基于移动 agent 的 DDoS 攻击协同防范技术研究

吴姗姗,李 俊
(南京航空航天大学 信息科学与技术学院,江苏 南京 210016)

摘 要:分布式拒绝服务攻击(distributed denial-of-service,DDoS)已经对 Internet 的稳定运行造成了很大的威胁。近两年来,DDoS 的攻击方法和工具变得越来越复杂,越来越有效,追踪真正的攻击者也越来越困难。从攻击防范的角度来说现有的技术仍然不足以抵御大规模的攻击。文中通过分析 DDoS 攻击原理以及 DDoS 攻击行为,提出了一个基于移动 agent 的分布式协同入侵检测模型。该模型通过构建本地入侵检测模块和反 DDoS 实体模块来协同对分布式拒绝访问攻击形成大面积网络预警机制,以达到在陷于大规模分布式拒绝访问攻击时,能够最小化检测和反应时间,并进行自动响应。

关键词:DDoS;移动 agent;入侵检测
中图分类号:TP393.08 **文献标识码**:A **文章编号**:1005-3751(2006)04-0230-04

Cooperative Detection to DDoS Attacks Based on Mobile Agent

WU Shan-shan, LI Jun
(School of Info. Sci. and Techn., Nanjing Univ. of Aeronautics and Astronautics, Nanjing 210016, China)

Abstract: Distributed denial-of-service attack (DDoS) brings a very serious threat to the stability of the Internet. In the last two years, it is discovered that DDoS attack methods and tools are becoming more sophisticated, effective, and also more difficult to trace to the real attackers. However, on the defense side, current technologies are still unable to withstand large-scale attacks. In this paper, first analyses the attacking rules and behaviors of DDoS, and then based on mobile agent, propose a cooperative intrusion detection framework focused on countering distributed denial-of-service (DDoS) attacks through the introduction of a distributed overlay early-warning network. The goal is to minimize the detection and reaction time, and automate responses, while involving as many networks as possible along the attack path.

Key words: DDoS; mobile agent; intrusion detection

0 引 言

分布式拒绝服务攻击(DDoS)是在传统 DoS 攻击基础上演变而成的[1]。DDoS 主要利用 Internet 上现有机器及系统的漏洞,攻占大量联网主机,使其成为攻击者的代理。当被控制的机器达到一定数量后,攻击者通过发送指令操纵这些攻击机同时向目标主机或网络发起 DoS 攻击,大量消耗其网络带宽和系统资源,导致该网络或系统瘫痪或停止提供正常的网络服务。

DDoS 攻击对传统的单一的、缺乏协作的入侵检测系统是一个很大的挑战[2]。一方面,DDoS 具有了分布式的特性,真正攻击者的位置非常灵活,发布命令的时间短而隐蔽,难以定位;攻击者往往还采用 IP 地址欺骗技术来进行攻击并安装特别设计的工具来隐藏入侵痕迹,这就为追踪攻击者的真正位置设置了障碍。另一方面,攻击者所用的协议都是一些非常常见的协议和服务,系统管理员就难以区分恶意请求和正常连接请求,从而无法有效分离出攻击数据包。目前,还没有有效的手段能够防范 DDoS 攻击。DDoS 攻击成为 Internet 中最主要的安全问题。

因此,针对 DDoS 的这些特点,文中以移动 agent 技术、协同工程、决策支持系统等技术为核心,详细探讨了一种 DDoS 攻击的协同防范系统框架及其实施方案。

1 DDoS 攻击防范基础架构

为了克服目前入侵检测系统单一的、缺乏协作的缺陷,更好地应对 DDoS 攻击,需要一个很好的分布式协同的入侵检测体系框架。在这一分布式系统框架的内部,还需要具有良好的通信能力,以支持低通信流量的可靠的安全通信。这样,就可以在本地检测攻击而不需要回溯机制。

1.1 基于移动 agent 的分布式协同框架

移动 agent 是一类特殊的 agent,它除了具有智能 agent 最基本的特性外,还具有移动性。它是一个能在异构网络中自主地从一台主机迁移到另一台主机,并可与其他 agent 自愿交互的程序。总的说来,移动 agent 具有在主机

收稿日期:2005-08-05
基金项目:国防科工委国防基础项目(S0500B003)
作者简介:吴姗姗(1981-),女,江苏南京人,硕士研究生,研究方向为计算机网络、网络安全;李 俊,研究员,硕士生导师,研究方向为计算机网络。

