基于多 Agent 的动态层次化分布式入侵检测系统*)

吴 骏 王崇骏 王 珺 陈世福

(南京大学计算机软件新技术国家重点实验室 南京 210093)

(南京大学计算机科学与技术系 南京 210093)

摘 要 随着入侵检测技术的发展,IDS越来越呈现出分布性、智能性的特征。传统的基于多 Agent 的分布式入侵检测系统,往往采取一种分布式数据采集和层次化数据分析的方法。这虽然使系统的逻辑结构简单严谨,却很大程度上限制了系统的分布性、智能性与实时响应能力。本文提出一种动态建立的层次化结构,并提出了对其运行进行支持的基本方法体系;引入了对手思维状态模型,实现了建立在多 Agent 合作关系上的协同检测;对传统的分布式入侵检测系统进行了有效的改进。

关键词 多 Agent 系统,分布式入侵检测,通信,协作

Dynamic Hierarchical Distributed Intrusion Detection System Based on Multi-agent System

WU Jun WANG Chong-Jun WANG Jun CHEN Shi-Fu

(National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093)

(Department of Computer Science and Technology, Nanjing University, Nanjing 210093)

Abstract At present, distributed intrusion detection systems are taking on more and more characteristic of distributing and intelligence. Traditional MAS-based distributed IDSs Commonly do their detection work within a framework that employs the method of distributed data collection and hierarchical data analysis. This is a simple and strict method, but it restricts the characteristic of distributing, intelligence and real-time response badly. In this paper, we present adynamically-created hierarchical framework, then bring forward the basic algorithm system to support it. We introduce the opponent BDI model to our Agents, and realize the detection based on Multi-Agent cooperation, which effectively improves the traditional distributed intrusion detection.

Keywords MAS, Distributed intrusion detection, Communication, Cooperation

1 引言

随着人工智能技术、博弈技术的不断发展,入侵技术和入侵检测技术都得到不断的发展,高级入侵活动变得越来越呈现出分布性和协调性的特点,具体表现在:

(1)一次入侵可能分布在网络上的多个机器上[12]。

(2)一次攻击可能只是一个更大规模的入侵的一个部分,它的最终目标可能是攻击别的系统或非法得到其它的资源,而只是使用当前被攻陷的网络作为跳板[14]。

(3)多次简单攻击可以组合成为一次更复杂的长时间协同入侵[13]。

分布式入侵检测有两层含义,一种可以解释为分布式入侵的检测,强调入侵是分布式的,还有一种解释是分布式的入侵检测,侧重于检测是分布式的。这里,我们两种情况都要考虑,因为对于分布式协同攻击,最好的方法也是使用分布式协同检测技术。

随着对入侵检测系统(Intrusion Detection System,简称IDS)研究的深入,如今的IDS逐渐呈现出智能性、分布性的特点。在最近10年中,入侵检测系统正走向这样一种结构:它们由一组分布式的监测器构成,在这个结构中每个监测器都负责本地的检测并为全局检测提供信息,如:DIDS[8],GrIDS[9],EMERALD[10]和AAFID[1]等等。可以发现,它们实际上都是采取一种分布式数据采集和层次化的数据分析的方式来对网域进行监控。采用这种方式来构造分布式入侵检测系统,具有结构简单、系统的逻辑结构严谨的优点。但也有明显的不足,主要表现在两个方面:

(1)集中的分析构件承受的负载较高,可能会成为系统的瓶颈与单一的失败点;

(2)层次化的分析降低了系统的实时性。

本文主要论述对传统的层次化结构的分布式入侵检测系统进行改进。为提高系统的健壮性,提出一种动态建立的层次化结构,并且在各层次内部通过多 Agent 的通信与协作来完成本层分析工作;为提高系统实时性与检测精度,在各 Agent 内部对入侵对手建立思维状态模型,从而实现对入侵者的意图进行跟踪与预测。本文第2节简介系统的逻辑结构;第3节给出支持多 Agent 系统运行的若干方法;最后总结我们的研究工作,并且讨论未来研究开展的思路。

