计算机科学
COMPUTER SCIENCE

基于双向蚁群算法的网络攻击路径发现方法

高文龙，周天阳，朱俊虎，赵子恒

引用本文

高文龙，周天阳，朱俊虎，赵子恒. 基于双向蚁群算法的网络攻击路径发现方法[J]. 计算机科学, 2022, 49(6A): 516-522.

GAO Wen-long, ZHOU Tian-yang, ZHU Jun-hu, ZHAO Zi-heng. Network Attack Path Discovery Method Based on Bidirectional Ant Colony Algorithm[J]. Computer Science, 2022, 49(6A): 516-522.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210500072

基于双向蚁群算法的网络攻击路径发现方法

高文龙　周天阳　朱俊虎　赵子恒
战略支援部队信息工程大学　郑州 450001
(2396833257@qq.com)

摘　要　在渗透测试领域,进行攻击路径发现对实现攻击自动化具有重要意义。现有的攻击路径发现算法大多适用于静态全局环境,且存在因状态空间爆炸导致求解失败的问题。为解决动态网络环境下的攻击路径发现问题,提高路径发现效率,提出了基于双向蚁群算法的网络攻击路径发现方法(Attack Path Discovery-Bidirectional Ant Colony Algorithm,APD-BACO)。首先,对网络信息进行建模表示,定义攻击代价;然后,提出一种新的双向蚁群算法进行攻击路径发现,主要的改进包括不同的搜索策略、交叉优化操作和新的信息素更新方式等,仿真实验验证了改进的质量和效率,同时与其他路径发现方法进行对比,结果表明所提方法在较大网络规模下具有一定的时间或空间优势。在攻击路径主机发生故障时,采用重规划机制实现局部区域的攻击路径发现,更适合实际自动化渗透测试下的攻击路径发现。

关键词:攻击路径发现;双向蚁群算法;重规划;自动化渗透测试;动态环境
中图法分类号　TP393

Network Attack Path Discovery Method Based on Bidirectional Ant Colony Algorithm

GAO Wen-long,ZHOU Tian-yang,ZHU Jun-hu and ZHAO Zi-heng
Information Engineering University,Zhengzhou 450001,China

Abstract　In the field of penetration testing,the discovery of attack paths is of great significance to the realization of attack automation. Most of the existing attack path discovery algorithms are suitable for static global environments,and there is a problem that the solution fails due to the explosion of the state space. To solve the problem of attack path discovery under dynamic network environment and improve the efficiency of path discovery,a method of network attack path discovery based on bidirectional ant co-lony algorithm is proposed. First,model the network information and define the attack cost. Then,a new two-way ant colony algorithm is proposed for attack path discovery. The main improvements include different search strategies,cross-optimization operations and new pheromone update methods,etc. Simulation experiments verify the improved quality and efficiency. At the same time,compared with other path discovery methods,it has a certain time or space advantages in large network scale. When the attack path host fails,the re-planning mechanism is used to realize the attack path discovery in the local area,which is more suitable for attack path discovery under actual automated penetration testing.

Keywords　Attack path discovery,Bidirectional ant colony algorithm,Re-planning,Automated penetration testing,Dynamic environment

1　引言

随着网络规模的扩大和技术的快速发展,出现了越来越多的安全问题。渗透测试作为一种受控的模拟入侵行为,其通过对漏洞的积极利用,可以发现潜在的安全缺陷或系统配置问题。传统的渗透测试主要由渗透测试团队手工执行,依赖特定领域的专家经验,当网络规模较大时会消耗大量人力和时间。自动化渗透测试通过前期的攻击路径发现,可以使用户摆脱为目标网络主机选择合适的漏洞利用程序的复杂操作,通过自动化脚本驱动 Metasploit 等漏洞利用工具,可以使用户摆脱对复杂专家经验的依赖,提高渗透测试的可操作性和测试效率[1-2],因此实施自动化渗透测试具有重大现实意义。

自动化渗透测试主要包括规划和移动两个阶段。规划阶段通常需要扫描分析网络和主机配置信息,进行攻击路径发现,得到由主机和相应的漏洞利用动作组成的攻击路径。移动阶段通常在自动化脚本中指定攻击路径和漏洞利用信息,驱动渗透测试工具进行漏洞利用,然后进行本地提权等操作以获得主机控制权,以此为据点进行下一轮攻击,直至到达目标主机[3-4]。利用渗透测试工具对攻击目标进行高强度攻击时,容易出现目标主机服务崩溃等问题,此时原本可行的路径节点变得不可行。不同于机器人、无人机等领域的寻路问题,智能体对网络主机的控守耗时耗力,如果重新输入网络初始信息进行路径规划,意味着将放弃已经攻陷的路径主机,无论对规划还是执行阶段而言都是一种浪费。因此考虑这类动态环境下的攻击路径发现问题,尽可能减小重规划的问题空间,提高重规划成功率。

现有的攻击路径发现方法大多基于攻击图模型[5]或智能规划器[6-7],此方法在网络规模较小、环境信息不变的条件下可以发现高质量的攻击路径。但是当网络规模扩大时,上述

通信作者:周天阳(aipteamzhouty@aliyun.com)

