MANET 分簇 IDS 报文丢弃攻击全局感知方案

靳倩慧,姚国祥,李 佩
(暨南大学 信息科学技术学院,广东 广州 510632)

摘 要:在分析现有报文丢弃攻击检测算法的基础上,提出了一种基于簇首协作的报文丢弃攻击全局感知方案,利用 IDS 簇首协同监视节点报文收发状态,改进现有算法的监测方式和节点状态判定算法。仿真结果表明,该算法具有良好的检测率和误检率,在规避网络中的恶意节点以及维护网络正常吞吐量等方面具有较好的性能。

关键词:移动自组织网络;丢弃攻击;恶意节点检测;入侵检测系统
中图分类号:TP393 **文献标识码**:A **文章编号**:1674-7720(2011)17-0049-05

A scheme of globally detecting packet dropping attacks for clustering IDS in MANET

Jin Qianhui, Yao Guoxiang, Li Pei
(College of Information Science and Technology, Jinan University, Guangzhou 510632, China)

Abstract: This paper presents a scheme of globally detecting packet dropping attacks for clustering IDS in MANET based on cluster heads in collaboration after analyzing the existing packet dropping attack detection algorithms. The solution monitors nodes' communication status by IDS cluster heads in collaboration, and improves the monitor methods and judging ways of nodes' status. Simulation results show that this algorithm has high malicious node detection rate and low false alarm rate, and it performs well in evading malicious nodes and sustaining throughput capacity.

Key words: MANET; packet dropping attacks; malicious node detection; IDS

移动自组织网络 MANET (Mobile Ad Hoc Networks)由众多节点以自组织的方式组成,网络中不存在中心控制节点,较远距离节点间以多跳的方式进行通信。然而,在大规模无线自组织网络中,部分节点为了保护自身较少的资源,可能不转发其他节点请求转发的数据包,产生自私节点问题[1],特别场景下(如战争环境)合法节点被攻陷后将成为恶意节点。

在数据报文传输过程中,自私节点和恶意节点可以故意随机丢弃部分需要自身转发的数据报文来对网络通信实施破坏,这就是内部节点丢弃报文攻击,它导致网络吞吐率下降、报文重传率增高,严重时会造成网络报文传输无法进行。然而,由于无线自组织网络中节点性质的不确定性,现有的适用于 Ad Hoc 网络的路由协议无法应对这种在网络层发起的攻击。报文丢弃攻击难以检测和防范,严重影响到了无线自组织网络的通信性能和实际应用。

目前,针对报文丢弃攻击的研究多数集中在基于邻居节点监测方法[2]对现有路由协议(如 DSR)的改进,希望构造一个安全的路由协议或模型来抵抗攻击。然而,邻居节点监测算法本身存在一些局限性,此类方案增加了 MANET 网络路由协议的复杂度,监测效果却不理想。

本文在分析现有解决方案的基础上,提出了一种基于簇首协作的报文丢弃攻击全局感知算法,在 MANET 入侵检测系统中利用分簇 IDS 的簇首节点协作进行全局监测。该方案将安全检测从路由协议中独立出来,由 IDS 来实现报文丢弃攻击的检测,简化了 Ad Hoc 路由协议的设计,同时弥补了邻居节点监测算法的不足,在报文丢弃攻击的检测率、误报率和检测响应速率方面有较好的性能。

1 邻居节点检测算法分析

1.1 报文丢弃攻击相关研究

为对抗内部节点在网络层发起的报文丢弃攻击,MARTI S 等人提出了 Watchdog 算法[3],节点在混杂模式下

工作，当节点把报文转发给下一跳节点后，利用无线信号暴露在空中的特性来监听下一跳节点是否继续转发该报文。LEE S和CHOI Y采用了类似Watchdog的邻居节点监测系统(NWS)来获取邻居节点的报文状况，但是局部管理的方法增加了节点能量耗费，降低了网络带宽使用效率。参考文献[4]中提出两种不合作的节点形式，并提出用CCS中央控制系统给每个节点分配信用值，如果节点A为节点B转发了到目的节点D数据报，则D要给A一些信用币，哪个节点用完了信用，可以向CCS要求补充。由于每次都是目的节点支付信用币，这就为DOS攻击提供了方便。参考文献[5]希望利用有限自动机构造自组网的入侵检测算法，此方案能够监测节点发送的报文，但在如何监测节点收到的报文方面没有给出有效的解决方法。

1.2 邻居节点监测算法介绍

上述研究大多以邻居节点监测算法为基础对节点的收发报文情况进行监听，从而达到判断入侵节点的目的。当一个节点转发数据报时，邻居监测系统会确认路由的下一跳节点是否转发了该报文。邻居监测系统监听下一跳节点采取的行为，并依此判断下一跳节点是否有恶意行为。