间动态迁移、降低网络负载、智能性、平台无关性,以及代理间的良好协作性等优点。

由于移动 agent[3]本身所固有的特性,使其在运用于分布式入侵检测中时可以很好地降低网络负载,克服网络延时,并具有与平台无关,动态适应性良好、可扩展性强等优点。基于以上考虑,文中提出了基于移动 agent 的分布式协同入侵检测系统。在设计上,系统遵循了 CERT (Computer Emergency Response Team)的样式,即每个 CERT 负责一个网络区域并和其他的 CERT 协作。在系统中,每个网络区域由本地的入侵检测系统(Local IDS)和反 DDoS 实体(Counter－DDoS Entity)共同负责管理并抵御攻击。其中,反 DDoS 实体由一组相互协作的 agent 组成,用以防御本地的网络区域的 DDoS 攻击并与临域的 agent 之间进行自主的安全信息传送,在遇到攻击时进行协同响应。系统协作如图 1 所示。

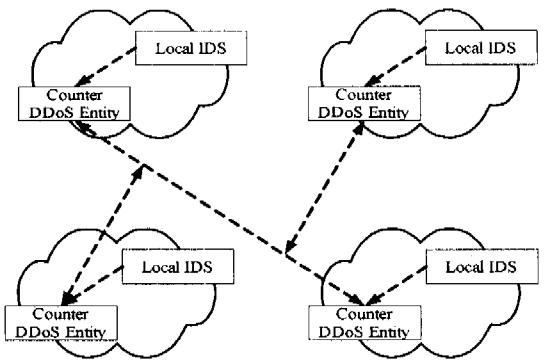

图 1　系统协作图

在系统的实现上,用了 Java 语言进行开发,并以 IBM 的 Aglets[4]系统作为移动 agent 的开发平台,使系统具有良好的跨平台性和扩展性。在每个网络域的控制服务器上设置管理控制台;在其余收检主机上构建一系列静态 agent、动态 agent 和相关机制,负责完成本地 IDS 和反 DDoS 实体的功能;并利用移动 agent 的安全通信机制协同抵御 DDoS 攻击。在消息传送格式上,将遵循 IDMEF 入侵检测消息交换格式的规则。IDMEF(Intrusion Detection Message Exchange Format)描述了表示入侵检测系统输出信息的数据模型。该数据模型用 XML 实现,并设计了 XML DTD(文档类型定义)。根据 IDMEF 的规则,系统中的消息分为两类:Heartbeat 和 Alerts。其中,Heartbeat 用于周期性的报告节点的运作情况并与系统取得联系,而 Alerts 则是用于通告各网络域检测并识别出的相关安全事件。

1.2　系统结构及运作

1.2.1　本地 IDS

本地 IDS[5]用于对子网的安全事件、入侵行为进行检测并将所检测到的有可能的 DDoS 攻击信息通告给反 DDoS 实体。本地 IDS 可分为用于对子网内部主机进行检测的静态 agent 和游走于子网内部对网络进行检测动态 agent。静态 agent 分布于子网内的各个主机上,多为基于主机的检测,主要对主机的网络适时连接以及系统审计日志进行智能分析和判断;动态 agent 则为基于网络的检测,对子网中的分组数据包进行攻击分析。静态 agent 和动态 agent 相互协作完成本地入侵检测,并将有可能的 DDoS 攻击通告给反 DDoS 实体。本地的 IDS 通告具有较高的优先级,以便在遭遇 DDoS 攻击时,系统作出及时的响应。

1.2.2　反 DDoS 实体的设计策略

反 DDoS 实体用于提供通信和响应的机制,协同各个网络区域共同完成对 DDoS 攻击的防御。它有以下几方面的任务:

①负责向其他实体传送本网络域的安全信息,并接收来自成员域的安全信息以及相邻网络域的实体状态信息。

②在本网络域内,实体运作于本地 IDS 之上。它借助于本地 IDS 的信息进行状态的迁移并推断正在发生的攻击。

③实体负责进行自动的 DDoS 攻击响应。在获得本地 IDS 的相关信息以后,对安全事件做出相关的评定,并针对 DDoS 攻击进行响应,如与本地网络组件相互联系,配置相关的过滤器等。

