动态自组织 P2P 僵尸网络的构建及其防御

赵 昊[1], 舒 辉[1], 刘潮歌[2], 邢 颖[1], 赵耘田[1]

[1] 数学工程与先进计算国家重点实验室 郑州 中国 450001
[2] 中国科学院信息工程研究所 北京 中国 100093

摘要 僵尸网络作为大规模攻击活动的基础平台, 严重威胁网络空间安全, 从预测的角度对其开展研究具有重要的现实意义。针对现有研究在终端感知、身份识别和动态对抗中存在的不足, 本文概括僵尸网络生命周期, 总结 P2P 结构僵尸网络的脆弱点, 建立 P2P 僵尸网络动态对抗模型, 分析节点真实性判断和网络拓扑优化重构的重要性。在此基础上, 从攻击者视角提出一种新颖的动态自组织 P2P 僵尸网络模型 DSBot。该模型在架构设计上可扩展至各类目标设备, 通过基于可信度矩阵和真实性验证的节点安全性评估机制增强终端对抗性, 并提出分阶段感染策略。借鉴无线自组网和多智能体的思路和方法, 刻画节点属性多维表示和基于状态标识的动态网络框架, 以此为基础设计 $O(N_i)$更新算法、均匀连接算法和节点主动移除算法, 并结合相应的初始化和调整机制提出网络自组织重构策略, 从而进一步提升网络的健壮性。其中, $O(N_i)$更新算法确保节点的可信度, 均匀连接算法降低网络暴露风险, 节点主动移除算法实时移除可疑节点。从平均等待时间、命令可达率、网络连接度和重构稳定时间等方面对 DSBot 模型进行评估。实验结果表明, DSBot 模型在效率和韧性上可满足僵尸网络命令控制机制的基本需求。最后, 从终端清除、命令控制服务器打击和命令控制过程等方面讨论了可能的防御策略。本文旨在通过预测新型僵尸网络模型来完善防御解决方案。

关键词 网络安全; P2P 僵尸网络; 动态自组织; 健壮性
中图法分类号 TP309.5 DOI 号 10.19363/J.cnki.cn10-1380/tn.2023.03.03

Research on Dynamic Self-organizing P2P Botnet

ZHAO Hao[1], SHU Hui[1], LIU Chaoge[2], XING Ying[1], ZHAO Yuntian[1]

[1] State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou 450001, China
[2] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

Abstract As the basic platform for large-scale attacks, botnets seriously threaten the security of cyberspace. It is of great practical significance to study botnet from the perspective of prediction. Aiming at the shortcomings of existing research in terminal perception, identity recognition and dynamic confrontation, this article outline the botnet life cycle, summarizes the vulnerabilities of P2P botnets, establishes a P2P botnet dynamic confrontation model, and analyzes the importance of node authenticity judgment and network topology optimization. On this basis, this paper proposes a novel dynamic self-organizing P2P botnet model DSBot from the attacker's perspective. The model can be extended to all kinds of target devices in architecture design, enhance terminal antagonism through node security evaluation mechanism based on reliability matrix and authenticity verification, and propose phased infection strategy. Based on the ideas and methods of wireless AD hoc network and multi-agent, the multi-dimensional representation of node attributes and the dynamic network framework based on state identification are describe. Then the $O(N_i)$ update algorithm, uniform connection algorithm and active node removal algorithm are designed, and the self-organizing network reconstruction strategy is proposed combining the corresponding initialization and adjustment mechanism to further improve the robustness of the network. Among them, the $O(N_i)$ update algorithm ensures the credibility of the node, the uniform connection algorithm reduces the risk of network exposure, and the node active removal algorithm removes suspicious nodes in real time. The DSBot model is evaluated from the aspects of average waiting time, command reachable rate, network connectivity and reconstruction stability time. Experimental results show that DSBot model can meet the basic requirements of botnet command control mechanism in terms of efficiency and resiliency. Finally, possible defense strategies are discussed in terms of terminal clearance, command control server strikes and command control process. This paper aims to improve defense solutions by predicting new botnet models.

Key words cyber security; P2P botnet; dynamic self-organizing; robustness

通讯作者: 舒辉, 博士, 教授, Email:shuhui123@126.com
本课题得到国家重点研发计划前沿科技创新专项基金(No. 2019QY1305)资助。
收稿日期: 2021-11-16; 修改日期: 2021-12-13; 定稿日期: 2023-01-04

1 引言

网络空间安全事件频发, 诸如敏感信息窃取、网络钓鱼欺诈、加密勒索威胁、垃圾邮件轰炸、恶意软件分发和分布式拒绝服务等攻击活动层出不穷, 引起学术界和工业界的广泛关注。僵尸网络(Botnet)作为发动上述大规模攻击行动的基础设施平台, 被视为当今网络空间的最大安全威胁之一[1]。僵尸网络是指通过入侵网络空间内若干非合作用户终端构建的、可被攻击者远程控制的通用计算平台[2], 通常由控制者(Botmaster)、命令与控制信道(C&C Channel, Command and Control Channel)以及僵尸主机(Bot)共同组成。僵尸控制者能够通过 C&C 信道实现对僵尸节点的一对多远程管理, 是僵尸网络区别于病毒、蠕虫和木马等传统恶意代码的最大特征。实际上, C&C 机制是僵尸网络的核心所在, 且唯一决定一种僵尸网络。随着网络技术的进步和网络环境的变化, 僵尸网络经历了不同的发展阶段, 从早期的单一攻击到如今的广泛攻击, 其影响范围从 PC、服务器扩展到了智能移动终端和物联网设备, 构建技术越来越复杂, 表现形态日趋多样化, 攻击方式不断多元化, 破坏能力逐渐上升, 严重影响网络安全, 给防御者带来巨大挑战。因此, 深入研究僵尸网络机理, 关注僵尸网络演化规律, 预测新型僵尸网络形态和攻击技术, 完善已有僵尸网络防御体系, 对于提高僵尸网络安全事件的应急处置能力、保障我国网络空间安全具有十分重要的意义。

本文的主要工作如下: (1)基于现有研究, 概括僵尸网络生命周期, 分析僵尸网络的脆弱性, 并提炼 P2P 僵尸网络动态对抗模型, 为全文建立理论基础; (2)说明本文提出的动态自组织 P2P 僵尸网络模型(Dynamic Self-organizing P2P Botnet, 下文简称 DSBot)的基本组成, 阐述节点安全性评估方法和分阶段动态感染策略; (3)基于无线自组网和多智能体的概念和方法, 从节点属性多维表示、基于状态标识的动态网络框架和网络自组织重构策略三方面具体说明 DSBot 模型的网络自组织重构机制的实现细节; (4)从平均等待时间、命令可达率、网络连接度和重构稳定时间等方面对 DSBot 模型的效率和韧性进行评估, 实验结果表明该模型能够满足僵尸网络命令控制的基本需求; (5)分别从终端清除、命令控制服务器打击和命令控制过程对抗三方面对 DSBot 僵尸网络模型可能的防御策略作简要分析。

2 相关工作

僵尸网络预测是国内外相关领域的主要研究方向之一[3], 目的是从攻击者角度提出未来可能出现的新型僵尸网络类型, 并提出前瞻性的防御方法。Wang 等人[4]提出了一种混合型 P2P 僵尸网络模型, 通过引入 sensor 节点监控 bot 状态, 并根据收集到的节点信息随机更新 peer-list, 从而增强网络的健壮性, 但该方案并没有解决节点的可信度问题, 难以规避污染攻击。Hund 等人[5]提出 P2P 僵尸网络模型 RamBot, 引入信誉评分以及工作量证明机制判断节点身份的有效性, 理论上可以对抗常见的 P2P 僵尸网络对抗手段。然而 White 等人[6]研究发现, 虽然工作量证明机制可以帮助僵尸网络控制者抵御 Sybil 攻击等防御方的对抗手段, 但是信誉评分机制反而增加了僵尸网络的暴露风险。Marupally 等人[7]从拓扑结构入手, 提出了 Multi-Server P2P 的僵尸网络模型, 能够规避单一中心的脆弱性, 且可以优化地理分布, 缺点是在构建阶段需要额外的规划和部署, 成本较高。Arora 等人[8]基于 Kademlia 协议构建 P2P 僵尸网络模型, 研究僵尸网络感染规模与网络整体隐蔽性的关系, 结果表明 P2P 结构的僵尸网络是以增加暴露风险为代价来提升网络的健壮性。Anagnostopoulos 等人[9]将洋葱路由协议(Tor)用于僵尸网络 C&C 信道, 提出 Tor fluxing 方案, 隐藏了僵尸节点身份并且增加了网络的健壮性, 但只做了简单的概念验证, 并未作进一步评估和测试。Yin 等人[10]提出 PrBot 僵尸网络模型, 首次将信誉评估机制扩展到网络构建和 peer-list 更新的全过程, 兼顾了僵尸网络的健壮性和抗污染能力, 但未对信誉评估方法作详细阐述, 且只从建模角度进行仿真评估, 缺少 bot 程序的模拟运行过程。针对防御方通过在网络中布置传感器(sensor)节点从而感知僵尸网络活动并发现僵尸节点的情况, Bock 等人[11]提出 TrustBotMC 模型, 使得 Bot 节点能够在本地自主识别和屏蔽防御方布置的 sensor 节点, 但这种方法在防御方传感器数量增加时会逐渐失效。