方法往往容易出现无法求解等情况。蚁群算法在路径规划领域应用广泛，搜索能力突出，而且具有高并发性、易与其他算法结合等优势。经典的蚁群算法存在容易陷入局部回路、收敛至次优解、收敛速度慢等问题，很多学者针对具体应用场景对蚁群算法进行了研究，提出了很多改进方法，如定制启发式函数，通过设计本领域适用的启发式函数[8]或融合多种启发信息[9]，以提高寻路质量；动态调整信息素挥发系数[10]，防止信息素积累或衰减过快导致探索能力下降；设计信息素更新方式[11]，如全局更新、局部更新或将两者结合；带精英策略的蚁群算法[12]，通过对每次迭代的最优路径进行额外信息素更新，以提高寻优能力；最大最小蚂蚁系统[13]，只对每次迭代的最优路径进行信息素更新，将信息素浓度设置在一定范围内以避免搜索停滞。此外，很多研究考虑将蚁群算法和其他算法相融合，如蚁群算法与粒子群算法相结合[14]，将适应度水平的粒子保持在一定浓度，以保证种群多样性；将蚁群算法与遗传算法相结合[15]，充分发挥遗传算法出色的全局收敛率和蚁群算法的并行性；蚁群算法与人工势场法相结合[16]，以人工势场法得到的合力方向为蚁群算法的激励因素，使前期蚁群的运动更有方向性；蚁群算法与 A* 算法相结合[17]，通过引入 A* 算法的评价函数等改进蚁群算法的启发式信息素，增加全局路径平滑度。随着研究的深入，为了更好地模拟现实情况，蚁群算法也被广泛用于复杂动态环境下的路径发现，如文献[18]在规划过程中加入滚动窗口，可以对动态障碍物进行观察和预测，提高了蚁群算法对动态环境的适用性；文献[19]采用模糊控制的蚁群算法，可以在复杂地形避开障碍物并找到最短路径。

综合已有的研究成果可以看出，目前基于蚁群算法的路径发现研究大多基于二维或三维地图，运动方向便于定义，建模简单且直观。而在网络领域，主机及其上运行的服务组成的状态复杂，可选择的漏洞利用动作多种多样，需要将网络信息与攻防知识信息进行转化和表达，增加了算法的应用难度。针对经典规划技术的局限性和蚁群算法在网络攻击路径发现领域的适用性问题，本文提出了基于双向蚁群算法的网络攻击路径发现方法，根据网络和主机配置信息对攻击路径发现问题进行设计，在此基础上采用双向蚁群算法发现攻击路径，在路径主机节点发生变化时，采用重规划机制进行局部攻击路径的调整，更好地适应在动态环境下的路径发现。

2 蚁群算法介绍

受生物界蚂蚁觅食行为的启发，Colorni 等于 1992 年提出了蚁群算法[20]，后来该算法被广泛用于求解旅行商、机器人导航和最优化等问题。蚁群算法借助路径上的信息素这种物质，指导蚂蚁优先选择信息素浓度高的路径行走，使得最优路径上的信息素浓度越来越高，最终实现最优路径发现。

蚁群算法的主要步骤包括：信息初始化、蚂蚁寻路和信息素更新。信息初始化主要包括蚁群状态的初始化和参数的初始化设置。蚂蚁寻路是指每只蚂蚁根据一定的转移概率依次进行转移，直至到达目标点。其中，蚂蚁 k 在 t 时刻从路径点 i 转移至路径点 j 的概率 $P_{ij}{}^{k}(t)$，如式(1)所示。

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum\limits_{s\in allowed_k}[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}, & j\in allowed_k\\ 0, & \text{otherwise}\end{cases} \tag{1}$$

其中，$allowed_k$ 表示下一步可以选择的路径点，α 和 β 分别表示信息素和启发式函数因子重要程度的参数，$\tau_{ij}(t)$ 为 t 时刻路径 (i,j) 之间的信息素浓度，$\eta_{ij}(t)$ 为 t 时刻路径 (i,j) 的启发式信息，通常取

$$\eta_{i,j}(t)=1/d(i,j) \tag{2}$$

其中，$d(i,j)$ 是路径点 i 和 j 之间的距离。

蚂蚁在 $P_{ij}{}^{k}(t)$ 的指引下完成寻路后进行信息素浓度的更新，更新方式如式(3)和式(4)所示：

$$\tau_{i,j}(t+1)=\rho\tau_{i,j}(t)+\sum_{n=1}^{N}\Delta\tau_{i,j}^{n}(t) \tag{3}$$

$$\Delta\tau_{i,j}^{n}(t)=\begin{cases}Q/L_n, & \{i,j\}\subset path_n\\ 0, & \text{otherwise}\end{cases} \tag{4}$$

其中，$\tau_{i,j}$ 是路径点 i 和 j 之间的信息素浓度；$\Delta\tau_{i,j}^{n}$ 是第 n 只蚂蚁在路径点 i 和 j 之间留下的信息素；N 是蚁群中蚂蚁的数量；L_n 是第 n 只蚂蚁的路径长度；ρ 和 Q 是常数，分别表示信息素挥发系数和信息素常数。这种信息素更新方式为蚁周模型，除此之外，信息素的更新方式还有蚁量模型和蚁密模型。

3 基于双向蚁群算法的网络攻击路径发现方法

3.1 网络场景描述

网络攻击路径规划的目的是找到一条可以到达目标主机的攻击序列，借助主机间的漏洞利用关系进行转移直至到达目标。如图 1 所示，攻击的工作空间包含了网络和主机的发现，以及对应的操作系统、端口、服务、被攻陷的主机等。本文假设工作空间的这些信息可以借助探测扫描提前获知，对某台主机进行漏洞利用后即认为获得了该主机的控制权，暂时忽略建立 C2(Comand & Control)、提升本地权限等过程。