实体在运作过程中会表现为以下几种状态形式,并且各个状态都设有相关的阈值。图 2 为反 DDoS 实体状态迁移图。

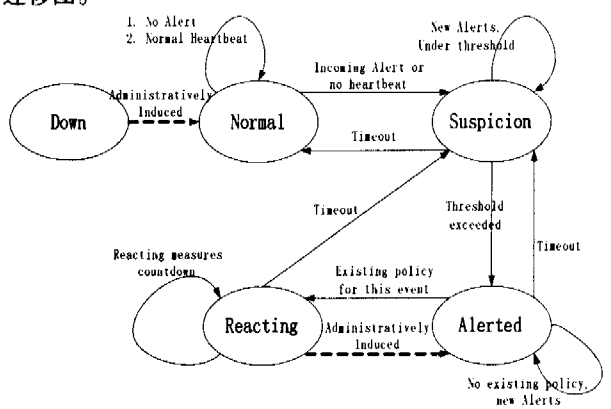

图 2　反 DDoS 实体状态迁移图

图中的各个状态分别为:

* Down:当系统没有任何操作或系统管理员中止了系统间的通信的时候,实体进入该状态。这是唯一的实体不传送 Heartbeat 的状态。

* Normal:当实体可以从其他的网络域的实体处接收到正常的 Heartbeat 并且没有接收到任何的 Alerts 信息时,实体处于 Normal 状态。

* Suspicion:当实体无法接收到来自其他实体的 Heartbeat 或是接收到 Alerts 的时候,实体将会进入 Suspicion 状态。Alerts 可以是来自其他实体的,也可以是本地 IDS 提供的。在实体中有专门的数据库存贮各种攻击信息,并有其专用的通报计数器。实体将检测每一个新来的 Alerts 是否与正在进行的事件匹配。一个确定的匹配对应通告计数器加 1。

本地的 IDS 信息对应反 DDoS 实体中的 Local_value 的值。在反 DDoS 实体的设计中,该值将导致与此 IDS 信息匹配的安全事件的通告计数器的成倍增加。也就是说,本地的信息相对于外来的 Alerts 具有更高的优先级。在成为 DDoS 的目标时,这一特征将加速实体的状态迁移并向其他的实体发送消息,以产生协同响应。

另一方面,无法接收到正常的 Heartbeat 并不会增加相关计数,在 Suspicious-timeout 的阶段中接收不到任何 Alerts,实体将返回 Normal 状态,并且相应的通告计数器清零。

* Alerted:当任何一类攻击的计数器大于 Alert_threshold 阈值时,实体将进入 Alerted 状态。当处于该状态时,系统控制器(management console)将会接收到相应的通告,同时在安全策略中查找与当前攻击特性相匹配的条目。如果没有相应的条目,这一状态将保持一段 Alert_timeout 时间。在这段时间中,如果又接收到类似的 Alerts 倒计时器将开始重新计数;接收不到类似的 Alerts,实体将回到 Suspicion 状态,并且相应的通告计数器设置为 Alert_threshold 的值。

* Reacting:如果有安全条目于所检测到的事件匹配,实体将进入 Reacting 状态。与 Alerted 状态相似,在 Reacting 状态时,也将有相应的 Reacting_timeout 值。在 Reacting_timeout 时间内,实体将对网络设备采取相应的安全防护措施。在此过程中,管理员可以选择是否进行相应的操作,也可以由实体自动响应。新的 Alerts 不会影响 Reacting_timeout 的倒计时。一旦时间到达,将回到 Suspicion 状态,并且计数器设为 Alert_threshold 值。

1.2.3 系统运作

在对目标域 DDoS 攻击的形成过程中,恶意的数据流将经过很多网络域。在经过这些网络域时,DDoS 都有可能被发现。如果未被发现,目标域由于其自身有限的带宽将会受到强烈的冲击。这将促使本地的实体发送 Alerts 警告,或是在通信中断的情况下,停止发送 Heartbeat 信息。

每一个收到 Alerts 信息(或是无法接收到 Heartbeat 信息)的实体,都会将这一信息与本地 IDS 的数据结合起来判断攻击流是否经过这一区域。接收到 Alerts 和 Heartbeat 都会使实体迁移到 Suspicion 和 Alerted 状态。越靠近 victim 的网络域的本地 IDS 的数据就越不规则也越会产生本地 IDS 的 Alerts 通告。因此,越靠近 victim 反 DDoS 实体就越有可能进入 Alerted 状态,根据相应的响应策略,实体自动地过滤恶意的数据流,以减轻攻击的影响。