如图1所示，假设S到D有通路，中间节点为A、B、C，节点A不能直接和C通信，但A可以监听到B发送的报文。A可以辨认B是否转发了报文。A缓存下刚发送的数据报，和监听到的B转发的数据报比较，看是否一致。如果一致，表明此数据报已经发送，A移除缓存的数据报，这一过程结束；如果不一致，表明B有篡改信息的恶意行为；如果数据报在缓存区保留的时间超过一个门限，则认为B没有正常工作。对于后两种情况，A会给这个没有尽责的节点记过。如果B被记过的次数超过一个门限值，A确定此节点有恶意行为，它向源节点S发送路由出错报文(RRER)通知B是恶意节点。在从A回到S节点过程中，路由到的节点都会记录下B是恶意节点，在以后的路由选择过程中会将B节点从路由链路中排除。

图1 A监听B正常转发数据报给C

1.3 邻居节点监测算法的缺陷

邻居节点算法利用无线网络信号暴露在空间中而且信号多向传播的特性来实现对邻居节点转发数据报的监测，然而这种方法却存在以下的局限性：

(1)邻居监测系统需要每个节点监督其邻居节点，并缓存有用的状态信息，要求节点的存储和计算功能得到加强。而且由于每个节点均被其所有邻居节点监视，导致监测信息存在大量冗余，浪费了节点电池能量和计算资源。

(2) 节点A只能孤立地监测其邻居节点的转发报文状态，并根据自身的监测记录判断邻居节点B是否有恶意丢包行为，不能综合B的其他邻居节点(如C)对节点B的监测结果。因此，这种判断带有很大的片面性，容易得出错误的监测结论。

(3) 恶意节点可以把无辜节点作为恶意节点放在RRER数据报中，发送给源节点S，比如，节点A可以向S打虚假报告B没有转发报文，使S认为B是恶意节点，这个问题的发现就需要依赖消息正常发送到D，节点D反馈消息给S，则S会认为路由是正常的。如果在消息给B传递后未能到达目的节点D，或者D的反馈消息没有正常到达A，则节点B就会被当作恶意节点，从而造成误报，本文称这种攻击为虚假丢报文丢弃攻击。

2 报文丢弃攻击全局感知方案

2.1 分簇IDS

移动自组网的组网特点决定了相应的入侵检测系统必须要采取分簇架构。分簇结构IDS典型层次如图2所示，整个MANET系统以簇为单位分成多IDS簇，每个簇选出簇头，IDS功能模块按需要置于簇头或簇成员节点上。

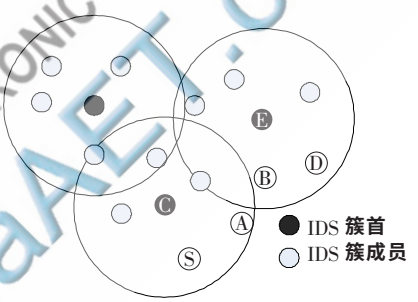

图2 分簇IDS结构

在邻居节点检测算法中，每个节点都监测其邻居节点的行为，产生大量冗余监测信息，耗费了节点资源。本文提出基于簇首的监测模式，即整个网络划分为簇，每个区域选出一个簇头作为监视节点，负责整个区域的入侵检测。该簇头收集整个区域内节点的行为信息，并按检测算法进行分析，确定入侵行为。IDS簇首节点周期性地广播告示报文，以维持簇首监测节点的地位。簇首节点服务时间到了后，就重新启动一个新的选举过程。为了保证公平和随机性，防止恶意节点一直占据簇首位置，发动虚假报文丢弃攻击，IDS分簇算法要求上一分簇周期的簇首节点将不能参加下一周期簇首节点的选举，除非整个区域只有一个节点存在。

2.2 簇首节点协作实现监测节点报文收发状态

不同于传统有线网络中的IDS，MANET中的簇首节点并不能像在有线网络中那样监测到节点的报文收发状态。由于无线网络的传播特性，分簇结构IDS中的簇首节点只能监测到其簇成员节点发送的报文，而对于簇成员接收到的数据包只能部分监测到，因此，在应对报文丢弃攻击之类的被动攻击时显得无能为力。图2

中，节点C、E为簇首，节点A、S为C的簇成员，节点B、D为E的簇成员。簇首节点E只能监测到节点B发送了数据包，如果节点A转发给B一个数据包，并要求B转发给D，如果节点B此时发动报文丢弃攻击，这时，簇首节点E并不能监测到B接收了这样一个数据包，因此，并不能发现B丢弃了应转发给节点D的数据包。