2 系统结构

设所监控的网域由一系列的局域网构成,则系统的逻辑

*)本文得到国家自然科学基金(项目编号:60503021)和江苏省自然科学基金(BK2005075)的资助。吴 骏 硕士生,研究方向是多 Agent 与计划识别;王崇骏 博士,研究方向是机器学习与分布式人工智能;王 珺 博士生,研究方向是人工智能与信息安全;陈世福 博士生导师,研究方向是人工智能。

结构如图1所示。整个系统由4个层次的Agent构成，自底向上分别是：跟踪Agent(Tracer Agent，简称TA)，基本Agent(Basic Agent，简称BA)，管理Agent(Supervisor Agent，简称SA)，监控Agent(Monitor Agent，简称MA)。各种Agent的结构特点与功能如表1所示。

表1 系统中各种Agent简介

	对手思维状态模型	产生方式	数量	主要能力与职责
TA	对从本机观察到的单个入侵对手建模	由BA产生	每台处于状态1下的主机上均有自然数N个(0<N<+∞)	跟踪某个IP的意图，当发现入侵意图时向所属BA报告
BA	对作用于本局域网内的潜在协同入侵者团队建模	系统初始化时产生	每台处于状态1的主机上均有且仅有一个	1)分析网络数据包并产生TA；2)与同一个局域网内的BA进行通信与协作，检测本局域网内的协同入侵；3)向本局域网的SA汇报检测到的入侵
SA	对作用于整个监控网段内的潜在协同入侵者团队建模	由一个BA转变而来	每个局域网内有且仅有1个	1)接收本局域网内所有BA的警报；2)与其他SA进行通信与协作，检测整个监控网段内的协同入侵；3)向MA汇报检测到的入侵
MA	无对手思维状态模型	由一个SA转变而来	整个系统内有且仅有1个	1)监控整个系统的运行状态；2)接收来自SA的报警；3)与用户界面进行交互

对于后3种Agent(BA、SA、MA)，我们统称其为分析Agent，因为它们担任着不同级别的分析工作。系统初始化时，在网域内的所有的主机上都将启动一个BA，然后每个局域网的BA之间进行选举，选出一个BA并且把它转化为SA来作为这个局域网的管理者。当系统中每个局域网都产生了自己的SA后，所有的SA之间再发起一次选举，选出一个SA并且把它转化为MA，也就是整个系统的全局管理者。MA与用户界面建立连接，系统的初始化完成。检测系统运行中，每个BA不断分析网络中的数据包，并且在自己所在的主机上产生一系列的TA来跟踪可疑用户的意图，至此建立了如图1所示的逻辑结构。

从以上描述可见，BA是系统中最基本的Agent，SA、MA均由BA通过某种合适的算法衍生而来，而TA由BA产生，用于跟踪某个用户的意图。在系统运行中的任一时刻，网域内任何一台主机H一定处于以下3种状态之一：

(1)状态1：H上运行着一个BA，以及有限个由该BA产生的TA。

(2)状态2：H上运行着一个SA。

(3)状态3：H上运行着一个MA。

整个系统实际上是一个互相合作的多Agent系统。为了能使这个多Agent系统能顺利地运行并且实现分布式入侵检测的功能，必须建立合适的方法体系来实现Agent之间的协调与合作，如下所述。

3 协调与合作

基于多Agent的分布式入侵检测本质上是一组基于共享心智的Teamwork。每个Agent通过彼此的通信协作完成共同的目标(入侵检测)。这包括：角色分配、协同检测、选举这3个方面。

3.1 角色分配

如果在一个局域网内，各个BA所能获得的网络数据完全相同。那么在这个局域网内的所有BA之间存在着角色分配的必要。即把所要检测的入侵任务互不重叠地分配到各个BA。目的是显而易见的，即避免BA之间的重复劳动，减轻BA所在主机的负载。