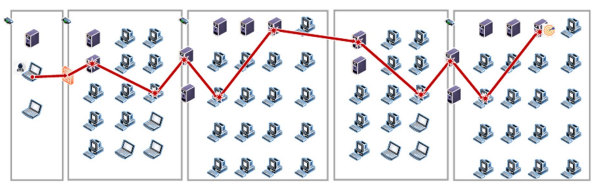

图 1 网络攻击路径示意图

Fig. 1 Schematic diagram of network attack path

网络信息可以用有向无环图 $G=\langle V,E\rangle$ 表示，其中，$V=\{v_0,v_1,v_2,\cdots,v_n\}$ 是网络中主机的集合，$E=\{e_{ij}\mid v_i,v_j\in V,i\neq j\}$ 表示主机之间的连接关系，这里所说的连接指直接相连，即从主机 v_i 访问主机 v_j 仅需一跳。机器人寻路问题中通常用欧几里得距离作为边的代价，但是在渗透测试环境下，主机之间通过漏洞利用关系形成攻击路径，路径代价需要结合漏洞信息进行设定。这里用 $e_{ij}=\langle acts,cost,minact\rangle$ 表示主机 i 可以通过漏洞利用动作集合 $acts$ 攻入主机 j，用 $cost$ 表示漏洞利用动作中攻击代价的最小值，即主机 i 攻入主机 j 的最小攻击代价，对应的漏洞利用动作是 $minact$。本文以攻击代价最小作为攻击路径发现的目标，攻击代价取决于主机的漏洞信息。通用漏洞评分系统(CVSS)是用来评测漏洞严重程度的指标，其分值越大说明漏洞越危险，被利用来攻击的可能性越大，攻击代价相对越小。本文将路径主机 i 和 j 之间的路径代价定义为式(5)。

$$cost_{ij}=\min\{100-10*CVSS(a_{jk})\} \tag{5}$$

其中，a_{jk} 表示主机 j 存在漏洞 k，可被利用；若主机 j 存在

多个漏洞$\langle k_1,k_2,\cdots,k_n\rangle$时，$cost_{ij}$为漏洞严重程度最高的漏洞对应的攻击代价，该漏洞对应的利用动作为 $minact$。

3.2 基于双向蚁群算法的网络攻击路径发现方法

经典蚁群算法中，种群所有的蚂蚁均从起点开始搜索，到终点结束。虽然采用了基于概率的选路策略，兼顾了一定的探索和利用平衡，但在算法开始时，蚂蚁因为缺乏引导，路径质量低，从而影响了信息素的分配，蚂蚁容易陷入局部最优解。

(1)双向蚂蚁搜索策略

为了使蚂蚁更好地探索未知区域，在加快收敛速度的同时尽量避免陷入局部最优解，本文采用双向蚁群算法进行路径发现。设置两个种群的蚂蚁，正向蚂蚁从起始主机开始搜索，搜索路径表示为 $L_1=\langle f_0,f_1,\cdots,f_s\rangle$；反向蚂蚁从目标主机进行反向搜索，为便于统一描述也将搜索路径表示为 $L_2=\langle b_0,b_1,\cdots,b_t\rangle$。其中，$f_i,i\in[0,s]$为正向路径的主机，$f_0$是起始主机，$f_s$是目标主机；$b_i,i\in[0,t]$为反向路径的主机，$b_0$是起始主机，$b_t$是目标主机。本文对正向蚁群和反向蚁群分别设置不同的路径转移策略，在求得蚂蚁下一步转移概率 $P_{ij}^{\ k}(t)$后，正向蚁群采用经典蚁群算法的寻路策略，通过轮盘赌的方式选择下一步要转移的路径主机，保证蚂蚁可以以一定概率探索未知区域；反向蚁群选择转移概率最大的候选路径主机，并将其作为下一步转移的路径主机，保证蚂蚁优先选择局部最优路径，提高了路径质量。正向搜索会因为分支过多导致搜索效率下降，而反向搜索从目标主机开始进行，可以在一定程度上减少访问的节点数量，且保证了和正向搜索路径具有一定差异。

(2)路径交叉优化操作

在正向蚂蚁 Ant_{m1} 和反向蚂蚁 Ant_{m2} 都搜索到一条路径后，本文借鉴遗传算法的交叉变异思想[21]，将两条不同寻路策略发现的路径进行交叉操作，得到交叉后的更优路径。假设两条路径 L_1 和 L_2 存在相同主机，即 $f_p=b_q$ 时，

$L_1=\langle f_0,f_1,\cdots,f_p,\cdots,f_s\rangle$

$L_2=\langle b_0,b_1,\cdots,b_q,\cdots,b_t\rangle$

根据交叉点将原路径进行切分，

$L_{1-1}=\langle f_0,f_1,\cdots,f_{p-1}\rangle,L_{1-2}=<f_{p+1},f_{p+2},\cdots,f_s\rangle$

$L_{2-1}=\langle b_0,b_1,\cdots,b_{q-1}\rangle,L_{2-2}=<b_{q+1},b_{q+2},\cdots,b_t\rangle$

如果

$$\sum_{i=0}^{p-2}cost(e_{i,i+1})\leqslant\sum_{i=0}^{q-2}\mathrm{cost}(e_{i,i+1}) \tag{6}$$

则 $L_{\mathrm{new1}}=L_{1-1}$，反之 $L_{\mathrm{new1}}=L_{2-1}$；同理，如果

$$\sum_{i=p+1}^{s-1}cost(e_{i,i+1})\leqslant\sum_{i=q+1}^{t-1}cost(e_{i,i+1}) \tag{7}$$

则 $L_{\mathrm{new2}}=L_{1-2}$，反之 $L_{\mathrm{new2}}=L_{2-2}$。更新后的路径 $L=L_{\mathrm{new1}}\cup f_p\cup L_{\mathrm{new2}}$。

如果两条路径没有相同主机，直接选择总代价最小的路径作为交叉后的路径。如果两条路径存在多个共同路径主机，为了使交叉优化后的效果更加明显，取中间位置的路径主机作为交叉点。首先获取两条路径的公共主机列表为$[h_0,h_1,\cdots,h_g]$，其中 $h_i\in L_1\cap L_2,i\in[0,g]$，$i$ 越小表明越靠近起始主机，越大表明越靠近目标主机。取交叉点 $h_{\lceil g/2\rceil}$，更新后的路径 L 作为蚂蚁第 m 次的搜索路径。

