一种DRDoS协同防御模型研究

张明清 揣迎才 唐 俊 孔红山
(信息工程大学 郑州 450004)

摘 要 针对现有DRDoS防御方法反应滞后和过滤不全面的问题,基于协同防御思想,提出了一种DRDoS协同防御模型——HCF-AST。该方法通过协同式自学习算法,实现设备间DRDoS防御知识的共享,过滤来自外网的攻击流量;并引入入侵追踪技术,与入侵检测和过滤技术协同,定位并阻断内网攻击源。仿真结果表明,该模型能够及时发现并有效消除来自内外网的DRDoS攻击。

关键词 DRDoS攻击,协同防御,自学习,入侵追踪,HCF-AST模型
中图法分类号 TP391.9 **文献标识码** A

Study on Collaborative Defence Model of DRDoS

ZHANG Ming-qing CHUAI Ying-cai TANG Jun KONG Hong-shan
(Information Engineering University, Zhengzhou 450004, China)

Abstract According to the defects of existing defence method on response lag and incomprehensive filtering, one collaborative defence model of DRDoS was proposed, based on collaborative defence theory. A collaborative self-learning algorithm was designed, which made it possible to share defence knowledge with other agents and could filter attack flows from external network. Intrusion tracking technology was used, together with intrusion detection and intrusion filtering, attack source in the internal network would be located and blocked. Simulation results show that this model could timely detect and effectively eliminate attack flows from both Internal and external network.

Keywords DRDoS attack, Collaborative defence, Self-learning, Intrusion tracking, HCF-AST model

1 引言

分布式反射拒绝服务攻击(Distirbuted Reflectjon Denial of Service Attack)[1]是一种特殊的DDoS攻击方式,它不需要在实际攻击之前占领大量傀儡机,而是巧妙地利用反弹服务器群作为跳板来将洪水数据包反弹给目标地址,是危害性更强的一种攻击。虽然目前对于DRDoS的研究已有不少,但是在实际的互联网上此类攻击的检测与防御效果不够理想,因此针对DRDoS的防护研究也成为了一个重要的课题。

Peng等人[2]提出了源IP地址监视的DRDoS检测方法,该方法把大量新的IP地址作为检测特征来监视,但若攻击者用我们熟知的IP地址来攻击,这种方法就无能为力了;Jin等人[3]也提出了一种在反射端检测和过滤的方法,其基于跳数和IP源地址的映射,在反射端的检测方法尽管可以减少DRDoS攻击,但是要在所有的潜在反射端都配置这样的检测法是不实际的;尽管安全状态检测方法[4]对区分异常包很有效,但是全状态检测法要求状态表处于不断的自动更新中,这样整个检测过程就变得很复杂;基于请求回应包的Hiroshima Tsunoda检测模型[5]对DRDoS攻击包有很好的检测效果,但是若对所有的回应包,不管是正常包还是攻击包都进行检测,反应将很慢,且系统开销大。文献[6]提出一种改进的DRDoS检测算法,该算法能够针对性地检测到利用特定网络服务产生的DRDoS攻击流量,但这种方法通过网络中请求包与回应包的比例异常来发现攻击,检测行为发生在攻击流量被反射后,防御反应比较滞后;Wang Hal. ning等人[7]提出了一个轻量级的可以粗略进行DDoS防御的防御模型——HCF(Hop Count Filtering),这种方法基于请求包的异常行为进行检测,能够及时发现攻击,且在训练(自学习)完全的情况下能够以较低的系统开销过滤掉大部分的DRDoS攻击,但是无法有效消除在与受害主机相似的网络拓扑位置上发起的攻击。

针对DRDoS防御的上述缺点,本文基于协同防御思想,提出一种DRDoS协同防御模型——HCF-AST(Hop Count Filtering-Attack Source Tracing)。该模型把DRDoS检测和过滤设备部署在网关,通过协同式自学习过程,实现设备间DRDoS防御知识的共享,并引入入侵追踪技术,弥补防火墙单独防御的不足。