设在一个网段内已经启动的BA的集合为$S=\{BA_0, BA_1, \cdots, BA_n\}$，当网段又内一台主机启动后，在其上启动一个$BA_{n+1}$。$BA_{n+1}$与$S$中的BA进行以下通信：

Step1：对$\forall BA_i \in S, (i=0,1,\cdots,n)$，Connect($BA_{n+1}$, BA_i)。即对网段内所有的BA，BA_{n+1}与其建立连接。

Step2：BA_{n+1}向S广播一条JOIN消息。各个BA收到这个消息后，都向BA_{n+1}回复一条WELCOME消息。这条消息包括两部分，一是自己的类型二是自己承当的检测任务数。如：BA_i发送〈FTP server，12〉表示所在主机是一个个FTP server，它承当的检测任务数是12种。BA收到消息后得到BA_i所在主机是FTP server这条知识，并且对各个BA所承担的任务数排序。

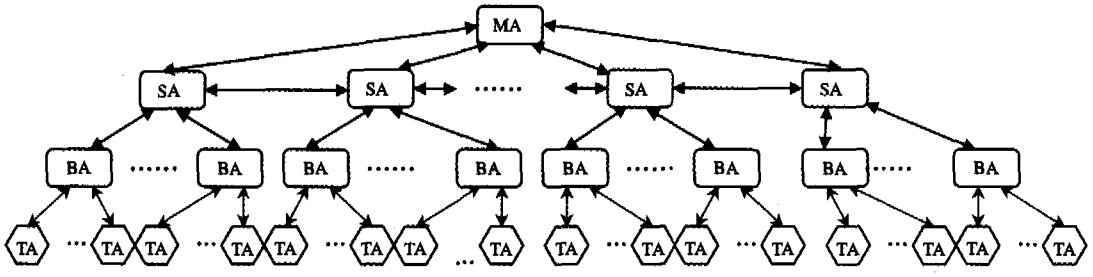


图1 多Agent系统的逻辑结构

Step3：计算每个BA所需要减少承担的任务数$k=[N/n]-[N/n+1]+1$。(其中[num]表示对num向下取整)。

Step4：BA_{n+1}向集合S中的每个BA发送一条REQUEST(k)，索要k种入侵的检测任务。

Step5：集合 S 中的各个 BA 收到 REQUEST(k)后，都向 BA_{n+1} 转让 k 种入侵的检测任务。

3.2 协同检测

对系统威胁较大的攻击都是由一系列相互联系的攻击行为完成的。现有的入侵检测系统都是输出大量低层的报警信息，不能表示报警的单个攻击行为之间的联系，使得重要攻击过程所引起的一系列相互关联的报警信息被混杂在大量的报警数据中，而无法对其采取及时的措施。我们把入侵行为分为原子入侵和复合入侵两类。原子入侵即为不能再分解为其他入侵的入侵行为。复合入侵由数个原子入侵按一定顺序构成。例如 SYN flood 攻击是一个原子入侵，而 mitnick[15] 攻击是一个复合入侵。

复合入侵往往是多个入侵者协同进行的，我们将一群互相合作的入侵者看作一个多 Agent 系统。于是检测协同入侵的问题就转化为对这个多 Agent 系统进行意图识别。要有效地解决这个问题，要依次进行两步工作：

(1)从大量的用户中发掘出作为对手的多 Agent 系统；

(2)对这个多 Agent 系统建立对手思维状态模型从而对其进行意图识别。

在我们的系统中，TA 对某个 IP 建立对手思维状态模型，通过不断分析其行为推测出它们的入侵意图。进一步 BA，SA 对作为对手的多 Agent 系统建立对手思维状态模型，通过对大量的原子警报进行分析，分别在局域网层次上和在整个监控网络的层次上发现各种协同入侵意图。也就是说协同检测体现在三个方面，分别是：TA-BA 协同；BA-BA 协同；以及 SA-SA 的协同。