(3)信息素更新策略

在一次迭代过程中蚂蚁完成搜索后，需对信息素分布进行更新。如果基于所有蚂蚁进行更新，而反向蚁群可以很快收敛至较优解，算法探索能力下降，容易早熟；如果只基于正向蚂蚁进行更新，不能充分利用蚂蚁寻路信息，最优路径信息素浓度积累慢，最终运行结果同样较差。为解决这一矛盾，本文提出了新的信息素更新方案。首先根据正向蚁群的寻路结果进行信息素更新，然后借鉴精英蚁群策略，对本次迭代产生的最优路径进行额外的信息素增强。

$$\tau_{i,j}(t+1)=\rho\tau_{i,j}(t)+\sum_{n=1}^{N}\Delta\tau_{i,j}^{n}(t)+E\times\Delta\tau_{i,j}^{\mathrm{best}} \tag{8}$$

$$\Delta\tau_{i,j}^{n}(t)=\begin{cases}Q/L_n, & \{i,j\}\subset path_n\\ 0, & \text{otherwise}\end{cases} \tag{9}$$

$$\Delta\tau_{i,j}^{\mathrm{best}}=\begin{cases}Q/L_{best}, & \{i,j\}\subset path_{best}\\ 0, & \text{otherwise}\end{cases} \tag{10}$$

其中，t 表示迭代次数，$\tau_{i,j}$是路径点 i 和 j 之间的信息素浓度，$\Delta\tau_{i,j}^{n}$表示所有正向蚂蚁在路径点 i 和 j 之间留下的信息素，$\Delta\tau_{i,j}^{\mathrm{best}}$是最优路径中路径点 i 和 j 之间的信息素。N 是正向蚁群的数量，E 是精英路径调节参数，L 表示路径总代价，$path$ 表示路径主机集合。这种更新方案既避免了反向蚂蚁带来的部分路径信息素过快积累，又通过对最优路径的信息素额外更新提升了寻优能力，这里的最优路径取自交叉优化后的路径集合。最终通过实验验证了本文方案的有效性。

(4)重规划机制

关于重规划起点的选择，传统的重规划算法直接将离故障点最近的路径点作为起始点，规划出一条到目标点的路径。但是如果当前路径主机连接较少，在其邻接路径主机发生故障后，以此为新的起始主机进行规划可能会出现无路可走的情况，导致重规划失败。例如，假设当前智能体已攻陷主机 h_1，h_1与 h_2 和 h_3 属于同一局域网，但互相之间缺乏漏洞依赖关系。h_1可访问的主机是 h_4 和 h_5，h_1 与 h_4 具有漏洞依赖关系，与 h_5 没有漏洞依赖关系。h_2 和 h_3 以及 h_4 和 h_5 均有漏洞依赖关系。原始规划路线中 h_1 的下一步攻击目标为 h_4，当 h_4 出现故障后，如果以 h_1 作为起始主机进行重规划，会导致规划失败，原因在于 h_1 的下一跳可攻击主机太少；如果以 h_1 之前的某台路径主机作为重规划起点，最终可以通过 h_2 或者 h_3 攻击 h_5，实现攻击路径的重规划。机器人导航领域经常使用到目标点的距离作为启发值信息，进行重规划起点选择[22-23]，而在自动化渗透测试中，这样的“距离”信息难以估计。本文考虑将路径主机的出度作为重规划起点的选择依据，路径主机的出度越大，可以选择的下一跳主机越多，越容易跳出“局部死路”，避免重规划失败。

假设智能体已经沿着既定规划路线移动至主机 h_{ob-1}，在尝试攻击主机 h_{ob} 时导致 h_{ob} 崩溃，则已攻陷主机列表 $L_{\mathrm{control}}=[h_0,h_1,\cdots,h_{ob-1}]$，$h$ 表示初始规划路径主机。选择出度最大的路径主机作为重规划的起始主机，距离障碍主机最近的靠近目标的主机作为重规划的目标主机，重新调用双向蚁群算法，快速发现绕过障碍物的最佳路径，然后沿着原来的规划路线继续移动，直至到达目标主机。