为了解决这个缺陷，本文提出了基于簇首协作监测的方案，利用各簇首节点间相互通信协作检测的方法实现对报文丢弃攻击的检测。图2中，虽然节点E不能够监测到B应该转发一个数据包，但节点A在给B转发数据包时，节点A的簇首节点C能够监测到节点A给节点B转发了数据包，并通过查询数据包中下一跳路由信息得知B接收了这个数据报并应该转发该数据包给D，如果E和C进行通信，交换彼此监测信息，节点B的簇首节点E就能够借助和C交换的消息来全面监测节点B应发的数据报和已发数据报状态。下面利用这一方法给出详细解决方案。

2.3 基于簇首协作的报文丢弃攻击全局感知方案

首先，为了监测节点收发报文状态，各IDS簇首节点需要为网络中所有节点维护一个数据结构，记录各节点收到报文的数量和发送报文的数量。数据结构定义伪码表示如下：

typedef struct {Long Receive; Long Send;} Record;

其中，Record表示此数据结构，下文简记为Recd；Send表示监听到节点转发一个数据报文；Receive表示根据监听到的报文中的路由信息，该数据报文下一跳节点应该转发此数据报文，其值为负，下文简记为Rece。

假设S到D有通路，中间节点为A、B，此时节点S转发数据报文到A，因为节点S处于簇首C的簇内，C能够监听到节点S转发了一个数据报文，同时，从该数据报文路由信处表中查得一下跳地址为节点A，则执行操作S.Recd.Send+1和A.Recd.Rece-1，表示节点A需要转发一个数据报文。如果节点A也正常转发了该数据报文，其簇首节点C会监听到该数据报文，执行操作A.Recd.Send+1和B.Recd.Rece-1。节点B并不在簇首C的监控范围内，但如果节点B正常转发此数据报文到节点D，则其簇首节点E会监听到该数据报文并执行操作B.Recd.Send+1和C.Recd.Rece-1。

在MANET网络中，各簇首节点在监测周期结束时多播Hello消息，彼此交换监测的簇内节点报文转发状态表，其中，服务周期的长短由系统根据需要设定。通过检测算法就可以对网络中是否存在报文攻击及恶意节点号做出检测。

2.4 报文丢弃攻击全局感知算法详细实现

设网络中共有m个IDS簇头，分别记为H_1、H_2、…、H_m。则经过一个Hello周期交换后，节点i应发送数据包数为$-\sum_{k=1}^{m}X_k.I.Recd.Rece$，已发送的报文总数为$-\sum_{k=1}^{m}X_k.I.Recd.Rece$。则节点$i$的报文转发率可以表示为：

$$P_i=\frac{\sum_{k=1}^{m}X_k.I.Recd.Rece+\sum_{k=1}^{m}X_k.I.Recd.Send}{-\sum_{k=1}^{m}X_k.I.Recd.Rece} \tag{1}$$

记节点i在一个Hello周期内加入和退出簇的次数为Ch_i，其值可以在簇首节点中得到，换簇次数越多，系统记录丢包率也就越大；节点移动速度M_i，其值在簇生成算法中可以获得，移动越快，丢包率就越高；此外，节点的吞吐量也会对丢包率产生影响，单位时间内到达的包越多，就越容易产生碰撞，碰撞会使簇首节点监听不到节点转发数据报。由于文中节点应发报文数量用负值表示，因此节点吞吐量可表示为：

$$Thp_i=\sum_{k=1}^{m}X_k.I.Recd.Send-\sum_{k=1}^{m}X_k.I.Recd.Rece \tag{2}$$

本文将这种由于节点移动速率、吞吐量及出入簇次数对簇首节点监测节点丢包率造成的影响定义为平衡因子μ。本模型采用的平衡因子函数为：

$$\mu_i=1/\ln(M_iThp_iCh_i/\lambda+e) \tag{3}$$

其中λ为系统归一化系数，对$M_iThp_iCh_i$作归一化处理，其值由$M_iThp_iCh_i$值最大的节点决定。μ随节点$M_iThp_iCh_i$的变化情况如图3所示。

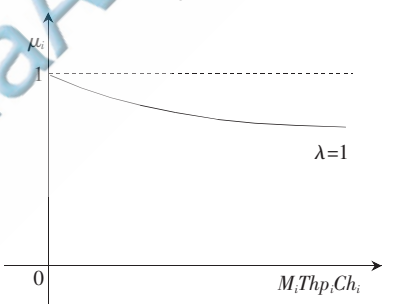

图3 μ随节点$M_iThp_iCh_i$变化情况

定义了平衡因子，就可以对节点的收发报文状态$Status_i$做一个比较精确的描述：

$$Status_i=P_x/\mu_i \tag{4}$$

2.5 恶意丢弃报文节点判定

本方案采用狄克逊准则对恶意节点进行判定。狄克逊准则是通过极差比判定和剔除异常数据。与一般比较简单的极差方法不同，该准则为了提高判断效率，对不同的实验量测定数应用不同的极差比进行计算。该准则认为异常数据应该是最大数据和最小数据，因此其基本方法是将数据按大小排队，检验最大数据和最小数据是否是异常数据。