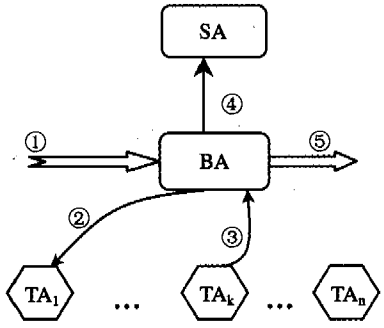
设 BA_i 所得到的检测任务为 $I=\{a_1,a_2,\cdots,a_m,b_1,b_2,\cdots,b_n\}$。其中 $a_1,a_2,\cdots,a_m$ 为原子入侵，$b_1,b_2,\cdots\cdots,b_n$ 为复合入侵。系统中存在的 3 种协同检测如下所述。

3.2.1 TA-BA 协同

TA-BA 协同的目的在于从大量的用户中逐个发现潜在的恶意用户。协同的过程如图 2 所示。在 BA 中存在着一个数据采集单元和一个异常检测单元。设观察窗口为 $w(p,\Delta)$。其中 p 当前事件的指针；Δ是一个正整数。则 $w(p,\Delta)$ 表示以当前事件为结尾的 Δ 个历史事件序列。随着窗口的移动。BA 不断从 $w(p,\Delta)$ 中提取出不同 IP 所进行的历史事件序列，并且把它们表示为：

$H=\{\langle IP_1;h_{11},h_{21},\cdots,h_{n1}\rangle,\langle IP_2;h_{12},h_{22},\cdots,h_{m2}\rangle,\cdots,\langle IP_k;h_{1k},h_{2k},\cdots,h_{ik}\rangle\}$的形式，这就是各个用户的历史事件信息。

H 为 BA 和它产生的 TA 所共享，是它们进行检测活动的基本依据。BA 通过对各个 IP 的历史行为进行异常检测(即把某个 IP 的历史行为去与预先建立的正常行为模式进行比较)，如果发现某个 IP 的行为异常，则说明这个用户可能是一个潜在的入侵者。为了进一步确定这个用户的意图，BA 产生一个 TA 来对该用户的行为进行跟踪和分析。在这个 TA 中存在着一个该用户的思维状态模型，并且根据历史信息对其进行初始化。TA 为了实现它的目标(即确定用户的意图)，不断地收集该用户的行为，然后把它们分解为对手思维状态模型中的各个成分，即对手的信念(B)、愿望(D)、意图(I)，并且利用 BDI 之间的内在约束关系消除其内在矛盾。最终得到用户可能意图的排序和其概率。当某种可能意图的概率超过一个阈值时，TA 向 BA 发出报警。此时 TA 实现了它的目标，它释放自己所占用的资源并退出。而 BA 获得了 TA 的检测结果后，一方面向 SA 汇报，以为局域网间的协同检测提供数据；另一方面，向需要这种入侵消息的本局域网内的其他 TA 汇报，目的是实现本局域网内的 BA 之间协同检测。



① BA 不断分析网络数据，并作异常检测。

② 若发现某个 IP 的行为异常则生成一个 TA 来对其进行跟踪。

③ 某时刻某个 TA 发现它所跟踪的用户有某种入侵意图，则向 BA 汇报。

④ BA 得知 TA 汇报的入侵，立即向 SA 汇报。

⑤ 向同一个局域网中预定了这种入侵消息的其他 BA 汇报。

图 2 TA 与 BA 的协同方式

下面以一个实例来说明这个过程：

设用户 A 和用户 B 同时对网段内的一台主机发动 ping of death 攻击。那么随着 $w(p,\Delta)$ 的移动，$w(p,\Delta)$ 中的事件“ping”会不断增多，BA 从各个用户的历史事件信息中先后发现用户 A 和用户 B 的行为异常。并先后产生两个 TA 来分别对其进行跟踪。此后这两个 TA 逐渐发现并确定 A 和 B 有 ping of death 的攻击意图，并且向 BA 发出报警。