(5)算法设计

图 2 给出基于双向蚁群算法的网络攻击路径发现方法的流程，其中具体的寻路算法如算法 1 所示。

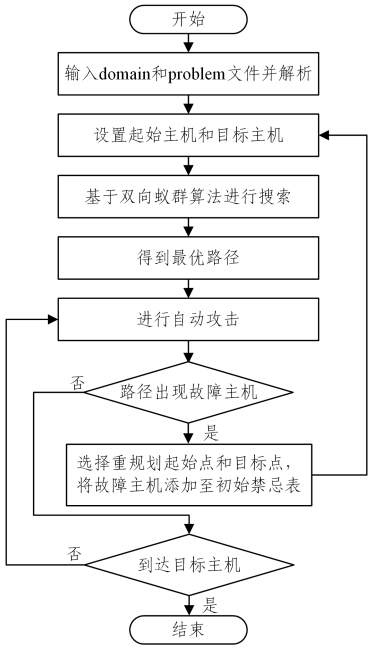

图 2 APD-BACO 算法流程

Fig. 2 APD-BACO algorithm flow

算法 1 双向蚁群算法

输入：domain. pddl，problem. pddl/ * 具体内容在实验章节介绍 * /

输出：攻击路径主机〈h_0，h_1，…，h_n〉，对应的漏洞利用动作及攻击代价{(h_i，h_{i+1})：[vul，cost]}

```
1. H,E,C←pddlParser(domain. pddl,problem. pddl)/ * 解析网络信息得到主机、边及边对应的攻击代价 * /
2. Initialize(pheromone_graph) / * 初始化边的信息素浓度 * /
3. Initialize(best_ant) / * 保存全局最优路径信息 * /
4. for i←1 to iterNum do / * iterNum 是迭代次数 * /
5.     for j←1 to antNum/2 do / * antNum 是蚂蚁种群数量，均分为正向蚁群和反向蚁群 * /
6.         Initialize(forward_ant,backward_ant) / * 初始化正向蚂蚁和反向蚂蚁 * /
7.         forward_ant. searchPath() / * 正向蚂蚁寻路 * /
8.         backward_ant. searchPath() / * 反向 * /
9.         Cross_Path() / * 进行路径交叉优化 * /
10.    localbest_ant←mincost(ANT) / * 保存本次迭代最优路径信息，用于信息素更新 * /
11.    Update_Pheromone() / * 信息素更新 * /
12.    IF localbest_ant<best_ant THEN
13.        best_ant←localbest_ant / * 记录全局最优路径信息 * /
14. END
FUNCTION searchPath()/ * 蚂蚁寻路 * /
1. next_host←start_host
2. While next_host not goal_host do
3.    pre_host←next_host
4.    IF pre_host. hasConect() THEN
5.        next_hosts←getNeighbor(pre_host)
6.        next_host←getNext(pre_host,next_hosts,id)
7.        IF next_host is NULL THEN
8.            break
9.            host_parent[next_host]←pre_host
10.   ELSE
11.       path. remove(next_host)
12.       open_table_host[next_host]←False
13.       next_host←host_parent[next_host]
14.       continue
15.   Move()
FUNCTION getNext(pre_host,next_hosts,id)
1. for each host∈next_hosts do
2.    IF open_table_host[host] THEN
3.        (pre_host,host). prob←getProb()
4. IF id is forward_ant THEN
5.    return next_host←Roulette()
6. ELIF id is backward_ant THEN
7.    return next_host←Maxprob()
```

localbest_ant 在每轮迭代中的初始值为∞，随着蚂蚁寻路的不断更新，其保持为最小的路径代价，主要用于发现精英路径并用于信息素更新。*best_ant* 是全局变量，初始值为∞，每结束一次迭代对其进行更新，保持为最小的路径代价，迭代停止时即为算法发现的最优路径代价。函数 searchPath() 中，*pre_host*. hasConect()用于判断当前主机是否与其他主机有连接，若无，则需要回溯重新选择路径；*open_table_host* 是禁忌表，保存已访问过的主机；*host_parent* 用于记录父子节点关系，回溯的时候会用到；Move()表示移动，将 *next_host* 加入路径 *path* 并添加禁忌表，当达到目标主机时，即得到该蚂蚁找到的路径及路径代价。函数 getNext()返回蚂蚁下一步要移动的主机，getProb()是根据式(1)计算得到的概率，获取所有候选主机的概率后，若为正向蚂蚁，返回轮盘赌选中的主机；若为反向蚂蚁，返回概率最大的主机。

4 实验

本实验所使用的漏洞信息均来自公开漏洞数据库及 Metasploit，网络信息及漏洞信息已使用规划领域定义语言 PDDL 进行形式化描述。PDDL 语言是连接规划问题和规划求解器的桥梁[24]，包含 domain 文件和 problem 文件。domain. pddl 是对模型的描述，比如动作、动作的前置条件以及后置条件等；problem. pddl 是对问题域的描述，包括初始状态和目标状态等。初始状态主要描述网络中包含的主机、主机间连接关系、主机的属性等。渗透测试环境的工作空间转换为 pddl 描述的初始状态，漏洞利用和攻击模块转换为动作，相应的 pddl 描述示例如表 1 和表 2 所列。

表 1 domain. pddl 示例

Table 1 Example of domain. pddl

```
(:action Dell_SonicWALL_Plixer_Scrutinizer_9_SQL_Injection
:parameters(?srcIP ?dstIP)
:precondition(and
(connectivity ?srcIP ?dstIP)
(compromised ?srcIP)
(iswindows ?dstIP)
(isscrutinizer8_6_5 ?dstIP)
:effect(and(compromised ? dstIP)(increase(total-cost) 45.0)))
```

表 2 problem. pddl 示例

Table 2 Example of problem. pddl

```
(isunix host_36)
(iszimbra_collaboration_suite6_0_4 host_36)
(islifesize_room_appliance_softwarels_rm1_3_5_9 host_36)
(iscakephp1_3_0 host_36)
(isgec_2fged1_4_3 host_36)
(isxml_rpc1_0_10 host_36)
```

首先应用改进后的蚁群算法进行网络攻击路径发现，以验证改进算法的有效性。将 APD-BACO 与经典蚁群算法(Ant Colony Algorithm，ACO)、精英蚁群算法(Elite Ant Colony Algorithm，EACO)进行比较，对最优路径代价、最优路径发现成功率、首次发现最优路径的迭代次数、平均路径代价、平均运行时间、运行时间范围这 6 个方面进行比较分析，为了不失一般性，本文分别在不同主机规模的网络下进行实验，其中网络 1 包含 50 台主机，网络 2 包含 100 台主机。实验使用 python 作为编程语言，实验平台包括 Intel i9，2.59 GHz 处理器和 128 GB 内存。结合相关文献研究和实验经验，选取的参数如表 3 所列。

