architecture enabling all-round design[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(3): 683-697.
[21] XING J, YANG M, ZHOU H, et al. Hiding and trapping: A deceptive approach for defending against network reconnaissance with software-defined network[C]// 2019 IEEE 38th International Performance Computing and Communications Conference (IPCCC). London, United Kingdom: IEEE, 2019: 1-8.
[22] GUTIERREZ M, KIEKINTVELD C. Online learning methods for controlling dynamic cyber deception strategies [C]// Adaptive Autonomous Secure Cyber Systems. 2020: 231-251.

GAO Ya-zhuo, born in 1998, master's degree. Her main research interests include cyberspace security, and so on.

XING Chang-you, born in 1982, Ph.D, associate professor. His main research interests include software defined network, network measurement.