3.2.2 BA-BA 协同

对于复合攻击，需要 BA 之间的协作来实现检测。设一个 BA_i 所分配的复合入侵检测任务为 $B=\{b_1,b_2,\cdots,b_n\}$。则 BA 之间通过图 3 所示的方式进行协作。BA 所承当的分析任务是发现作用于本局域网范围内的协同攻击，而它们分析的依据是来自 TA 以及其他 BA 的原子入侵报警信息。对于收到的报警信息集合，BA 对其进行关联分析。找出各个入侵者所有可能的关联方式，即所有可能的入侵者组合的集合。然后 BA 把它们看作一个个对手多 Agent 系统，并且分别对其建立对手思维状态模型。在以后的时间中通过观察各个对手的行为，逐渐识别出各个对手的意图。一旦识别出某个对手的意图是某种协同攻击，则向 SA 发出报警。

下面以经典的 Mitnick 攻击为例来说明 BA 之间的协同。Mitnick 攻击由拒绝服务攻击，TCP 序列号猜测与 IP 欺骗三种攻击组成。整个攻击过程是通过 A 和 B 两台机器进行的。其中 B 是目标机，而 A 是网络中 B 所信任的一台机器。具体攻击过程如下：

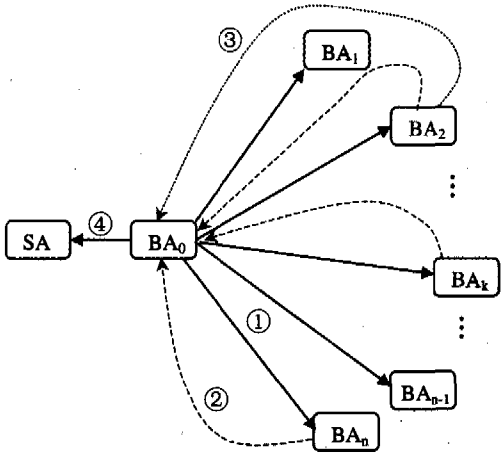
Step1：攻击者对 A 发起一个 SYN flood 攻击，以剥夺 A 对 B 进行响应的能力。当然也可以是其他的拒绝服务攻击方式。

Step2：攻击者向 B 发送多个 TCP 包，借此预测出主机 B 产生的 TCP 序列号的值。

Step3：通过冒充主机 A 的 IP 地址，攻击者伪装作 A，它向主机 B 发送一个 SYN 包以在 A 和 B 之间建立一个 TCP 连接。

Step4：主机 B 向 A 回复一个 SYN/ACK。攻击者看不到这个包。而此时主机 A 由于已经受到 SYN flood 攻击而不能向主机 B 发送一个 RST 消息。

Step5：使用计算出来的 B 的 TCP 序列号(注意攻击者没有看到主机 B 发送给主机 A 的 SYN/ACK 消息)攻击者发送一个包含预测出来的 TCP 序列号的 ACK 包给 B，以响应 B 发出的 SYN/ACK 包。



① BA_0 对于所有对于 $\forall b_i \in B$，查攻击本体论库，得知它由 $A=\{a_1,a_2,...,a_n\}$ 这 n 种原子攻击按一定顺序构成，于是向其他所有 BA 广播一条订阅消息，订阅这些入侵的检测结果。

② 承担这些入侵检测任务的其他 BA 向 BA_0 回复一条确认消息。

③ 某时刻一个 BA 检测到了 BA_0 订阅的入侵消息，则向 BA_0 汇报。

④ 对收到的原子入侵消息(包括 TA 汇报的和从其他 BA 汇报的)进行关联分析，对可疑的用户集合建立对手思维状态模型，逐渐推测出其可能的意图。如果确定对手有入侵意图，则向 SA 发出报警。