表 3 算法参数设置

Table 3 Algorithm parameter settings

参数	信息量因子 α	启发式因子 β	信息素蒸发系数 ρ	信息素浓度增强因子 Q	精英路径调节参数 E
取值	1.0	2.0	0.5	100	0.5

因为算法具有一定的随机性，所以将 3 种算法各运行 100 次后进行统计分析。每次运行迭代次数均设置为 200 次，3 种算法的蚂蚁总数相同，其中网络 1 蚂蚁总数设置为 30，网络 2 蚂蚁总数设置为 50，其余参数保持一致。在网络 2 的实验过程中，将 APD-BACO 与 ACO，EACO 的最优路径代价随迭代次数的变化情况进行对比，结果如图 3 和图 4 所示。

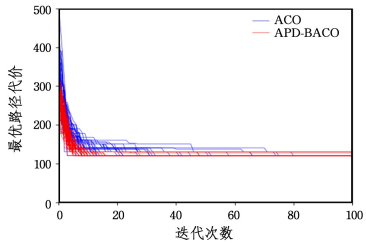

图 3 ACO 与 APD-BACO 最优路径代价随迭代次数的变化

Fig. 3 Change of the optimal path cost of ACO and APD-BACO with the number of iterations

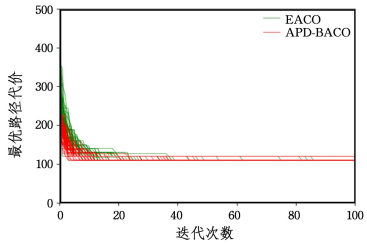

图 4 EACO 与 APD-BACO 最优路径代价随迭代次数的变化

Fig. 4 Change of the optimal path cost of EACO and APD-BACO with the number of iterations

表 4 ACO、EACO 和 APD-BACO 算法的运行结果统计

Table 4 Statistics of running results of ACO, EACO and APD-BACO algorithms

	网络 1			网络 2		
	ACO	EACO	APD-BACO	ACO	EACO	APD-BACO
最优路径代价	50	50	50	110	110	110
最优路径发现成功率/%	100	100	100	73	99	96
最优迭代次数	3.10	2.95	1.52	35.13	25.94	11.31
平均路径代价	50.22	50.30	50.04	118.82	124.13	113.52
平均运行时间/s	0.87	0.78	0.80	5.61	6.02	3.92
运行时间范围/s	[0.85,0.95]	[0.74,1.20]	[0.79,0.82]	[4.92,6.38]	[5.09,7.05]	[3.40,4.37]

由表 4 可以看出，在网络规模较大时，EACO 和 APD-BACO 的最优路径发现成功率远高于 ACO，有效避免了算法陷入局部最优。EACO 在 ACO 信息素更新的基础上，增加了对最优路径信息素的额外更新，因此增加了计算量，在网络规模扩大时耗时明显增加；APD-BACO 也有最优路径信息素的额外更新，但相较于其他两种算法，其只更新正向蚁群的路径信息素，更新次数是其他两种算法的一半，交叉优化操作虽然增加了计算量，但在迭代过程中路径质量得到了提升，加之反向蚂蚁寻路策略简单，算法最终的运行时间更短，且保证了最优路径发现的成功率。从图 3 和图 4 可以看出，相较于 ACO，EACO 在更短时间内迭代至最优解，APD-BACO 的最优迭代次数比其他两种算法更少，从初始时刻开始，蚂蚁的寻路质量已经得到了提高，而且保留了蚁群算法的高并发性，便于进行并行计算。

Metric-FF 规划器[25]是著名的 FF 规划器的最新版本，集成了多种路径规划算法，可以处理数值效果。D*(D_star)算法是一种高效的启发式搜索算法，是曾用于火星探测器核心的寻路算法。将 APD-BACO 与 Metric-FF 规划器和 D* 算法进行对比，分别在不同规模的网络中进行攻击路径发现。图 5 为随着网络规模的扩大，3 种算法的运行时间和内存占用情况对比。

图 5 中红色实线表示算法的运行时间，蓝色虚线表示算法的内存占用情况。可以看出，Metric-FF 算法所需的运行时间呈指数上升，当主机数量达到 400 时，Metric-FF 的运行时间为 6279.25 s；D* 算法的运行时间最少且最平稳，从开始时的 1.41 s 增加至 6.21 s。从内存占用情况看，当主机数量较少时 APD-BACO 和 D* 算法的内存占用情况基本相同，均大于 Metric-FF 算法；主机数量增加时，D* 算法的内存占用最多，Metric-FF 算法虽然内存占用较少但增长速度快。APD-BACO 的运行时间和内存占用介于其他两种算法之间，主要原因在于 D* 算法需要维护 OPEN 和 CLOSE 表，通过记录和更新所有节点的统计信息缩短运行时间。Metric-FF 算法的启发式评估通过 Graphplan 完成，状态空间增长过快。APD-BACO 是一种渐进收敛算法，寻路带有一定的随机性以兼顾探索和利用的平衡，因此时间和空间随主机数量的增加变化较为平缓。

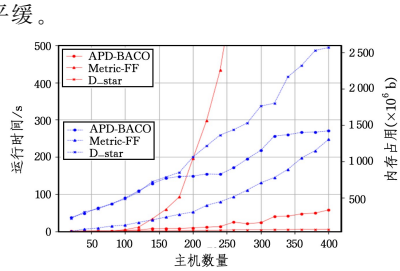

图 5 3 种算法运行时间和内存占用随主机数量的变化情况(电子版为彩图)

Fig. 5 Running time and memory usage of three algorithms vary with the number of hosts

设置如图 6 所示的实验网络，模拟层层设防、目标主机位于内网深处的攻击情形。防火墙左侧是攻击者所在网络，防火墙右侧是渗透测试场景，每个子网包含不同数目的主机。其中网络连接的规则设置如下：每个子网内的主机可以互相访问，且只能被前一子网的主机访问。例如，Host-15 只能访问子网 172.31.20.0/24 和 172.31.30.0/24 的主机，不能访问子网 172.31.10.0/24 的主机；Attacker 仅能访问子网 172.31.10.0/24 的主机。主机配置信息由漏洞利用动作所需要的前置条件随机生成。首先利用 APD-BACO 进行路径发现，得到的攻击路径序列如图 6 所示，红线表示攻击路径序列。