总体而言, 已知的僵尸网络模型预测大多聚焦在僵尸网络的结构设计和协议实现等方面, 对僵尸程序在终端环境中感知对抗以及节点身份识别等问题的研究不够深入, 且较少涉及僵尸网络在动态对抗过程中如何保持弹性的问题。不同于上述工作, 本文提出动态自组织 P2P 僵尸网络模型 DSBot, 设计的架构可扩展至各类目标对象, 通过基于可信度矩阵和真实性验证的节点安全性评估机制增强终端对抗性, 并基于节点独立自主决策能力的网络自组织重构策略进一步提升网络的健壮性。

3 P2P 僵尸网络动态对抗模型

不同于木马、病毒等攻击形式, 僵尸网络作为发

动大规模网络恶意活动的基础设施平台, 形态和架构更为复杂, 其生命周期[12]涵盖多个阶段。本文从对抗的角度出发, 将僵尸网络生命周期概括为资源拓展、节点交互、组织维护和命令执行等状态。表 1 列出了僵尸网络生命周期各阶段状态的具体含义。在僵尸网络实际运行中, 各个状态之间并不相互独立, 例如资源拓展获取更多节点后需要组织维护过程进行管理, 而组织维护和命令执行均通过节点交互来完成。因此, 僵尸网络处于动态变化中, 而节点交互过程可以说是僵尸网络对抗的关键。

表 1　僵尸网络生命周期

Table 1　Life-cycle of botnet

状态	含义
资源拓展	通过漏洞利用、端口扫描、社会工程等感染方法, 获取节点资源, 扩展网络规模
节点交互	设备被感染后, 通过节点间的交互通信加入网络, 等待获取命令, 反馈必要信息
组织维护	网络处于动态变化中, 新增或失去节点需要维护, 调度和使用节点需要组织
命令执行	根据下发的命令类型, 各节点有组织地发起大规模协同式攻击活动

中心结构僵尸网络的节点交互过程实际上就是中央 C&C 服务器与僵尸节点之间的通信, 存在单点失效(single point of failure)[13]的固有缺陷, 一旦 C&C 服务器被打击、渗透或劫持, 整个僵尸网络也随之瘫痪失效。P2P 结构的僵尸网络具有分布式的特性, 健壮性相比于中心结构更强。然而随着研究的深入和对抗的升级, P2P 僵尸网络的脆弱性也被逐步揭露。针对 P2P 僵尸网络, 常用的对抗手段如图 1 所示。结构化 P2P 僵尸网络借助已知的 P2P 协议构建信道, 一般基于分布式哈希表(Distributed Hash Table)实现 P2P 通信, 使用索引结构<key, value>定位查找资源, 因此可采用索引投毒(Index Poisoning)[14]的方式提前写入破坏性命令从而干扰僵尸网络的正常运行。Sybil 攻击[15]则是通过将可控的良性节点伪装成僵尸节点加入到网络中, 从而实现对僵尸网络的渗透和控制。非结构化 P2P 僵尸网络通常使用自定义 P2P 协议, 每个节点维护并同步一个记录其他节点信息的 Peer list, 这类僵尸网络易受 Peer list 污染[10]使得无效的节点地址占据列表空间, 从而割裂僵尸网络的正常通信。随机地址扫描是另一种纯 P2P 网络构建方式, 但是这种 P2P 僵尸网络节点间连接度低, 通信效率欠佳, 产生的模式化流量很容易引起察觉, 并导致僵尸网络的暴露。

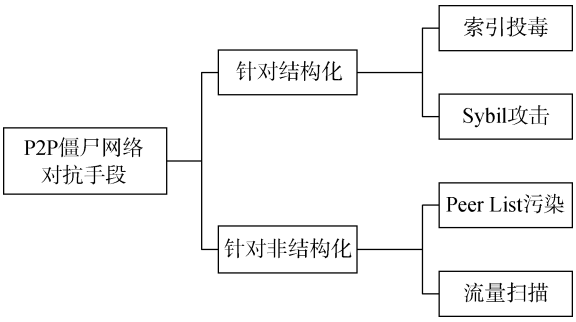

图 1　P2P 僵尸网络对抗手段

Figure 1　Countermeasures of P2P botnet

不难发现节点交互过程中认证机制的缺失是 P2P 僵尸网络的脆弱点之一, 无论是命令的伪造还是节点的伪装, 其本质都是利用了节点身份真实性评估不完善的缺陷。特别是近年来, 僵尸网络的感染范围进一步扩散至物联网层面, 且主要利用口令暴破和漏洞利用等手段入侵目标设备, 因此很容易落入防御方布置的“陷阱”。安全研究人员通过逆向工程等方法对捕获的样本进行分析, 从而对僵尸网络进行反制。针对这种情况, Mirai[16]等典型的物联网僵尸网络加入了反调试、反分析等手段来增强 bot 程序的终端对抗性, 但这种主要依赖初始化状态的一次性硬编码检查策略十分脆弱, 很容易就被防御方绕过。因此, 对于僵尸网络控制者而言, 建立节点真实性识别机制是提升网络健壮性和安全性的一种重要方法。节点身份识别包括两个层面: 一是对新加入节点的身份判别; 二是对网络运行过程中节点状态的实时监控。一旦身份被识别为不可信或可疑, 该节点在僵尸网络中的地位就会下降, 甚至被主动移除。除此之外, 受到网络条件变化、昼夜周期规律和系统运行故障等因素影响, 节点存在被动下线或暂时离线等状态。上述情况都会导致节点的数量和分布的动态变化, 从而引起僵尸网络拓扑的改变。虽然 P2P 僵尸网络对于个别节点的丢失具有天然的抗性, 但如果出现节点数量减少到一定程度或者是重要节点损失等情况, 会进一步影响 P2P 结构本就相对较低的通信效率, 从而导致僵尸网络攻击威胁的下降。若简单采取加强剩余节点连接的策略, 则会产生由连接度过高而引起的暴露风险。针对这类情况, 僵尸网络控制者需要通过组织维护过程, 动态管控网络拓扑, 优化节点间通联关系, 维持网络的通信效率。

综合以上分析, 本文提出 P2P 僵尸网络的动态对抗模型, 如图 2 所示, 子图Ⅰ-Ⅳ分别描绘了正反双方对抗过程中的不同状态, 表 2 给出图中各个符号的定义。