图 3　BA 之间的协同方式

Step6：此时主机 B 确信是和信任的主机 A 建立了一个 TCP 会话。而攻击者已经和攻击目标建立了一个会话，它可以向 B 发送命令。攻击完成。

设一群协同的入侵者 $S=\{A_1,A_2,\cdots,A_n\}$ 联合对监控网络中的某个局域网中的两台主机发动此攻击。而 S 的三个子集 S_1,S_2,S_3 分别负责完成拒绝服务攻击，*TCP* 序列号猜测与 *IP* 欺骗。其中对于此入侵，局域网中存在一个负责检测它的 BA_i。而 BA_j,BA_k,BA_l 分别负责检测拒绝服务攻击，TCP 序列号猜测与 IP 欺骗。实际上，当 BA_j,BA_k,BA_l 先后向 BA_i 汇报自己的检测结果的过程中，BA_i 就已经分析出了 S 中各个攻击者的合作关系，并对其建立对手思维状态模型。此后在 BA_j,BA_k,BA_l 的不断汇报中逐渐识别出集合 S 的 Mitnick 攻击意图。

3.2.3　SA- SA 协同

SA 之间的协同与 BA 之间的协同类似。其目的是发现跨越局域网段的协同攻击。比如一个用户以一个局域网里的一台主机为跳板，向另一个局域网里的一台主机发起 mitnick 攻击。

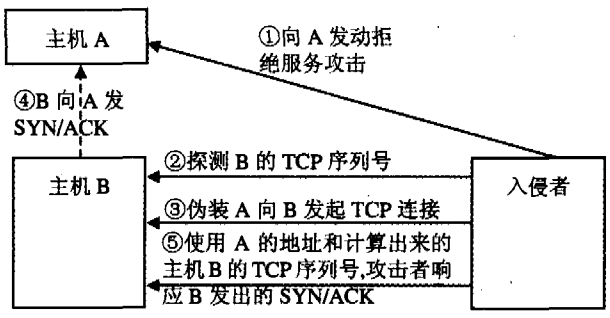


图 4　Mitnick 攻击的过程

3.3　选举

系统初始化时，以及当一个 BA 发现它所在网段内的 SA 崩溃或者退出时，或者当一个 SA 发现 MA 崩溃或者退出时，都要进行选举。选举的目的为选出一个负载最轻的 BA 作为 SA，或者选举出一个负载最轻的 SA 作为 MA。负载的大小可用 CPU 平均使用率和内存使用率等标准来衡量。SA 的选举较为简单，选举使用经典的 Bully 算法进行，步骤如下：

Step1：向所有的其他 BA 发送一个 ELECTION(X)消息，其中 X 为它的负载。

Step2：网段内所有其他 BA 若发现自己的负载小于 X，则向发起选举的 BA 发送一条 OK 消息，否则退出选举。

Step3：若发起选举的 BA 没有收到 OK 消息，则它成为新的 SA。选举结束否则退出选举，转到 step4。

Step4：剩余 BA 继续按 step1—3 的顺序执行。

对于 MA 的选举，则不仅要利用上述方法从 SA 中选出 MA，而且还要在 SA 被选中的局域网中从 BA 中选出新的 SA 来代替原来成为 MA 的 SA。

结束语　本文改进了传统的基于多 Agent 系统的分布式入侵检测模型，提出了一种动态建立的层次化结构，并提出了对其运行进行支持的基本方法体系。动态建立的逻辑结构，提高了系统的健壮性；对手思维状态模型的引入，使系统能提早发现各种入侵意图，在一定程度上提高了系统的智能性和实时性。而各层次 Agent 之间的协同检测，则提高了分布性，降低了较高层次 Agent 的负载。为了使系统能检测到各种新新型的入侵，将在系统中引入 Agent 学习的功能，使系统能够发掘和记忆各种未知的入侵模式，这将进一步提高系统的实用性。

参 考 文 献

1　Balasubramaniyan J S, Farcia-Fernandez J O, Isacoff D, et al. An Architecture for Intrusion Detection Using Autonomous Agents: [Technical report]. 98/05. Purdue University, 1998

2　Shajari M, Ghorbani A A. Application of Belief-Desire-Intention agents in intrusion detection and response. In: Proceedings of Privac, Securit, Trust (ST04) Conference, Fredericto, New Brunswick, October, 2004. 181～191