2 DRDoS攻防原理分析

一般DDoS攻击中,受控主机伪造的每个IP包的IP地址是在一个指定网段内通过随机函数产生的,而DRDoS攻

到稿日期:2012-11-30 返修日期:2013-03-18
张明清(1961-),男,副教授,硕士生导师,主要研究方向为系统建模与仿真,E-mail:chuaiyingcai@126.com;揣迎才(1985-),男,硕士生,主要研究方向为网络安全仿真;唐 俊(1976-),男,硕士,讲师,主要研究方向为系统工程;孔红山(1981-),男,硕士,讲师,主要研究方向为系统建模与仿真。

击中受控主机伪造的每个 IP 包的源 IP 地址却是真实的，并对应网上的一台主机或路由器，即攻击的目标。因此，DRDoS 攻击的隐蔽性更强，危害也更大[8]。如图 1 所示，多个分布式攻击主机伪造目标主机的 IP 地址向多个服务器发送服务请求，经服务器反射后，大量的回应报文涌向目标主机，造成通信堵塞或工作瘫痪。

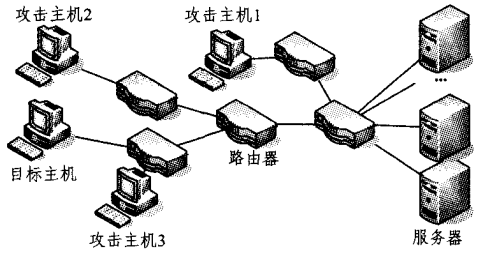

图 1　DRDoS 攻击示意图

根据图中攻击主机与目标主机拓扑位置的不同，将 DRDoS 攻击分为 3 类：①与目标主机拓扑位置不同，如攻击主机 1，攻击路径中到达某个路由器的跳数与目标主机截然不同；②与目标主机拓扑位置相似，如攻击主机 2，攻击路径中到达某个路由器的跳数与目标主机相同，但它们位于不同的子网；③与目标主机拓扑位置相同，如攻击主机 3，与目标主机位于同一子网，拓扑位置完全相同。在 DRDoS 攻击中，第①类攻击方式所占的比例最大，第②类次之，第③类较少。

基于以上分析，DRDoS 防御应该遵循以下原则：(1)不宜采用 BPS(Bit Per Second)过滤，因为该方法在过滤攻击流量的同时，可能把合法的服务请求也过滤掉，如第③类攻击；(2)一般情况下，采用“黑名单”过滤，除非短时间内发现大量新的 IP 地址；(3)采用协同防御技术，实现不同防御手段的协同，以及同一防御手段的不同设备间的协同，提升整体防御能力。

3　协同式 HCF 自学习算法

Wang Hai-ning 等人提出的 HCF 过滤方法[7]，各防御设备独立工作，其感知能力有限，即自学习范围有限，只能收集经过自己的流量信息。如果某个防御设备内没有与当前攻击相关的防御信息，就会导致防御失败，如传统 HCF 方法只能过滤掉第①类 DRDoS 攻击。为此，根据协同防御思想，设计了协同式自学习算法来实现防御设备间自学习知识的共享，以解决自学习不完全的问题。

一般地，向部署在网关的入侵检测设备和防火墙中添加 HCF 规则，实现 DRDoS 攻击的检测与过滤。协同式 HCF 自学习算法主要包括以下参量：

- 自学习条件 C_1：不存在网络攻击；
- 自学习时机 C_2：定时学习，也可根据具体攻防状况加强学习；
- 学习时长 T：视网络规模大小而定；
- 学习内容：SNFL=($srcAddr$,hop)；SNFL 为标准网络流量库(Standard Network Flow Library)，$srcAddr$、hop 分别为数据包的源地址和跳数信息。