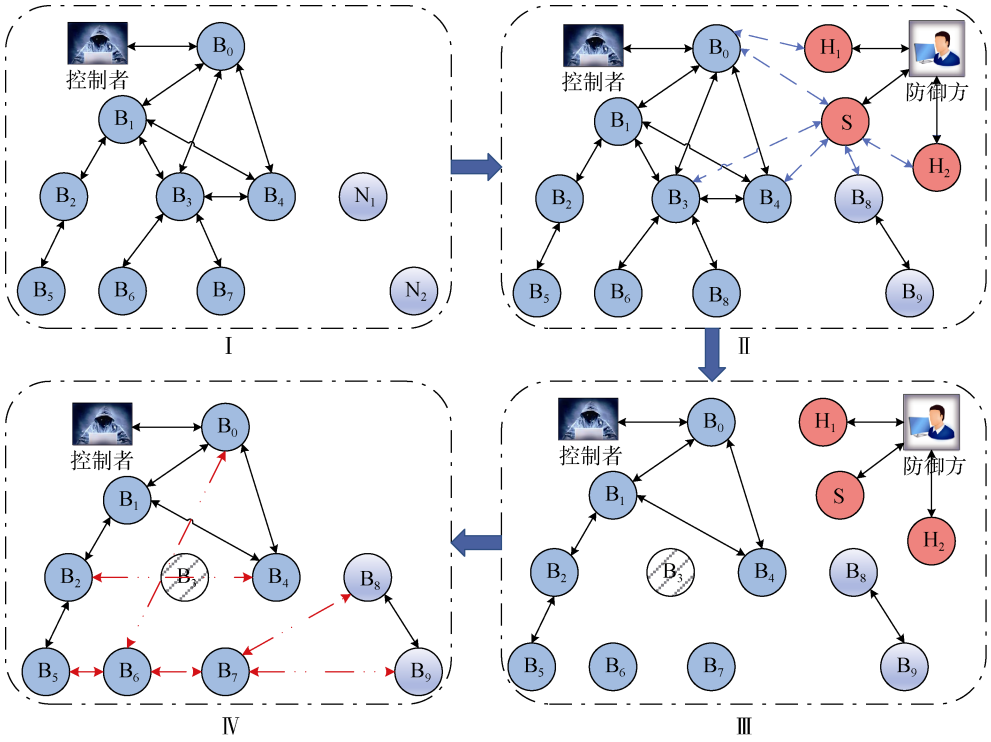

图 2　P2P 僵尸网络动态对抗模型

Figure 2　Dynamic confrontation model of P2P botnet

表 2　符号说明

Table 2　Description of symbol

符号	定义
控制者	僵尸网络的实际控制者
B_i	被感染后运行 Bot 程序的僵尸节点
N_i	可被僵尸网络感染的普通节点
防御方	对抗僵尸网络的组织或个人
H_i	防御方部署的蜜罐节点
S	防御方部署的 Sybil 节点

下面依次说明图 2 中各个子图所代表的状态:

状态Ⅰ: 僵尸网络在控制者操控下正常运行, 节点 B_0~B_7 之间以一定规则相互连接进行节点交互, 其中节点 B_3 的相邻节点数量较多, 在某些僵尸网络中将这类节点称为超级节点。N_1 和 N_2 是互联网中尚未被感染的脆弱节点。

状态Ⅱ: 防御方开始部署节点加入僵尸网络。H_1 和 H_2 是蜜罐(Honeypot)节点, 用以捕获僵尸程序样本, 逆向分析僵尸程序, 伪造通信消息, 并作为活跃节点加入僵尸网络。S 则是 Sybil 节点, 在防御方控制下运行良性僵尸程序, 通信及交互行为模仿正常的僵尸节点, 但不会造成对网络的真实危害。状态Ⅰ中的 N_1 和 N_2 节点此时已被感染成为正常的僵尸网络节点 B_8 和 B_9。

状态Ⅲ: 1)从控制者角度, 通过相应的节点身份识别算法, 在防御方发起反制前, 主动断开与 H 和 S 等处于防御方控制下的节点的连接, 而节点 B_3 则是由于断电或断网等客观原因处于离线状态; 2)从防御方角度, 发动反制措施, 使处于其直接控制下的H和S节点主动停止与僵尸网络的连接, 同时通过伪造命令的方式致瘫关键节点 B_3。

状态Ⅳ: 上一状态最终导致僵尸网络的拓扑发生较大变化, 节点间的连接状态发生改变, 为了重构网络提升通信效率, 采用相应的网络重构算法, 以节点自组织的方式智能化重构网络拓扑, 从而优化剩余节点间的通联关系并规避单一节点出现连接度过高的情况。

从 P2P 僵尸网络动态对抗模型可以看出, 攻防双方处于不断的博弈中, 对抗过程是动态且持续的。对于僵尸网络控制者而言, 提升网络对抗性的关键在于节点真实性判断和网络拓扑优化重构, 前者是“预防”机制, 而后者属于“修复”过程。这类高对抗性的僵尸网络是未来网络空间安全的潜在威胁。为了增加对此类僵尸网络的理解, 探讨有效的防御手段, 防范可能到来的安全威胁, 本文提出一种高对抗性的动态自组织 P2P 僵尸网络模型 DSBot。

4　DSBot 基础架构

4.1　基本组成

与典型的僵尸网络类似, DSBot 的组成结构可表

示为:

$$DSBot=\{Botmaster,C\&C,bot\} \quad (1)$$

定义 1 botmaster: DSBot 僵尸网络的控制者, 向僵尸网络下达攻击命令或操作指令, 通过跳板网络、匿名网络或其他手段隐蔽自身。

定义 2 C&C: Command and control, 命令控制机制, 即僵尸网络节点之间的交互方式和通信协议, 一定程度上也与僵尸网络拓扑结构相互匹配。

定义 3 bot: 运行在受僵尸网络感染的设备上的受控端程序, 等待获取命令并解析执行, DSBot 的目标终端覆盖全平台, 主要包括 PC 机、服务器以及家用路由器、IP 摄像头和智能家居等物联网设备。

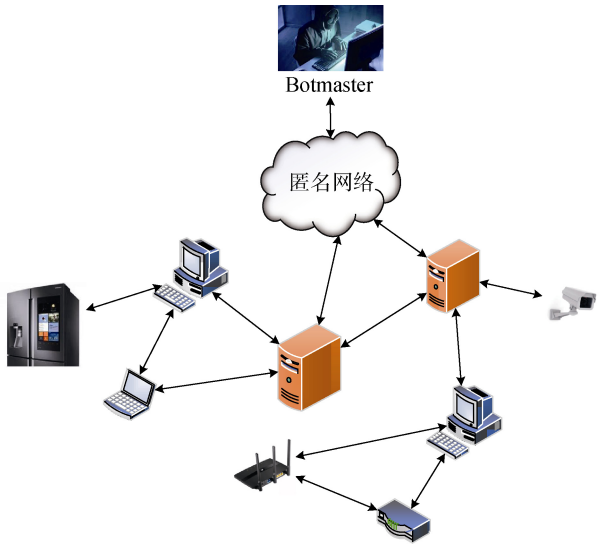

图 3 DSBot 基本组成

Figure 3 Basic composition of DSBot

DSBot 的基本架构如图 3 所示, 虽然不存在传统意义上的 C&C 服务器, 但节点之间的重要程度仍有所区别。由于设备类型和网络位置的不同, 节点的计算力和连通性有所差异。在 DSBot 僵尸网络中, 服务器、PC 等高性能且公网可达节点被视为主干节点, 而物联网设备则可看作是边缘节点。从这一层面来讲, DSBot 僵尸网络可归属于混合 P2P 结构的范畴, 即整体 P2P、局部中心化。

4.2 节点安全性评估

为了满足多类型节点共存的需要, 同时也为了更好发挥各节点自组织的特性, DSBot 建立了节点安全性评估机制。安全评估由两部分组成: 一是判断节点的设备性能和类型; 二是鉴别节点身份真实性。前者主要考虑节点性能是否允许其作为主干节点存在, 后者则是针对蜜罐和 Sybil 节点的对抗性措施, 起到规避样本捕获和抗渗透攻击的作用。DSBot 中的节点身份评估由被动探测和主动识别相结合的方式进行。

4.2.1 被动探测

被动探测的作用是确定设备性能和类型并识别蜜罐等不可信节点。判断设备类型较为简单, 只需增加探测模块, 获取系统类型 E_1、内核版本 E_2、网络带宽 E_3、CPU 型号 E_4、内存占用 E_5 和磁盘容量 E_6 等软硬件信息, 分别对每项指标赋予量化评估比重系数 e_i, 通过计算后给出设备性能 $E(t)$的一个综合判定值 $E(t)$, 由(2)式给出。

$$E(t)=e_1E_1+e_2E_2\cdots+e_nE_n \quad (2)$$

对蜜罐等不可信节点的探测则较为复杂, 这些节点通常由虚拟机、容器等虚拟化技术部署, 对其进行识别需要做综合性的判断, 从默认配置 A_1、系统指令 A_2 和异常行为 A_3 三方面对节点进行评估, 构建节点可信度评估矩阵 M, 如表 3 所示。显然, 覆盖的检测指标 a 越多, 非真实节点的可能性就越大, 然而如网络时延、端口开放数量、调用延迟等指标会存在误差, 正常系统在某些情况下也可能表现出与蜜罐类似的特征。为了提升检测的准确性, 本文引入加权参数 w_{ij} 以反映检测指标 a_{ij} 的重要程度。