1.3 软件结构

反 DDoS 实体主要由检测 agent、通信 agent、分析 agent 和响应 agent 组成,如图 3 所示。

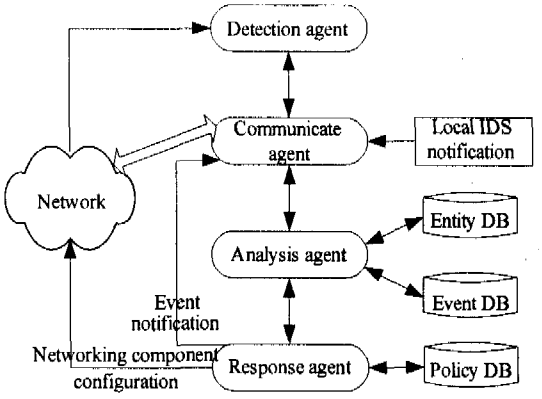

图 3 实体软件结构图

①检测 agent:游走于子网内部,负责监测子网内部的通信。当子网内部通信正常时,检测 agent 就向本网的通信 agent 发送周期性的 Heartbeat 信息。

②通信 agent:通信 agent 作为信使负责传输信息并对输入信息进行语法分析。通信 Agent 所接受到的信息是 Heartbeat 或是 Alerts,以及本地 IDS 检测到的有关 DDoS 攻击通告。一般说来,Heartbeat 是来自本网的检测 agent 和其他实体的通信 agent 的;而 Alerts 则是来自本网的分析 agent 和其他子网的。通信 agent 负责监测接收到的 Heartbeat 或是 Alerts 的有效性。如果与所希望的值不符,通信 agent 将向子网的管理控制台发送一个通告;如果符合,当前的 IDMEF XML 有效负载通过解析器。数据信息存储在子网预先设置的 Event Database 中,有待于分析 agent 和系统管理员做进一步的分析。Heartbeat 每隔一段时间就在数据库中更新一次。通信 agent 还要在规定的时间间隔内向其他子网的通信 agent 传送 Heartbeat 信息。同时,通信 agent 还要传送分析 agent 发来的 Alerts 信息。

③分析 agent:分析 agent 是反 DDoS 实体中最核心的部分,它负责综合分析本地 IDS 的信息和来自系统中其他子网的信息,并决定了实体状态的迁移。分析 agent 的运作和灵敏度由 Suspicious_timeout, Local_value, Alert_threshold, Alerted_timeout 和 Reacting_timeout 这几个值决定。

分析 agent 还负责初始化 Alerts,并将 Alerts 信息交付给通信 agent,由通信 agent 传送。为了避免 Alerts 的泛滥,分析 agent 根据以下几种情况产生 Alerts 信息:

* 实体必须处于 Alerted 状态才能发送一个 Alerts。
* Alerts 必须是有关于(D)DoS 攻击的。
* 实体进入 Alerted 状态仅仅是因为本地 IDS 的通告或者关于一个事件的本地通告在 Notification_min_time 内超过了 Notificaion_threshold 的值,乃至于实体还受到了外部的关于该事件的通告。本地通告的快速的增长表明网络区域处于攻击路径上的可能性越大。这两个配置参数不会影响实体本身的运作,但是可以控制 Alerts 的产生。