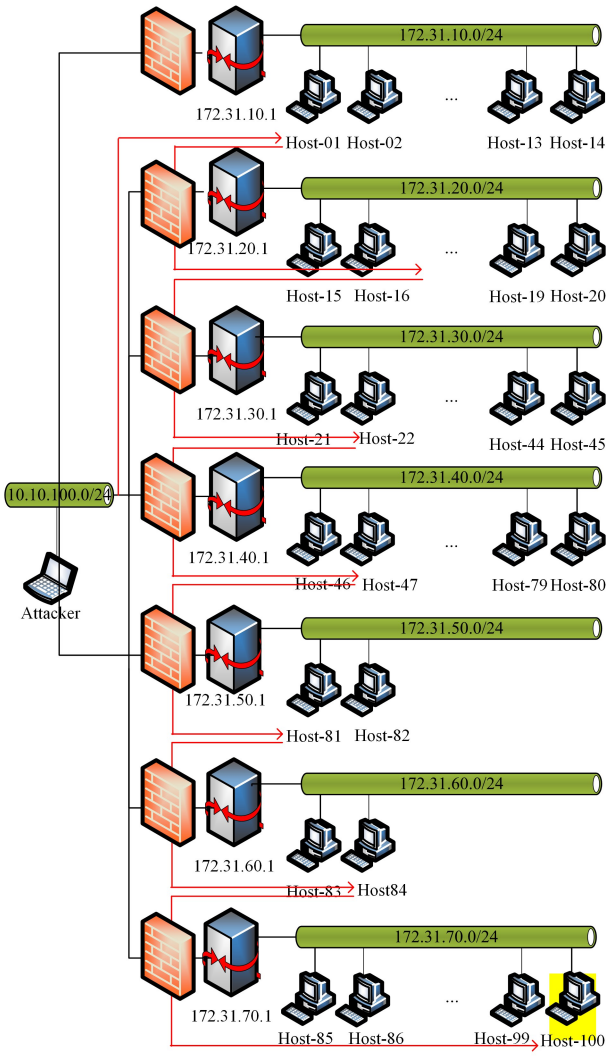

图 6 实验环境设置及攻击路径展示(电子版为彩图)

Fig. 6 Experimental environment setting and attack path display

假设当智能体以 Host-81 为跳板机攻击 Host-84 时，因过于频繁的访问导致 Host-84 主机崩溃，环境信息发生了变化，如果直接选择故障主机最近的 Host-81 主机作为新的起点，则需将 Host-84 添加至初始禁忌表进行重规划，当 Host-81 和 Host-83 的运行状态不满足漏洞利用动作的前置条件时，会导致重规划失败。为了避免重规划失败带来的多次规划，提高重规划成功率，本文选择出度最大的路径主机 Host-22 作为重规划的起点，将故障主机附近靠近目标的主机作为重规划的终点，此方法可以快速发现局部路径，避免重规划再次失败，最终得到的规划路线如图 7 所示。

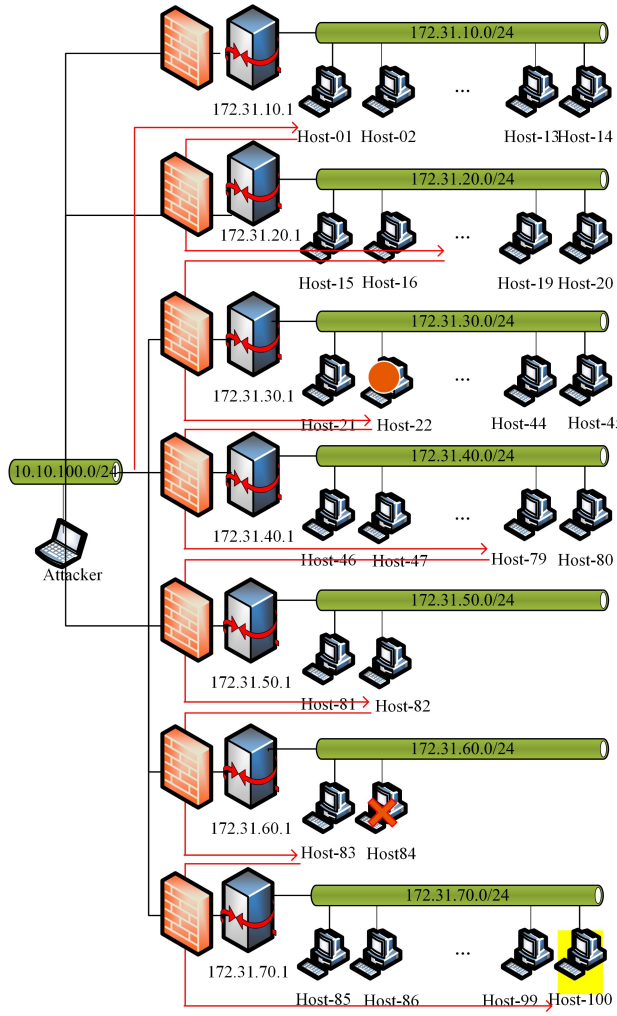

图 7 路径主机故障后的攻击路径展示

Fig. 7 Attack path display after path host failure

结束语 本文提出了一种适用于动态网络环境的攻击路径发现方法。首先对网络信息进行建模，定义了攻击代价；然后提出了一种新的双向蚁群算法进行全局攻击路径发现，当路径主机发生故障导致环境变化时，选择较优的起始主机和目标主机进行局部重规划，提高重规划成功率。