表 3 可信度评估矩阵 *M*

Table 3 Credibility Evaluation Matrix *M*

类别 A	指标 a	权重 w
A_1(默认配置)	a_{11}: 检查 ping 程序的返回结果(蜜罐中往往返回相同结果)	w_{11}
	a_{12}: 检查常见用户名口令是否能够登陆	w_{12}
	a_{13}: 检查主机名是否正常	w_{13}
	a_{14}: 检查 Telnet、OpenSSH、Apache 等软件版本	w_{14}
	……	…
A_2(系统指令)	a_{21}: 检查常见 Linux 命令行操作(如“cd ~/, $?”)的执行结果	w_{21}
	a_{22}: 检查循环结构指令的执行结果(蜜罐中会报错)	w_{22}
	a_{23}: 检查带选项(如 -help)的命令行操作结果	w_{23}
	……	…
A_3(异常行为)	a_{31}: 检查网络延迟	w_{31}
	a_{32}: 检查系统组件调用的延迟	w_{32}
	a_{33}: 检查端口开放数量(正常系统一般不超过 5 个)	w_{33}
	a_{34}: 检查重新连接前后创建的文件是否仍然存在	w_{34}
	a_{35}: 检查监控软件标志	w_{35}
	……	…

$$P(t)=\sum_i A_iW_i=\sum_{i,j}a_{ij}w_{ij} \quad (3)$$

$$\sum_{i,j} w_{ij} = 1 \tag{4}$$

由此得出, 针对任一待检测目标 t, 可信度 $P(t)$ 的计算方法如式(3)所示, 且权重系数满足(4)式。计算得到可信度数值后, 设定阈值 β 和 γ, 节点可信度评估结果由(4)式给出。

$$\begin{cases} 0 \leqslant P(t) < \beta, & t\text{为高可信节点} \\ \beta \leqslant P(t) < \gamma, & t\text{为普通可信节点} \\ \gamma \leqslant P(t), & t\text{为不可信节点} \end{cases} \tag{5}$$

4.2.2 主动识别

主动识别的目的是从所有感染节点中发现并排除 Sybil 等非真实僵尸网络节点。Sybil 等良性节点是由处于防御方的安全专家为了对抗僵尸网络而部署的, 目的是渗透和接管僵尸网络, 缓解并遏制僵尸网络攻击威胁。因此在通常情况下, 这些节点不被允许参与真实的攻击活动。对于收到的攻击命令, Sybil 等良性节点通常采取不响应或者限制恶意流量的规模和速率等方式, 从而尽量避免对真实网络造成危害[17]。据此, 有理由提出以下假设:

假设: Sybil 等被防御方控制的非正常节点受到道德和法律的约束, 不能参与或不能过多参与真实的网络攻击活动。

基于此假设, 如图 4 所示, DSBot 采用以下的主动识别方法:

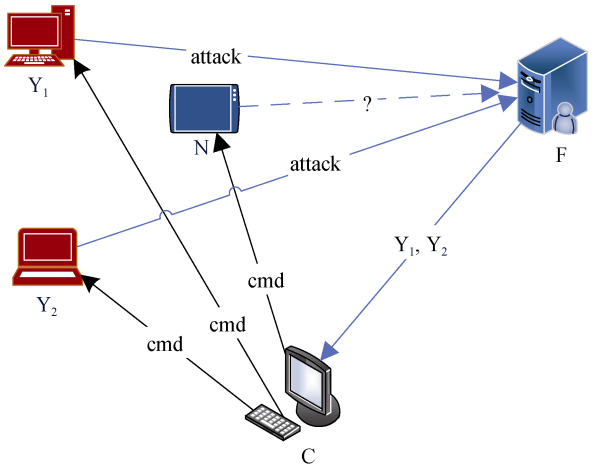

图 4 主动识别方法

Figure 4 Active identification strategy

1) F 为 DSBot 中的完全可信节点, 在网络中被设置为感知节点, 作为虚假的攻击目标接收来自 bot 节点的恶意攻击流量。

2) 僵尸控制者(在 DSBot 中由高可信的骨干节点充当局部网络的实际控制者)下达命令(*cmd*), 要求所有 bot 节点(图中以 Y_1, Y_2, N 为例)向感知节点 P 发起大流量攻击(attack)。

3) Sybil 等良性节点(N)受到道德和法律约束, 一般不会对目标发送真实的恶意攻击流量(在图中由虚线标识)。

4) 感知节点 P 根据受到的攻击情况, 向僵尸网络控制者反馈攻击流量的源 IP、持续时长等信息, 确认 Y_1, Y_2 为真实网络节点。节点 N 未发送恶意流量或减小了流量规模, 则可判定其为非真实节点。

4.3 分阶段动态感染策略

如图 5 所示, 结合主被动的探测或识别方法, DSBot 在资源拓展阶段采取分阶段动态感染策略。当僵尸网络扫描发现新的可感染目标后会第一时间关闭目标节点上其他的竞争性 bot 进程, 随后僵尸控制者(高可信骨干节点)向目标中注入被动探测模块。然后, 僵尸控制者根据探测程序返回的结果来决定是否向目标注入攻击模块, 以便进行下一步的主动识别。若目标节点通过了基于主动识别的真实性检查, 则将其正式转化为 DSBot 的僵尸节点并向其注入完整的 bot 程序。需要指出的是, 为了防止节点在对抗过程中被渗透、污染和接管, 安全性评估贯穿于僵尸网络的整个生命周期, 即周期性进行可信度评估和真实性验证, 从而进一步提升终端动态对抗能力。

5 网络自组织重构机制

僵尸网络的对抗是动态持续的过程, 主要体现在节点的数量和分布不断变化。DSBot 通过节点安全性评估和分阶段感染策略等贯穿僵尸网络全生命周期的对抗措施, 主动从网络中移除不可信节点从而提高网络对抗性。因此, 各节点之间的连接状态和网络的拓扑变化需要一种机制加以动态调控。本文借鉴无线自组网和多智能体的概念和方法, 将节点看作具备独立自主决策能力的智能体。在无线自组网中, 各节点间通过距离远近和信号强弱来进行网络的构建和优化, 对应地在 DSBot 中, 选择节点状态量化数值作为“基准”, 同时考虑到节点的性能配置复杂多样, 设计尽可能简洁高效的独立决策算法和规则。

5.1 节点属性多维表示

对于网络结构相关的问题, 通常以复杂网络理论为基础, 运用图论的方法加以分析和讨论。而在僵尸网络中, 节点种类多样且状态时刻变化, 需要进一步引入复杂异构网络理论加以刻画包含拓扑特征、物理特性(如运行环境、攻防能力)和设备类别等

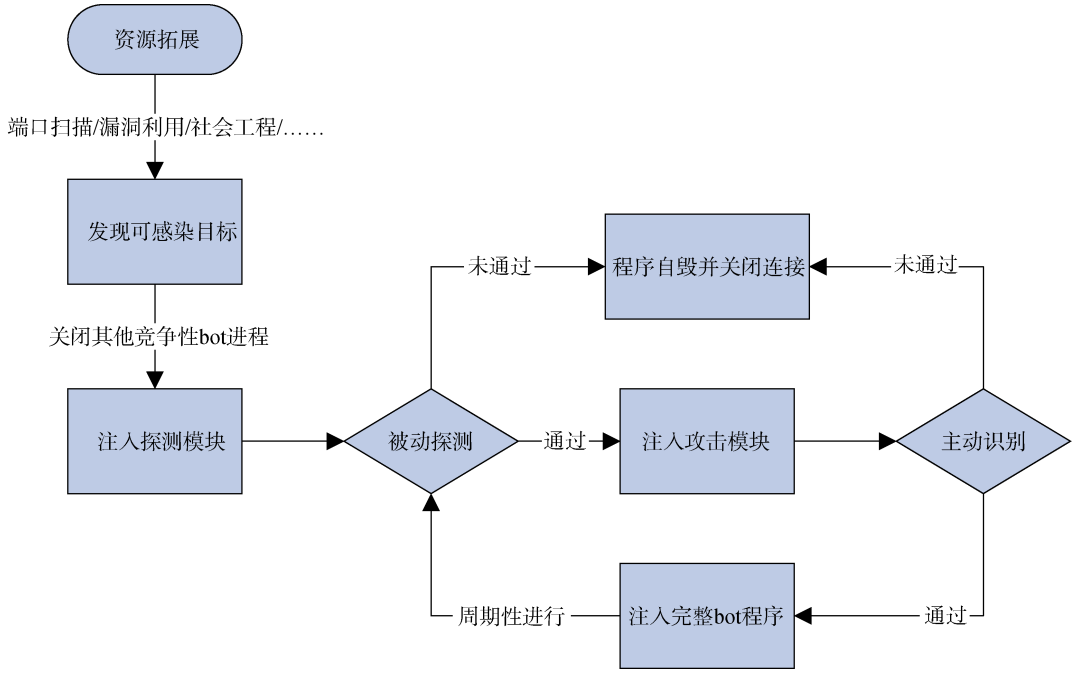

图 5　DSBot 分阶段动态感染策略

Figure 5　Multi-stage infection strategy of DSBot

的节点属性多维表示方法, 从而满足网络动态重构的需要。

在上一节的安全性评估中, 本文给出了对节点身份进行探测和识别的原理和方法, 针对的目标是待转变为僵尸节点的可感染节点。一旦节点加入网络后, 为了全周期的加以管控, 综合考虑节点多维度的各种属性, 制定节点的状态标识, 如图 6 所示。