3　Boudaoud K, Foukia N, Gessoun Z. An Intelligent Agent Approach for Security Management. In: HPOVUA'2000, June 2000

4　Gopalakrishna R. A framework for distributed intrusion detection using interest driven cooperating agents. In: Paper for Qualifier II examination, Department of Computer Sciences, Purdue University, May 2001

5　Gorodetsky V, Karsaev O, Samoilov V, et al. Asynchronous Alert Correlation in Multi-agent Intrusion Detection Systems. In: MMM-ACNS, 2005. 366～379

6　Liu Hung-Hao, Soo Von-Wun. Intrusion Detection Using Alert Correlation Methods Based on Multi-agent Systems. TAAI, 2004

7 Labrou Y, Finin T. A proposal for a new KQML specification. In: ARPA Knowledge Sharing Initiative[R]. External Interfaces WorkingGroupPaper. February1997
8 Snapp S, Brentano J, Dias G, et al. DIDS (Distributed Intrusion Detection System)-motivation, architecture, and an early prototype. In: Proceedings of the 14th National Computer Security Conference, October 1991
9 Staniford-Chen S, Cheung S, Crawford R, et al. GrIDS-a graph based intrusion detection system for large networks. In: Proceedings of the 19th National Information Systems Security Conference, September 1996
10 Porras P A, Neumann P G. EMERALD: event monitoring enabling responses to anomalous live disturbances. In: 1997 National Information Systems Security Conference, Oct. 1997
11 李毅,石纯一. 基于 BDI 的对手 BDI 模型[J]. 软件学报,2002,13(4):644~648
12 Northcutt S. Network Intrusion Detection: An Analyst's Handbook. New Riders,1999
13 Tseng C Y, Balasubramanyam P, Ko C, et al. A specification-based intrusion detection system for AODV. In: Proc. of the 2003 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03). Farirfax Virginia,2003. 125~134
14 Vigna G, Kemmerer R. NetSTAT: A network-based intrusion detection system. In: Proc. of the 14th Annual Computer Security Applications Conference. Scottsdale, U S A, Dec. 1998
15 Undercoffer J, Joshi A, Pinkston J. Modeling Computer Attacks: An Ontology for Intrusion Detection. In: RAID 2003, LNCS 2820,2003. 113~135

(上接第 40 页)

发模式即广度优先遍历网格来实现。下面是一种通过 BFST 来获得网络拓扑的算法:

```
if(当前节点地址≠封装体的源地址){
  if(节点深度>0)
    {  count=该节点的相邻节点的个数;
       neighbor 是相邻节点网络地址的数组;
       向该节点中所有相邻且未被访问过的节点发送
       该封装体并设置其访问标记;}
    else
    {  count=0;//该节点是叶子}
     向管理节点返回报文(节点地址、父节点地址、邻节
     点个数及地址);}
else if(当前节点地址≠目的地址){
       向目的节点转发报文;}
```

如果单从封装体的转发例程逻辑上看,上述的拓扑发现是由一种广度优先搜索来实现的,但是整个拓扑发现过程是分布在每个节点上进行的,因此它并不是严格的广度优先搜索。

在生成了拓扑结构后,系统还应以最小代价维护网络最新的拓扑结构,这种功能可以用转发模式为 DFST 的封装体来实现,其转发例程基本上与拓扑发现的例程相似。当封装体到达一个节点后,它首先检查该节点的所有相邻节点是否全部已记录在节点的访问状态中。如果没有,则选择一个未记录的节点并向它转发报文。DFST 可以用最小网络带宽的情况来实现这一功能,但是它的遍历网络速度较慢。