通过与 ACO 和 EACO 算法进行实验对比，发现 APD-BACO 具有收敛速度更快、搜索过程中的平均路径代价更低、运行时间更短等优势。与 Metric-FF 规划算法和 D* 算法进行实验对比，结果表明随主机数量增加，APD-BACO 的时间和空间代价增长较为平稳，适应性强。本文的重规划策略可提高局部重规划成功率，指导智能体在动态环境中进行攻击移动，因此 APD-BACO 更加适合动态网络环境下的攻击路径发现。

参 考 文 献

[1] STEFINKO Y, PISKOZUB A, BANAKH R. Manual and automated penetration testing. Benefits and drawbacks. Modern tendency[C]// 2016 13th International Conference on Modern Problems of Radio Engineering. Telecommunications and Computer Science(TCSET). IEEE, 2016: 488-491.

[2] ALMUBAIRIK N A, WILLS G. Automated penetration testing based on a threat model[C]// Internet Technology & Secured

Transactions. IEEE, 2017: 413-414.
[3] LUAN J, JIAN W, XUE M. Automated Vulnerability Modeling and Verification for Penetration Testing Using Petri Nets[C]// International Conference on Cloud Computing & Security. Cham: Springer, 2016: 71-82.
[4] KUCAN B. Automated Penetration Testing with CORE IMPACT 4. 0[J/OL]. https://www. helpnetsecurity. com/2004/05/24/automated-penetration-testing-with-core-impact-40/.
[5] OU X, BOYER W F, MCQUEEN M A. A scalable approach to attack graph generation[C]// Proceedings of the 13th ACM Conference on Computer and Communications Security. 2006: 336-345.
[6] CASTILLO L A, FERNÁNDEZ-OLIVARES J, GARCIA-PEREZ O, et al. Efficiently Handling Temporal Knowledge in an HTN Planner[C]// ICAPS. 2006: 63-72.
[7] RICHTER S, WESTPHAL M. The LAMA planner: Guiding cost-based anytime planning with landmarks[J]. Journal of Artificial Intelligence Research, 2010, 39: 127-177.
[8] SUN Y, DONG W, CHEN Y. An improved routing algorithm based on ant colony optimization in wireless sensor networks [J]. IEEE Communications Letters, 2017, 21(6): 1317-1320.
[9] ZHENG B L, LI Y H. Study on SDN Network Load Balancing Based on IACO[J]. Computer Science, 2019, 46(6A): 291-294.
[10] CAO J. Robot Global Path Planning Based on an Improved Ant Colony Algorithm[J]. Journal of Computer & Communications, 2016, 4(2): 11-19.
[11] DENG W, XU J, ZHAO H. An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem [J]. IEEE Access, 2019, 7: 20281-20292.
[12] HU X, LUO P, ZHANG X, et al. Improved ant colony optimization for weapon-target assignment[J]. Mathematical Problems in Engineering, 2018: 1-14.
[13] YUE L, CHEN H. Unmanned vehicle path planning using a novel ant colony algorithm[J]. EURASIP Journal on Wireless Communications and Networking, 2019, 2019(1): 1-9.
[14] XUAN C, DAN L. Task scheduling of cloud computing using integrated particle swarm algorithm and ant colony algorithm[J]. Cluster Computing, 2017(4): 1-9.
[15] LUAN J, YAO Z, ZHAO F, et al. A novel method to solve supplier selection problem: Hybrid algorithm of genetic algorithm and ant colony optimization[J]. Mathematics and Computers in Simulation, 2019, 156: 294-309.
[16] CHEN G, LIU J. Mobile Robot Path Planning Using Ant Colony Algorithm and Improved Potential Field Method[J]. Computational Intelligence and Neuroscience, 2019, 2019(b): 1-10.
[17] DAI X, LONG S, ZHANG Z, et al. Mobile Robot Path Planning Based on Ant Colony Algorithm With A * Heuristic Method [J]. Frontiers in Neurorobotics, 2019, 13: 15-23.
[18] NECULA R, BREABAN M, RASCHIP M. Tackling Dynamic Vehicle Routing Problem with Time Windows by means of Ant Colony System[C]// Evolutionary Computation. IEEE, 2017: 2480-2487.
[19] YEN C T, CHENG M F. A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance[J]. Microsystem Technologies, 2018, 24(1): 125-135.
[20] COLORNI A, DORIGO M, MANIEZZO V. An Investigation of some Properties of an Ant Algorithm[C]// Parallel Problem Solving from Nature 2, PPSN-II. Brussels, Belgium, Elsevier, 1992: 28-30.
[21] WHITLEY D. A genetic algorithm tutorial [J]. Statistics and Computing, 1994, 4(2): 65-85.
[22] SUI L L, CHEN X. Improved Algorithm of Robot Online Path Planning[J]. Chinese Journal of Computer Engineering and Applications, 2010, 46(28): 36-39.
[23] CHEN R N, WEN C C, PENG L, et al. Application of Improved A* Algorithm in Robot Indoor Path Planning[J]. Chinese Journal of Computer Applications, 2019, 39(4): 1006-1011.
[24] OBES J L, SARRAUTE C, RICHARTE G. Attack Planning in the Real World[J]. arXiv: 1306. 4044, 2013.
[25] HOFFMANN J. The Metric-FF Planning System: Translating Ignoring Delete Lists to Numeric State Variables[J]. Journal of Artificial Intelligence Research, 2011, 20: 291-341.

GAO Wen-long, born in 1997, postgraduate. His main research interests include cyber security and intelligent planning.

ZHOU Tian-yang, born in 1979, associate professor. His main research interests include software vulnerability analysis, virtualization-based security technology and application and penetration test.