ID	*Addr*	*E*	*P*	*R*	*Score*	*Command*

图 6　节点状态标识

Figure 6　Node status identifier

ID 表示节点的唯一标识符, Addr 表示节点的地址(由通信协议决定其格式)。Command 表示同步自其他节点的命令数据。*E* 表示节点性能评分(包括计算能力、网络环境等), *P* 表示节点可信度的客观评分, *R* 表示节点真实性的主动识别结果("1"为正常节点, "0"为非真实节点), Score 表示节点的综合评分, 由前几项数据依据一定规则计算得出, 对任一节点 i 的综合评分 Score(i)(简写为 S(i), 下同)的计算方法由(6)式给出。

$$S(n)=\left[\delta E(n)+\frac{\varepsilon}{P(n)}\right]\times R(n) \tag{6}$$

其中, $E(n)$和 $P(n)$的计算策略已由(2)、(3)式给出: 前者的数值越大表示节点的性能越高, 后者的数值越小则表示节点可信度越高, 故取其倒数, δ 和 ε 则分别为加权系数。$R(n)$作为节点真实性判断系数, 默认值为"1", 若 $R(n)$值变为"0", 则节点综合评分值也为"0"。因此, 节点状态的综合评分越高就代表该节点安全性更有保障, 同时也反映节点对于网络的重要程度, $S(n)$数值较高的节点可作为网络中的骨干节点, 承担更多职能。

另一方面, DSBot 僵尸网络建立类似"心跳包"的机制, 周期性确认其他节点的存活状态, 一旦判定对方节点离线, 其相应的 $S(n)$评分也会在当前节点的本地 $O(N_i)$缓存中被直接置"0"。

5.2　基于状态标识的动态网络框架

如图 7 所示, 本文提出的 DSBot 僵尸网络以多维属性的节点状态标识为基础, 按照"端-网-云"的多层次立体架构, 建立适用于自组织重构的动态网络模型框架。

$$SoBot=\{C_0,\ C_1,\ \ldots C_i,\ C_{i+1},\ \ldots C_t,\ \ldots\} \tag{7}$$

(7)式表明 DSBot 可从逻辑上划分为多个形如 C_i 的子集, 也可看作是在网络空间的一朵朵"云"。而 C_i 内部, 可看作是由各节点组成的网络, 形式化描述为(8)式。

$$C_i=\{N_i \mid i=0,\ 1,\ 2,\ 3,\ \ldots\} \tag{8}$$

节点部分是整个网络模型的核心, 在 DSBot 中, 节点需要维护和更新数据 $D(N_i)$, 包含两部分数据, 分别是自身的状态标识 $data(N_i)$以及同步到本地的

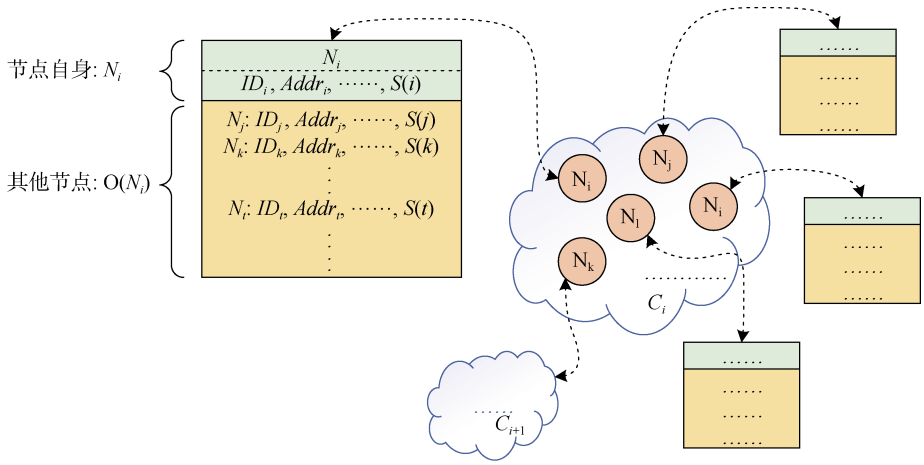

图 7　DSBot 动态网络模型

Figure 7　DSBot dynamic network model

其他节点状态信息的缓存列表 $data[O(N_i)]$, 可描述为(9)式。

$$D(N_i)=\{data(N_i),\ data[O(N_i)]\} \tag{9}$$

5.3　网络自组织重构策略

在节点属性多维表示和动态网络框架的基础上, DSBot 提出了网络自组织重构策略, 在 bot 中设计了一系列准则和算法用以提升僵尸网络的对抗性。由于网络处于时刻变化的过程中, 因此节点状态信息需要周期性地更新, 在接收到来自网络中其他节点的同步数据后, 在节点本地运行 $O(N_i)$更新算法(算法 1)。下面给出 $O(N_i)$更新算法的描述:

算法 1. $O(N_i)$更新算法.

1) WHILE(TRUE) DO
2)　t++;　　//计时器运行
3)　IF(t> T_1) THEN
4)　　同步节点信息(含 Command 字段);
5)　　更新 $O(N_i)$列表;
6)　　按节点评分 $S(i)$从高到低对列表排序;
7)　　保留 $O(N_i)$中前 c 项
8)　　t← 0;　　//计时器置“0”
9)　END　　IF
10) END WHILE

其中, T_1表示周期, 即每隔时间 T_1进行一次数据的同步, t 则表示计时器, 每完成一次更新即重新置零。其中的“同步节点信息”过程, 包括各字段的同步, 节点列表的更新和控制者命令的下发, 都是通过该动作完成。排序指的是对 $O(N_i)$中的每条节点信息按照关键字 $S(i)$的数值进行从高到低的排序, 便于后续操作。在 DSBot 中, 节点之间交互后会交换各自的缓存数据, 故 $O(N_i)$的列表长度必然会增多, 为了防止 $O(N_i)$列表体量过大导致的风险, 将列表长度限定为 c, 即任意节点本地缓存的其他节点信息数量最多为 c。节点信息同步自当前所连接的节点, 由于节点评分会动态更新, 且 $O(Ni)$列表只保留安全性较高的前 c 个节点, 所以能够尽可能保证信息来源的可靠性。完成 $O(N_i)$列表更新后, 下一步要解决的是节点间的连接度问题。关于节点间的连接度, 前文已经分析过, 关键在于维持通信效率和流量暴露之间的平衡。在 DSBot 中, 基于节点多维表示中的综合评分和以此为基础的动态网络模型, 本文提出均匀连接算法。需要指出, 这里的“均匀”不是传统意义上的平均, 而是指根据节点本身的综合评分确定其与列表中其他节点的选择性连接, 可理解为评分越高则连接度按比例增加, 即性能越好、可信度越高的节点, 相应地会在网络中连接更多其他节点, 承担更为重要的职能。记任意节点 N_i的综合评分值为 $S(i)$, 设置比例系数 η, 则该节点默认的连接数 $fix(i)$可由(10)式表示。

$$fix(i)=\eta * S(i) \tag{10}$$

可以看到, 最大连接数与当前节点的状态直接相关, 随着 $S(i)$数值的更新而变化, 需要定时更新, 周期记为 T_2。确定 $fix(i)$后, 接下来就是判断当前连接的节点数 $Q(i)$是否与之相符, 然后根据判断结果作出相应的调整。均匀连接算法(算法 2)的描述如下:

算法 2. 均匀连接算法.