协同式 HCF 自学习过程：

当 C_1 满足，C_2 到达时，开始周期为 T 的自学习过程；当数据包到达时，解析并提取 $srcAddr$ 和 hop 信息，关联存入 SNFL；当周期 T 结束时，各防御设备提取自身的 SNFL 信息，写入协同消息，经 h 跳后，传递到其它防御设备；当其它防御设备收到协同消息后，提取消息携带的防御知识，变量 $srcAddr$ 值不变，变量 hop 加 h 后，存入本地 SNFL。

在防御模式下，一旦检测到攻击，就启动 HCF 过滤，将当前流量与 SNFL 匹配，匹配失败，则认为是攻击数据包。

采用协同式自学习算法的 HCF 过滤方法，实现了防御设备间自学习知识的共享，克服了自学习不完全的问题，能够有效地过滤第①和第②类 DRDoS 攻击，但无法应对第③类 DRDoS 攻击。

4　HCF-AST

无论是传统的 HCF 方法还是改进的协同式 HCF 方法，都无法检测并过滤第③类 DRDoS 攻击，也不能盲目地采用 BPS 方法，因为该方法过滤掉攻击流量的同时，也过滤掉了来自目标主机的合法服务请求，间接达到了攻击目的。为此，引入攻击源追踪(attack source tracing，AST)技术，提出 HCF-AST 协同防御模型(如图 2 所示)，它包括入侵检测、防火墙和入侵追踪 3 个子系统，各子系统内部与系统之间通过信息流和控制流进行协同，其中，信息流主要是协同防御知识，而控制流主要是互操作指令。图 2 中虚线表示子系统内部的协同，实线表示子系统之间的协同。

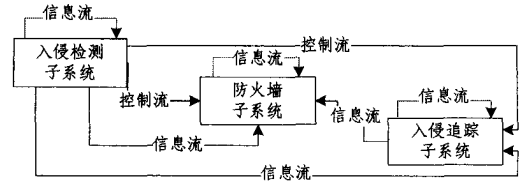

图 2　HCF-AST 协同防御模型

具体的 HCF-AST 协同防御流程如图 3 所示。

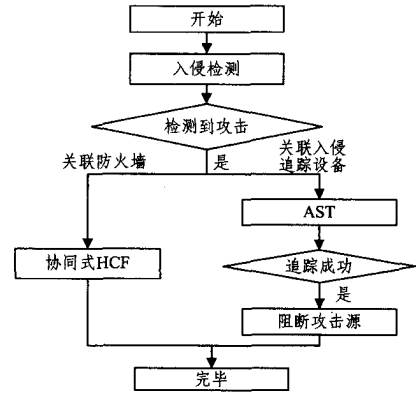

图 3　HCF-AST 协同防御流程

向部署在网关的入侵检测设备中添加 HCF 规则，当它检测到 DRDoS 攻击后，同时向与其关联的防火墙和入侵追踪设备发送入侵警报消息，启动协同式 HCF 攻击过滤及攻击源追踪(AST)；当追踪到攻击源后，采取措施阻断攻击源，消除第③类 DRDoS 攻击的影响。

在该模型中，设置设备间协同关系和确定追踪重构时机非常重要。

由于 DRDoS 攻击呈现出分布式、协作性和智能性等特点，选择合适的协同关系，既能保证有效消除攻击，又能避免不必要的系统开销。这是一个研究重点和难点，与实际网络

攻防环境密切相关，这里我们采用一种基于攻击受损关联度的强化学习算法来确定设备间的协同关系。

定义 1 攻击受损关联度 λ，表征两个设备是否都受到同一轮网络攻击的影响；如果是，则 $\lambda=1$，否则 $\lambda=0$；且 λ 值可累加。如果设备 i,j 存在关联，那么可以认为设备 i 受到了攻击，设备 j 也可能受到攻击，且 λ 值越大，可能性越大。