④响应 agent:响应 agent 在分析 agent 确定实体进入 Reacting 状态时被促发,负责将事件和已有的响应策略进行对比检查,如图 4 所示。

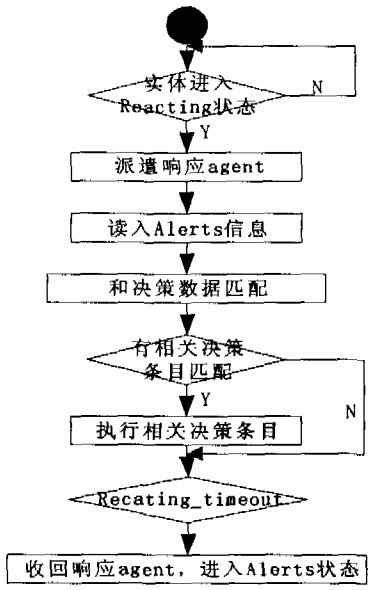

图 4　响应 agent 执行框图

一旦有特殊的事件使得实体进入 Reacting 状态,响应 Agent 就在决策数据库中搜索相关的条目并更新网络域的配置。根据事件的属性、攻击的形式和攻击的目标的匹配来执行相关的决策条目,如数据流的阻止、整形等。决策条目所对应的 action 的执行时间在实施中是由 Reacting_timeout 配置参量决定的。每当实体处于 Alerted 状态时,响应 agent 还要将信息传送给管理控制台,以便子网管理员做出及时响应。

2　结　论

系统所采用的移动 agent 和分布式协同机制为抵御 DDoS 攻击带来了很多优势:

(1)基于移动 agent 的分布式协同体系结构,减轻了网络的负载,提高了工作性能;

(2)系统采用的 Aglets 系统和 Java 语言实现,使得系统能够跨平台在异构的环境中工作;

(3)检测 agent 的可移动性,增强了系统的抵抗 DDoS 攻击的能力;

(4)Alerts 和 Heartbeat 的通信格式能够确保系统通信的有效性;

(5)在各个网络域在本地进行检测并做出响应,使系统能够最小化检测和响应的时间并不需要回溯机制。

当然,系统仍需要进一步的完善,以增进移动 agent 间通信的安全性,更好地抵御未来的 DDoS 攻击。

参考文献:

[1] 徐　恪,徐明伟,吴建平. 分布式拒绝服务攻击研究综述[J]. 小型微型计算机系统,2004,25(3):337－346.

[2] 严　毅,宁　葵,李陶深. 分布式拒绝服务攻击手段及其防范技术研究[J]. 微机发展,2004,14(9):81－83.

[3] Kolaczek G, Pieczynska－Kuchtiak A. A Mobile Agent Approach to Intrusion Detection in Network Systems [Z]. Springer－Verlag GmbH. Lecture Notes in Computer Science. /2005.842 － 849.

[4] Vigna G, Cassell B, Fayram D. An Intrusion Detection System for Aglets[Z]. Springer－ Verlag GmbH. Lecture Notes in Computer Science. 2002.64－77.

[5] 肖建华,张建忠. 基于移动 agent 的分布式入侵检测系统 MAIDS 的设计与实现[J]. 计算机工程与应用, 2003,17:164－165.

(上接第 229 页)

```
iGuard_read::iGuard_read()
{
  iGuard_TRACE ("iGuard_read::iGuard_read");
  data_block_ =0;
  stream_ =0;
  this→activate (THR_NEW_LWP); //开新的线程
}
```

这样,当在第一个连接正在传送文件的时候,假如又来了一个文件传输的请求,系统将开一个新的线程用来处理这个请求,因此多个传输过程可以同时进行[5]。

4　结束语

整个系统经过快一年的时间,终于开发成功。其中,对于 Server 端的通信平台也达到了预期的效果,利用 Sniffer 把接入的一个网卡设置成混杂模式,用 hub 组网,分析截获的数据包,均为密文,而且,在拥有近 200 台电脑的网吧进行压力测试,网吧电脑自由上网一个月后,服务器运行依然正常。

总之,在 Linux 环境下,利用 ACE 合理地搭建通信平台,能够做到安全、高效。

参考文献:

[1] 何　青. ACE 在开发健壮可靠的 C++ 系统中的应用研究[J]. 微机发展,2005,15(5):43－45.

[2] Schmidt D C, Huston S D. C++ 网络编程·卷 1:运用 ACE 和模式消除复杂性[M]. 於春景译. 武汉:华中科技大学出版社,2003.

[3] Buschmann F, Meunier R, Rohnert H, et al. Pattern－Oriented Software Architecture － A System of Patterns[M]. USA:Wiley and Sons,2003.

[4] Huston S D, Johnson J C E. ACE 程序员指南:网络与系统编程的实用设计模式[M]. 马维达译. 北京:清华大学出版社,2004.

[5] Schmidt D C. Acceptor and Connector: Design Patterns for Initializing Communication Services[A]. in Martin R, Buschmann F, Riehle D. Pattern Languages of Program Design [C]. Reading, MA: Addison－Wesley, 2002.