1) WHILE(TRUE) DO
2)　t++;　　//计时器运行
3)　IF(t> T_2) THEN
4)　　更新 $S(i)$数值;
5)　　$fix\ (i)$ ← η*$S(i)$;

6) 获取节点 i 当前连接数 $Q(i)$;
7) IF($Q(i) > fix(i)$ THEN
8) 按顺序解除 $O(N_i)$列表中[$Q(i)$ - $fix(i)$]个节点的连接
9) ELSE
10) 按顺序增加 $O(N_i)$列表中[$fix(i)$ - Q(i)]个节点的连接
11) END IF
12) END IF
13) END WHILE

通过节点列表($O(N_i)$)更新算法和均匀连接算法, DSBot 僵尸网络能够实现节点间自组织的重构互连, 但是考虑到资源占用率和硬件计算力, 上述算法中的更新和连接策略都是周期性进行的。考虑到僵尸网络的对抗博弈是持续进行的, 因此需要加入一些实时性的规则和算法, 用以提升重构策略的健壮性和完整性。算法 2 中虽然已有断开可信度较低节点的策略, 但属于被动性质的断连, 且存在时延的脆弱性。综合上述分析, 本文基于节点综合评分 S(i)的变化设计节点主动移除算法。首先对节点的 S(i)数值作实时的跟踪监控, 一旦发现其数值低于给定的值 θ, 则执行一系列操作, 使该节点主动断开连接, 从网络中移除, 算法描述如下:

算法 3. 节点主动移除算法.
1) CHECK($S(i)$) //实时监控
2) IF($S(i)\leqslant\theta$) THEN
3) 向 $O(N_i)$中连接的节点主动推送断连信号;
4) CLEAN( ); //清除相关日志、文件和数据
5) 执行程序自毁操作;
6) END IF

至此, 本文给出了构成 DSBot 网络自组织重构策略的主要算法, 考虑到网络构建和运行的完整性, 再明确如下问题: 一是网络的初始化, DSBot 模型本质上还是基于 peer-list 的 P2P 网络, 因此不可避免地需要在 bot 程序中硬编码包含原始节点的 $O(N_i)$列表, 但是随着更新算法的周期性运行, 节点评分较低的节点都会被淘汰并从 $O(N_i)$列表中被删除, 并通过节点移除算法被网络“除名”, 原始节点也不例外, 从而尽可能保证新感染节点上的的初始化 $O(N_i)$列表中均为高可信节点; 二是节点的当前连接数 $Q(i)$, 首先明确当前的 $Q(i)$由 $S(i)$计算得到, 与 $O(N_i)$列表长度 c 没有任何关系, 在数值上 $Q(i)$小于 c, 从而给均匀连接算法保留“余量”, 避免出现无节点可连接的情形, 在具体实现时, 节点会对 $O(N_i)$列表中的每个节点设置状态值{0,1}, 即当前连接的节点标记为“1”, 未连接的节点标记为“0”。

综合上述算法 1~3 以及相应的初始化和调整机制, 本文提出DSBot的自组织重构策略, 旨在从节点可信度、网络连接度和策略简洁度这三方面提升动态对抗条件下的僵尸网络健壮性, 其本质是基于分布式自组织网络的架构, 赋予各节点一定程度的自主决策能力。自组织重构策略运作过程的一个简单示意如图 8 所示: 初始状态时, $O(N_i)$列表中的节点并没有按照节点评分排序; 运行 $O(N_i)$更新算法后, 各节点的综合评分数值得到更新并按顺序排列; 经过均匀连接算法后, $S(i)$的数值得到更新, 并且断开了与评分相对较低节点的连接; 通过节点主动移除算法对 $S(i)$数值的实时监控, 感知状态变化后, 节点 i 主动向其他节点推送断连消息, 完成网络状态的动态更新。

6 实验评估

6.1 评估指标

对于僵尸网络关键属性的评估, 通常从效率(efficiency)和韧性(resiliency)等方面进行讨论[18]。效率是僵尸网络命令控制机制的基本要求, 可视为在尽量短的时间内将命令下发给尽可能多的节点的能力。而僵尸网络的韧性, 也称为弹性, 等同于健壮性, 主要关注僵尸网络在由于节点状态改变而导致的网络动态变化时, 具备自我调节和修复的机制, 能够继续正常工作并提供服务。基于上述分析, 从量化评估的角度, 给出下列用于评价 DSBot 僵尸网络的指标。

(1) 平均等待时间 Avgw_time: 即表示僵尸网络中控制者下发命令至僵尸节点所需的时间, 由于不同的节点收到命令的时间不同, 因此用各节点的平均等待时间刻画这一指标。以 n 表示接收到命令的节点数量, t_i 表示任意节点 i 的命令等待时间, 则计算方法由(11)式给出。

$$Avgw\_time = \frac{1}{n}\sum_{i=1}^{n} t_i \tag{11}$$

(2) 命令可达率 Rate: 即表示僵尸网络中的命令发布扩散至稳定状态后, 接收到命令的节点数占僵尸网络所有节点数的比例。受网络条件、节点状态等因素制约, 命令并不能下发至所有节点, 以 m 表示经过足够长时间后接收到命令的节点数量, M 表示所有节点的数量, 则 Rate 可由(12)式得到。

$$Rate = \frac{m}{M} \times 100\% \tag{12}$$

(3) 网络连接度 Degree: 即表示僵尸节点间的连通性, 并间接反映网络的健壮性。在图论中, 度数指的是节点的邻居个数, 本文将这一属性推广到网络

图 8　自组织重构策略示意

Figure 8　DSBot dynamic network model

层面，用所有节点的平均度数作为僵尸网络整体连接度的刻画指标。以 d_i 表示节点 i 的度数，s 表示节点个数，则 Degree 可由(13)式得到。

$$Degree=\frac{1}{s}\sum_{i=1}^{s}d_i \tag{13}$$

(4) 重构稳定时间 T_R：即表示僵尸网络节点状态和网络结构变化后，通过自修复策略，重新达到稳定状态所需的时间。这一指标反映了僵尸网络调整重构的反应能力，对于实际运用具有重要意义。以 t_s 表示僵尸网络状态变化的时刻，t_e 表示僵尸网络重构完成的时刻，则重构稳定时间 T_R 由(14)式给出。

$$T_R=t_e-t_s \tag{14}$$

6.2　实验分析

考虑到真实环境下使用物理机作为僵尸节点会受到数量限制，本文通过在多台服务器上部署规模化的 Docker 实例作为节点，模拟僵尸网络集群，从而测试评估 DSBot 的相关指标。实验环境如图 9 所示，宿主机系统使用 Ubuntu 桌面系统，版本为 18.04.5, Docker 版本为 20.10.7，加载的基础镜像为官方 ubuntu(latest)，制作内嵌 bot 程序的镜像并打包后，使用 Docker Compose 进行规模化部署，通过网络配置实现 docker 间跨主机的互相通信。针对不同的评估指标，在实验中采取相应的条件设置，通过改变 T_1、T_2、η、θ 等参数的值，记录对应的实验数据并作讨论分析。实验开始前，设置好网络中每个节点的 $S(i)$值，并在改变参数重复实验时，保持各节点相同的初始状态，统一 $O(N_i)$列表长度 c=30。

6.2.1　效率评估

对于 DSBot 僵尸网络的效率，主要从平均等待时间和可达率两个指标进行评估。分析 DSBot 自组织重构机制的算法原理，在节点状态不变的情况下，不难发现参数 T_1 和 η 的值可能会对上述反映效率的指标有所影响。因此，本文假定节点状态不变，简化实验场景，在相同的初始条件下，通过设置不同的参数组合，记录对应的实验数据，从而完成评估。

在节点规模为 100 的情况下，表 4 列出了各种 T_1 和 η 的参数组合下各节点收到命令的平均等待时间。可以看到，周期 T_1 越小，连接系数 η 越大，相应