定义 2 设 $A=\{x_1,x_2,\cdots,x_n\}$ 是防御个体集合，$F(x_i,x_j)$ 是 A 上的一个表示协同关系的二元谓词，令 $\lambda_{ij}=F(x_i,x_j),i,j=1,2,\cdots,n$。

强化学习算法流程如下：

在一个强化学习周期内，

步骤 1 第一轮攻击发生，如果设备 i 检测到攻击，则向所有其它设备广播入侵警报消息；设备 j 收到了警报消息，且自身确实受到攻击，则令 $\lambda_{ij}=1$，并向设备 i 发出反馈信号，令 $\lambda_{ji}=1$；

步骤 2 第二轮攻击发生，设备 i 检测并发送警报消息；设备 j 收到了警报消息，且自身确实受到了攻击，则令 $\lambda_{ji}+1$，并向设备 i 发出反馈信号，令 $\lambda_{ij}+1$；

如此重复，直至强化学习周期结束，得到

$$(\lambda_{ij})=\begin{bmatrix}\lambda_{11} & \lambda_{12} & \cdots & \lambda_{1n}\\ \lambda_{21} & \lambda_{22} & \cdots & \lambda_{2n}\\ \cdots & \cdots & \cdots & \cdots\\ \lambda_{n1} & \lambda_{n2} & \cdots & \lambda_{nn}\end{bmatrix}$$

将其称为 F 在 A 上的逻辑关系矩阵，即攻击受损关联度矩阵，也可表示为 $F(A)$。

强化学习周期结束后，当设备检测到攻击时，读取 $F(A)$，选择向 λ 值较大的设备发送协同消息。

基于受损关联度的强化学习算法可以应用于入侵检测设备与防火墙之间，以及入侵检测设备与入侵追踪设备之间的协同关系选择。该算法的目的是把入侵警报消息直接发送到最可能受到攻击的其它设备，避免不必要的系统开销，提高交互效率，增强协同防御效果。

另外，攻击源追踪的关键在于确定重构时机，也可以说是重构收敛时间，即收集到多少标记数据包后开始重构攻击路径，因为时机过早或过晚都可能导致重构失败。文献[9]将收敛时间的计算问题看作是票券收集问题，给出平均收敛时间的上界是：

$$E[X]\leqslant\frac{\ln(d)}{p(1-p)^{d-1}}$$

式中，d 为可能的平均攻击路径长度，p 为标记概率，取 $p=1/d$。

5 仿真实验

为了验证 HCF-AST 协同防御模型的有效性，设计了如图 4 所示的网络攻防仿真场景。其中，Cli_1－Cli_21 是 HTTP 客户端，server1－server3 是 HTTP 服务器，detector1－detector7 是入侵检测设备，filter1－filter7 是防火墙设备，investigator1－investigator3 是入侵追踪设备，target 是受害主机，attacker1－attacker4 是攻击主机。分别在入侵检测和防火墙设备中添加协同式 HCF 检测及过滤算法，在入侵追踪设备中添加包标记追踪算法。

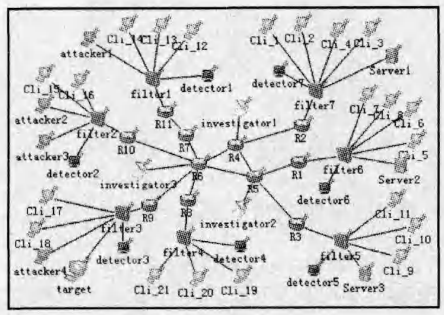

图 4 网络攻防场景

在该仿真场景中，分别部署了 3 种防御策略：①传统 HCF 策略，各设备独立防御；②协同式 HCF 策略，各防御设备间共享自学习知识；③HCF-AST 策略。

设置仿真时间为 300s。从 147s 开始，4 个 attacker 以 target 为攻击对象，发动 DRDoS 攻击。设置仿真数据统计量，分别配置 3 种防御策略，运行仿真。