5 主动网络管理系统中的流量分析

NMS(Network Management System)的流量是指 NMS 在网络层与所有节点交换的数据量。如果在应用层传送的数据量为 X,则在网络上传送的数据量就为 $\lambda=\xi(X)+\psi(X)X$,式中 $\xi(X)$表示在面向连接方式中建立连接时交换的控制信息,$\psi(X)$由数据包的封装格式决定。为了简单起见,数据量可简化为 $\lambda=k(X)X=kX$,称 k 为加权函数。在后面讨论中,用 M 表示被管理的网络设备的数量,用 Q 表示该任务需要对 MIB 查询的次数。

(1)基于 SNMP 的网络管理系统流量:在 SNMP 系统中,当 NMS 需要请求节点完成一项任务时,NMS 就向节点上的 SNMP 代理发送请求。假定第 i 条请求消息的大小为 Get_i,为了完成该请求任务,这 Q 条请求消息必须全部发送到所有 M 个被管理设备节点上。设备 m 收到第 i 个请求时,会做出响应,向 NMS 发送大小为 R_{im} 的响应消息。因此 NMS 的流量 T_{nms} 就可以表示为

$$T_{nms}=\sum_{m=1}^{M}\sum_{i=1}^{Q}(k_{nms}\mathrm{Get}_i+\tilde{k}_{nms}\tilde{R}_{im})$$

其中 k_{nms} 表示对请求信息 Get_i 封装时的加权函数,$\tilde{k}_{nms}$ 表示对响应信息 $\tilde{R}_{im}$ 封装时的加权函数。

(2)基于 ANMS 的网络管理系统流量:ANMS 向第 1 个节点发送一个主动包,这个包依次访问每一个节点,并在本地得到响应消息,然后带着响应消息传送到下一个节点,最后将所有响应消息传送到 ANMS。我们用 CAN 表示主动包的初始大小,则 ANMS 的流量 T_{AN} 可以表示为

$$T_{AN}=k_{AN}C_{AN}+\tilde{k}_{AN}\sum_{m=1}^{M}\sum_{i=1}^{Q}\tilde{R}_{im}$$

前一部分表示 ANMS 向第 1 个节点发送一个包,后一部分表示最后一个节点上的包将所有的响应消息发送到 ANMS。

通过分析和数据推导(过程略去),可以证明:$T_{nms}\geqslant T_{AN}$。随着网络所管理的节点数的增加,SNMP 的 NMS 流量大于 ANMS 结构的流量。

结束语 在建立原型系统时,我们采用链路层连通模式在局域网中搭建主动网络环境。主动节点的实现采用的是 ANTS 扩展环境—EANTS 以及 Janos,操作系统选用 Linux。非主动节点运行 Windows 作为操作系统。EANTS 系统是用 Java 编写的,所以,在 EANTS 上开发的基于主动网络技术的管理系统也采用 Java 编写。

主动网络管理体现了主动网络的思想,将一部分网络管理功能动态地分布在主动节点上,充分利用了主动节点的计算能力,使节点能够自动发现、解决问题,从而极大地优化了网络管理。本文讨论了一种主动网络的管理模型,该模型中各个模块相互独立、任务明确,而且在每个层都可以动态更新以适应主动网络中主动节点的易变性和主动应用的扩展性,因此网络管理的稳定性和扩展性都大为提高,适应了现代网络管理的需要。

参 考 文 献

1 Kawamura R, Stadler R. Active Distributed Management for IP Networks[J]. IEEE Communications Magazine, 2000, 38(4): 114~121
2 Shaer A E. Active Management Framework for Distributed Multimedia Systems [J]. Journal of Networks and Systems Management, 2000,16(8):49~54
3 Marshall I W, et al. Active management of multiservice networks. In: Proc. IEEE NOMS, 2000. 981~983
4 Shaer A E. Active Management Framework for Distributed Multimedia Systems [J]. Journal of Networks and Systems Management, 2000, 8:49~72
5 Brunner M, Stadler R. Service Management in Multi-Party Active Networks [J]. IEEE Communications Magazine, 2000, 38(3):281~286
6 Kiwior D, Zabele S. Active Resource allocation in Active Networks [J]. IEEE JSAC, 2000,19(3): 452~459