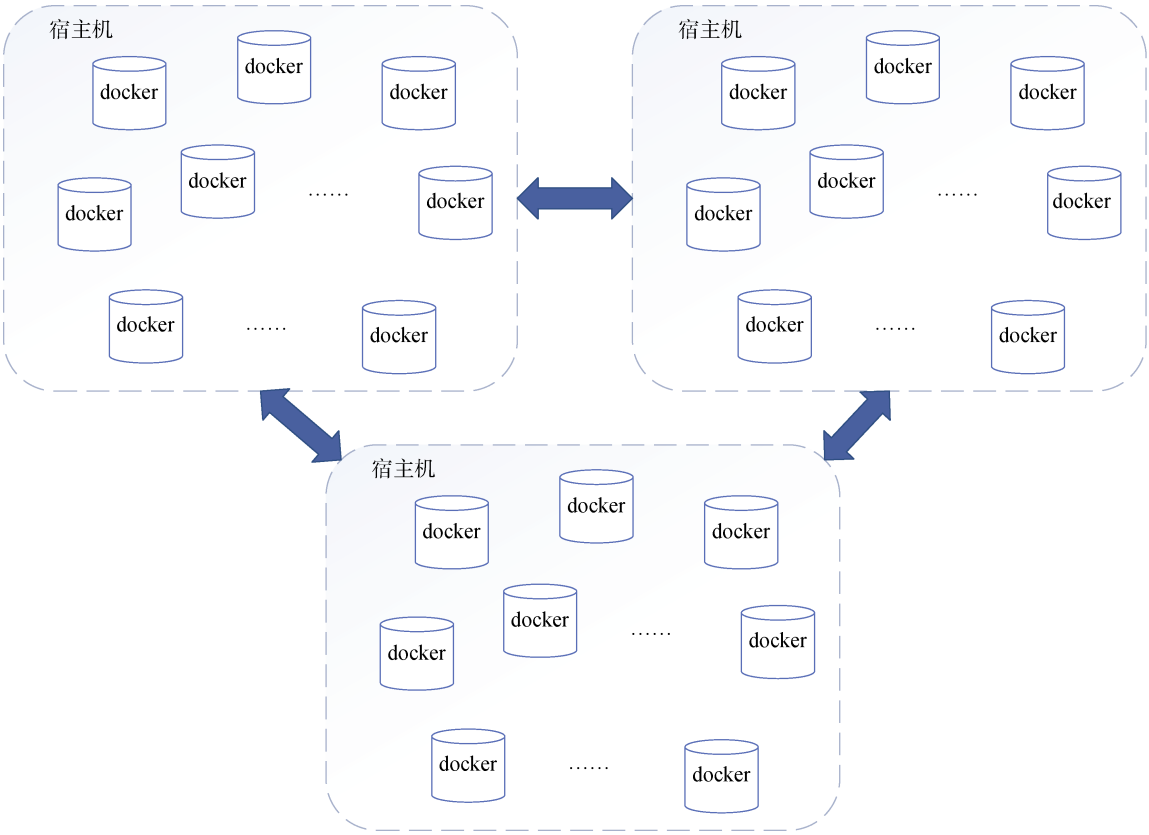

图 9　实验环境

Figure 9　Experimental setup

表 4　不同 T_1 和 η 下的平均等待时间

Table 4　The Average waiting time under different T_1 and η

序号	周期 T1(s)	连接系数 η	平均等待时间 Avgw_time(s)
1	60	0.1	125.88
2	30	0.1	67.91
3	10	0.1	23.34
4	10	0.2	19.86
5	10	0.3	18.37

的平均等待时间就越小, 即 Avgw_time 与 T_1 负相关, 与 η 正相关, 这样的实验结果与理论基本相符。需要指出的是, 上述实验过程中限定了节点规模, 因此在一定时间段内(约 200s)所有节点都能接收到命令, 未考虑更大节点规模对命令可达率的影响。为了进一步评估 DSBot 的效率, 本文固定 T_1=30s、η=0.1 的模拟条件进行实验, 测试给定时间段内不同规模下的命令可达率。如图 10 所示, 实验结果表明, 在初始条件相同的情况下, 随着网络规模的扩大, 命令可达率达到最大的时间逐渐增加, 但均在 120s 前达到最大值。上述结果表明, DSBot 的网络模型能够基本满足命令发布与传递的时延和可达率要求。

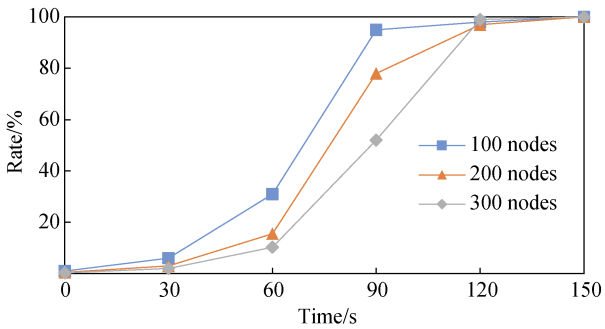

图 10　不同规模下 Rate 随时间的变化

Figure 10　The change of command reachability rate over time under different scales

6.2.2　韧性评估

通常来讲, 各节点间的连接越紧密, 网络的健壮性就越好, 但对于僵尸网络而言, 为了避免暴露, 需要加入一定的干预机制来控制网络的连接度。DSBot 僵尸网络通过自组织重构机制中的均匀连接算法, 选取合适的连接比例系数 η, 从而在对抗过程中保持网络连接度的相对稳定。为了模拟真实僵尸网络节点中的节点的损失, 本文采取随机关闭节点的策略, 即将 $S(i)$置为 θ, 这里取 θ=20。实验过程中为了缩短观察时间, 固定 T_1=T_2=10s, 节点规模为

100, 在其他条件不变的情况下, 选择不同的比例系数进行重复实验, 记录网络连接度随时间的变化。实验结果如图 11 所示, 经过随机去点后, 在自组织重构机制的作用下, DSBot 在经过一段时间的调整后, 网络连接度都会趋于原水平附近, 且连接系数 η 越大, 则网络连接度越高, 说明该机制能够起到稳定僵尸网络连接度的作用。实际上, 网络连接度还与各节点的综合评分值 $S(i)$正相关, 本文不再展开讨论。

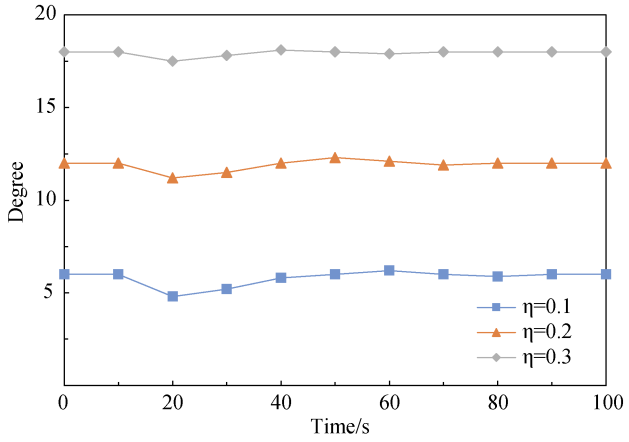

图 11 η=0.1,0.2,0.3 时 Degree 随时间的变化

Figure 11 Change of Degree over time when η=0.1, 0.2, 0.3

评估僵尸网络韧性的另一个重要指标是重构稳定时间 T_R, 即表示节点状态发生改变时, 网络进行调整重构需要的反应时间。本文采取随机变动节点 $S(i)$值的策略, 观察网络从开始重构到趋于稳定的时间, 即网络连接度开始波动到基本不变所经历的时间。实验过程中, 控制其他条件不变, 仍固定 T_1=T_2=10s, 在 η 和节点规模的不同组合下进行重复实验并观察记录数据。表 5 列出了各组实验的数据, 在节点规模一定的情况下, 连接系数 η 越高, 则重构完成时间越短; 连接系数 η 一定的情况下, 节点规模越大, 则重构完成时间越长。总体而言, 上述实验结果初步验证了 DSBot 僵尸网络模型的自组织重构机制的有效性, 且一定程度上表明了其健壮性和可扩展性。

表 5 不同条件下的重构稳定时间

Table 5 T_R under different conditions

序号	节点规模	连接系数 η	重构稳定时间 T_R(s)
1	100	0.1	82.28
2	100	0.2	57.19
3	100	0.3	30.34
4	200	0.3	46.93
5	300	0.3	60.41

由于相关工作(如文献[4,10]等)在实验部分选取的评估指标不同, 包含许多自定义的属性特征, 且实现代码不开源, 因此难以进行有效比较。另一方面, 本文区别于现有研究普遍采用的建模仿真方法, 而是利用半实物化的方式构建小型集群模拟僵尸网络真实节点运行过程, 实验手段本身就具备一定的创新性。特别地, 本文从僵尸网络构建和运用的实际情况出发, 选取和定义的评估指标能够较为全面地反映网络的综合性能。

7 防御策略分析

本文从攻击者角度提出了一种高对抗的僵尸网络理论模型, 该模型基于终端感知策略, 提升了网络的抗分析、抗污染和防渗透等能力, 并以此为基础, 通过自组织重构机制进一步增强网络的健壮性和抗毁性。针对这类潜在的新型高对抗僵尸网络, 讨论其可能的防御策略, 先于攻击者提出有效的防御措施, 具有十分重要的现实意义。常见的僵尸网络防御措施可概括为终端清除、命令控制服务器打击和命令控制过程对抗和三大类[3]。下面分别讨论这 3 种防御方法对 DSBot 僵尸网络模型的影响和作用。