下面，以 filter1 例，描述设备间防御知识的共享情况。表 1 所列为 filter1 的 HCF 过滤知识表。在传统 HCF 策略下，filter1 只能获取局部防御知识（第 1－6 项）；在协同的情况下，可以获取更全面的防御知识（增加了第 7－10 项），而第 10 项正是 filter1 过滤 DRDoS 攻击需要的防御知识。

表 1 HCF 过滤知识表(filter1)

序号	源地址	跳数	获取时间(s)
1	131	1	0.5
2	92	1	2
3	93	1	2
4	94	1	2
5	72	7	2.1
6	73	7	3.6
7	134	6	101.56
8	97	6	101.56
9	98	6	101.56
10	102	6	101.56

图 5 所示为 investigator3 的攻击源追踪结果。通过攻击路径重构，找出距离攻击源最近的 3 个路由器的 IP 地址，并与其关联的防火墙协同，阻断攻击流。

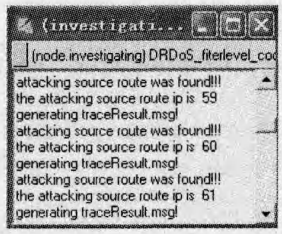

图 5 investigator3 攻击源追踪结果

正是由于设备间防御知识的共享和攻击源追踪的引入，使防御体系产生了整体防御效果。

图 6 所示为 3 种防御策略下 target 收到的回应包速率统计数据，表 2 为 3 种策略下防御效果的量化统计情况。攻击开始后，大量攻击流量经服务器反射后涌向受害主机 target，造成其回应包速率远远高于正常值。传统 HCF 策略下，target 收到的回应包速率从 189s 开始减少，而由于协同的原因，另外两种防御策略下回应包速率提前 6s 开始减少。

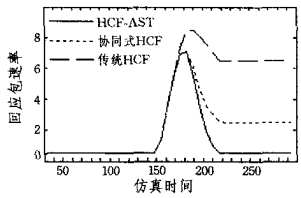

图 6　3 种防御策略下 target 回应包速率统计

表 2　3 种策略防御效果比较

防御策略类型	回应包速率最大值(个/s)	攻击开始减弱时间(t)	过滤攻击类型
传统 HCF	8.5	189	第①类
协同式 HCF	7.0	183	第①②类
HCF-AST	7.0	183	第①②③类

比较 3 种防御策略下的回应包速率最高值可知，传统 HCF 策略最高，另外两种防御策略稍低。比较图 6 中 3 条曲线(183s 后)的斜率可知，HCF-AST 策略下攻击流量减弱最快，协同式 HCF 策略次之，传统 HCF 策略最差。这是因为，在传统 HCF 策略下，只有 detector6 能检测到攻击，并令 filter6 过滤攻击流量，而在其它两种策略中，防御设备之间共享防御知识，使距离攻击源较近的 detector1 和 detector2 能够较早地检测到攻击并开始过滤，把攻击流量阻断在攻击源处，大大减少了进入网络内部的攻击流量。

在传统 HCF 策略下，防火墙只能消除第①类 DRDoS 攻击；在协同式 HCF 策略下，防火墙能够有效消除第①、②类 DRDoS 攻击；而在 HCF-AST 策略下，经过 53s，协同防御体系消除了所有的 DRDoS 攻击。另外，HCF-AST 防御发生在攻击流量反射前，在一定程度上弥补了传统方法反应滞后的不足。

总之，HCF-AST 协同防御模型能够及早发现并较快过滤所有类型的 DRDoS 攻击，有效地减少了攻击造成的影响损失。

结束语　本文提出的 DRDoS 协同防御模型，实现了不同防御手段及同一防御手段不同设备间的协同合作，充分利用有限的防御资源，弥补了设备单独防御的不足，能够有效消除所有类型的 DRDoS 攻击，并且防御反应速度较以往方法有较大提高。仿真实验证实了防御模型的有效性。在该模型中，防御设备间协同关系的设置至关重要，文中只考虑了不同类防御设备间的交互协同，而实际上，同类防御设备之间也可以通过协同共享防御资源，这将是今后需要关注的方向。

参 考 文 献

[1] 严芬，高玉龙，殷新春. DDoS 攻击检测进展研究[J]. 苏州大学学报：自然科学版，2011，27(13)：35-41

[2] Peng T. Detecting reflector attacks by sharing beliefs[J]. IEEE Global Telecommunication Conference，2003，46：1358-1362

[3] Jin C，Wang H，Shin K，et al. An effective defense against spoofedtraffic[C]// ACM International Conference on Computer and Communications Confercnce Security. 2003，10：30-41

[4] Noureldien N A，Osinan I M. A stateful inspection module architecture[C]//IEEE/RENCON. 2000，2：259-265

[5] Tsunoda H，Ohm K，Yamamoto A，et al. Detecting DDoS attacks by a simple response packet confirmation mechanism[J]. Computer Communications，2008：3299-3306

[6] 何雪妮. 一种改进的 DRDoS 检测算法[J]. 自动化与仪器仪表，2012，161(3)：150-151，155

[7] Wang Hai-ning，Jin Cheng，Shin K G. Defense against spoofed IP traffic using hop-count filtering[J]. IEEE/ACM Trans on Networking，2000，15(1)：40-53

[8] 张永花，崔永君. DRDoS 攻击及其防御技术研究[J]. 计算机安全，2009，4：53-55

[9] Mitzenmacher M，Upfal E. Probability and Computing，Randomized Algorithms and Probabilitic Analysis[M]. Cambridge：Cambridge University Press，2005：217-223

(上接第 98 页)

[21] Davi L，Sadeghi A，Winandy M. ROPdefender：A detection tool to defend against return-oriented programming attacks [C]// Proceedings of the 6th ACM Symposium on Information，Computer and Communications Security. Hong Kong：ACM New York Press，2011：40-51

[22] Chi-Keung Luk，Cohn R，Muth R，et al. Pin：Building Customized Program Analysis Tools with Dynamic Instrumentation [C]// Proceedings of 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI). Chicago：ACM New York Press，2005：190-200

[23] Adobe CoolType SING Table 'uniqueName' Stack Buffer Overflow [EB/OL]. http://www.exploit-db.com/exploits/16619/，2010-09-25

[24] Integard Pro 2.2.0.9026 (Win7 ROP-Code Metasploit Module) [EB/OL]. http://www.exploit-db.com/exploits/15016/，2010-09-25

[25] MPlayer (r33064 Lite) Buffer Overflow + ROP exploit [EB/OL]. http://www.exploit-db.com/exploits/17124/，2011-04-06

[26] Checkoway S，Davi L，Dmitrienko A. Return-Oriented Programming without Returns [C]//Proceedings of ACM Conference on Computer and Communications Security (CCS). Chicago：ACM New York Press，2010：559-572

[27] Zovi D D. SOURCE Boston 2010：Practical return-oriented programming[EB/OL]. http://trailofbits.files.wordpress.com/2010/04/practical-rop.pdf

[28] Bhatkar S，Sekar R，DuVarney D C. Efficient Techniques for Comprehensive Protection from Memory Error Exploits [C]// Proceedings of 14th USENIX Security Symposium. Baltimore：USENIX Association，2005：105-120

[29] Roglia G，Martignoni L，Paleari R，et al. Surgically returning to randomized lib(c) [C]//Proceedings of Annual Computer Security Applications Conference. Honolulu：ACM New York Press，2009：60-69

[30] Chiueh T-C，Hsu F-H. RAD：A compile-time solution to buffer overflow attacks [C]// Proceedings of the 21st International Conference on Distributed Computing Systems. Phoenix：IEEE Computer Society，2001：409-420

[31] Schwartz E J，et al. Q：exploit hardening made easy[C]// Proceedings of 20th USENIX Security Symposium. San Francisco：USENIX Association，2011：379-394