(1) 终端清除。终端清除是指挖掘利用 Bot 程序自身漏洞或联合厂商对已感染的终端进行定位和清理。DSBot 僵尸网络采取分阶段动态感染策略, 一旦检测到环境异常, 就会停止感染并自毁程序, 理论上可以避免 Bot 程序被捕获分析, 从而不会暴露自身存在的漏洞。不过, 防御方可以通过对攻击流量进行溯源分析, 从而定位到感染终端, 但此时僵尸网络已经发动过攻击行动, 清除 Bot 程序可以避免僵尸网络发起下一次攻击活动, 对于已经或正在发生的恶意行为无能为力。

(2) 命令控制服务器打击。DSBot 作为完全分布式的非结构化 P2P 网络模型, 底层通信是基于优化的节点列表实现的, 因此不存在传统意义上的命令控制服务器。在 DSBot 僵尸网络中虽然存在节点综合评分较高的节点, 但各节点在维持网络通信过程中的地位是相同的, 在防御方对节点进行打击而致其关停后, DSBot 可以通过自组织重构机制在一定程度上弥补由节点损失而引起的连接度下降。不过, 考虑到高性能节点在执行攻击任务中所能发挥的重要作用, 关停重要节点对于 DSBot 僵尸网络还是能起到一定的防御作用。

(3) 命令控制过程对抗。常见的 P2P 僵尸网络命令控制过程对抗手段有 Sybil 攻击、索引污染和 peer-list 污染等。这些手段的本质是在网络中布置大

量防御方控制的节点, 从而渗透、污染、劫持或关闭僵尸网络。DSBot 模型通过节点安全性评估机制, 能够识别节点身份, 理论上能够免疫上述对抗手段。然而, 考虑到节点类型多样, DSBot 在节点通信过程中并未使用加密算法, 存在控制指令和攻击命令明文暴露的风险, 防御方能够在命令尚未下达至节点前预判僵尸网络的行动。另一方面, 虽然 DSBot 把网络连接度控制在合理范围内, 但自组织重构过程中仍然会不可避免地产生异常流量, 防御方可利用这一点进行感知和检测。

8 结论

网络空间斗争形势日益严峻, 僵尸网络作为大规模攻击活动的基础设施平台, 成为关注热点之一。僵尸网络攻击者始终致力于构建更为健壮和可用的僵尸网络 C&C 机制, 为此不断尝试各种架构和协议。与此同时, 研究人员通过分析僵尸网络机理特性, 揭示演化规律, 预测新型僵尸网络模型, 试图先于攻击者提出相应的防御解决方案, 围绕僵尸网络的攻防博弈由此被推向了新阶段。本文通过对 P2P 结构僵尸网络的脆弱性分析, 针对性地提出动态自组织P2P僵尸网络模型DSBot, 旨在从终端对抗和网络重构两方面提升僵尸网络的健壮性, 并简单分析讨论了相应的防御策略。本文认为, 终端感知和网络重组能力的提升是新型僵尸网络的演化趋势。重视这一潜在的网络安全威胁, 提出切实有效的应对措施, 是目前亟需深入开展的研究和工作。

致 谢 在此谨向对本文工作提出建议的评审专家以及对本文提出指导的老师、同学表示感谢。

参考文献

[1] Vormayr G, Zseby T, Fabini J. Botnet Communication Patterns[J]. *IEEE Communications Surveys & Tutorials*, 2017, 19(4): 2768-2796.

[2] Fang B X, Cui X, Wang W. Survey of Botnets[J]. *Journal of Computer Research and Development*, 2011, 48(8): 1315-1331.
(方滨兴, 崔翔, 王威. 僵尸网络综述[J]. *计算机研究与发展*, 2011, 48(8): 1315-1331.)

[3] Li K, Fang B X, Cui X, et al. Study of Botnets Trends[J]. *Journal of Computer Research and Development*, 2016, 53(10): 2189-2206.
(李可, 方滨兴, 崔翔, 等. 僵尸网络发展研究[J]. *计算机研究与发展*, 2016, 53(10): 2189-2206.)

[4] Wang P, Sparks S, Zou C C. An Advanced Hybrid Peer-to-Peer Botnet[J]. *IEEE Transactions on Dependable and Secure Computing*, 2010, 7(2): 113-127.

[5] Hund R, Hamann M, Holz T. Towards Next-Generation Botnets[C]. *2008 European Conference on Computer Network Defense*, 2008: 33-40.

[6] White A, Tickle A, Clark A. Overcoming Reputation and Proof-of-Work Systems in Botnets[C]. *2010 Fourth International Conference on Network and System Security*, 2010: 120-127.

[7] Marupally P R, Paruchuri V. Comparative Analysis and Evaluation of Botnet Command and Control Models[C]. *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 2010: 82-89.

[8] Arora D, Godkin T, Verigin A, et al. Assessing Trade-Offs between Stealthiness and Node Recruitment Rates in Peer-to-Peer Botnets[C]. *2012 Seventh International Conference on P2P*, *Parallel*, *Grid*, *Cloud and Internet Computing*, 2012: 148-155.

[9] Anagnostopoulos M, Kambourakis G, Drakatos P, et al. Botnet Command and Control Architectures Revisited: Tor Hidden Services and Fluxing[M]. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017: 517-527.

[10] Yin J, Cui X, Fang B X, et al. A Pollution-Resilient Hybrid P2P Botnet[J]. *Journal of Cyber Security*, 2018, 3(1): 68-82.
(尹捷, 崔翔, 方滨兴, 等. 一种抗污染的混合 P2P 僵尸网络[J]. *信息安全学报*, 2018, 3(1): 68-82.)

[11] Böck L, Vasilomanolakis E, Wolf J H, et al. Autonomously Detecting Sensors in Fully Distributed Botnets[J]. *Computers & Security*, 2019, 83: 1-13.

[12] Rodríguez-Gómez R A, Maciá-Fernández G, García-Teodoro P. Survey and Taxonomy of Botnet Research through Life-Cycle[J]. *ACM Computing Surveys*, 2013, 45(4): 1-33.

[13] Zhu Z S, Lu G H, Chen Y, et al. Botnet Research Survey[C]. *2008 32nd Annual IEEE International Computer Software and Applications Conference*, 2008: 967-972.

[14] Wang P, Wu L, Aslam B, et al. A Systematic Study on Peer-to-Peer Botnets[C]. *2009 Proceedings of 18th International Conference on Computer Communications and Networks*, 2009: 1-8.

[15] Davis C R, Fernandez J M, Neville S, et al. Sybil Attacks as a Mitigation Strategy Against the Storm Botnet[C]. *2008 3rd International Conference on Malicious and Unwanted Software*, 2008: 32-40.

[16] Antonakakis M, April T, Bailey M, et al. Understanding the Mirai Botnet[C]. *The 26th USENIX Conference on Security Symposium*, 2017: 1093-1110.

[17] Wang P, Wu L, Cunningham R, et al. Honeypot Detection in Advanced Botnet Attacks[J]. *International Journal of Information and Computer Security*, 2010, 4(1): 30-51.

[18] Li J, Ehrenkranz T, Kuenning G, et al. Simulation and Analysis on the Resiliency and Efficiency of Malnets[C]. *Workshop on Principles of Advanced and Distributed Simulation*, 2005: 262-269.

赵昊 于 2019 年在数学工程与先进计算国家重点实验室计算机科学与技术专业获得硕士学位。现在数学工程与先进计算国家重点实验室网络空间安全专业攻读博士学位。研究领域为网络安全。Email: 754094629@qq.com

刘潮歌 于 2019 年在中国科学院大学信息安全专业获得博士学位。现任中国科学院信息工程研究所副研究员。研究领域为恶意代码、Web 安全和网络攻击追踪溯源。Email: liuchaoge@iie.ac.cn

赵耘田 于 2013 年在信息工程大学计算机技术专业获得硕士学位。现任数学工程与先进计算国家重点实验室助教。研究领域为网络空间安全。Email: 545108726@qq.com

舒辉 于 2001 年在信息工程大学计算机软件与分析专业获得博士学位。现任数学工程与先进计算国家重点实验室教授。研究领域为网络空间安全。Email: shuhui123@126.com

邢颖 于 2010 年在河南师范大学计算机科学与技术专业获得硕士学位。现在数学工程与先进计算国家重点实验室计算机科学与技术专业攻读博士学位。研究领域为网络安全检测。Email: xingying_vip@163.com