依式(4)分别计算各节点的$Status$值，从小到大排列为$Status_{(1)}$，$\leqslant Status_{(2)}$，……$\leqslant Status_{(n)}$，因为丢包率大的节点才有可能是恶意节点，所以这里舍弃最小值，仅取最

大值，应用狄克逊准则进行节点类型进行判定。如果 $f_0 > f_{(n,\alpha)}$ 就可以判定该值所对应节点为可疑节点。$f_{(n,\alpha)}$ 可以查表获得，其中 n 是节点个数，α 为检验水平，根据检测精度需要可以取 0.01 或是 0.05。f_o 的计算如下：

$$f_0=\frac{Status_{(n)}-Status_{(n-1)}}{Status_{(n)}-Status_{(2)}} \tag{5}$$

为防止恶意簇首节点发送虚假报文丢弃通告对正常节点进行攻击，节点 i 连续在两个系统监测周期内都被判定为可疑节点后，系统即判定该节点为恶意节点，发出告警并通过 Hello 消息广播将该节点隔离出网络后，重新进行报文丢弃恶意节点检测。

如果簇首节点 j 在自己的监测周期内发动虚假报文丢弃攻击，使正常节点 i 在该监测周期内被判定为可疑节点，但下一监测周期中，由于更改了 IDS 簇头，节点 i 将不再被判定为可疑节点，则系统记录节点 j 为可疑节点，认为 j 在自己的监测周期内可能发动了虚假报文丢弃攻击，如果在节点 j 再次当选为 IDS 簇首周期内，再次有发送虚假报文丢弃攻击嫌疑，则系统认为节点 j 为恶意节点，发出告警并通过 Hello 消息广播将该节点隔离出网络。

为便于算法实现，需要定义节点状态数据结构：

```
tepydef struct{
Bool Drop;          //上一监测周期节点状态判定结果
Bool LastDrop;      //上一监测周期节点状态判定结果
Bool Cheat;         //簇首节点虚假报文丢弃攻击嫌疑
}Node;
```

算法伪码表示如下：当系统一个监测周期结束后，对每个节点执行如下算法，设节点 i 的上一周期 IDS 簇首节点为 j。

```
while(true)                          //系统按周期循环监测
{
if (node.Drop)                       //被判定为可疑报文丢弃节点
    if(node.LastDrop)                //上一周期也是可疑结点
    {Alarm(node);                    //报警并隔离节点，
    next;
    }                                //进入下一周期
else node.LastDrop = true;           //记录 node 为可疑节点
else                                 //node 未被判定为可疑节点
 if(i.LastDrop)                      //上一周期是可疑节点
    if(j.Cheat)//如果 j 已是可疑节点
    {Alarm(j);//报警并隔离节点 j，
    netxt;
    }                                //进入下一周期
else
{ j.Cheat = true;                    //记录 j 可疑节点
    node.LastDrop = false;           //节点 node 取消怀疑
}
}
```

3 实验与性能分析

本文使用 Qualnet[6] 作为模拟仿真实验平台。在 2 000×2 000 的区域内随机布置 100 个节点。仿真时间为 1 000 s，节点使用 Random Waypoint 移动模型，以 0~20 m/s 的速度运动。通信模型采用 CBR (Continuous Bit Rat) 流，CBR 流大小为 512 B。每秒钟发 10 个数据包，从源节点到目的节点有不间断的数据包发送，网络层采用 AODV 路由协议，实验中采用不同数量的恶意节点发动报文丢弃攻击和虚假报文丢弃攻击。影响仿真结果的因素很多，主要是环境的设置，如节点间发报的速率、节点移动速度、IDS 簇首监测周期大小、节点活动的范围。

评价算法性能的主要指标是恶意节点的检测率和恶意节点的误检率。检测率是指被检测出来的恶意节点个数占网络中实际的全部恶意节点个数的比例。误检率是指正常节点被误检为恶意节点的个数占整个网络中正常节点的比例。对检测率和误检率各做 10 次实验，然后取平均值。

图 4 表示了不同恶意节点数量下，全局感知算法的检测率。无论是否存在虚假报文丢弃攻击，算法对恶意节点的检测率均随着恶意节点在网络中所占比例的增加有所下降，因为恶意节点的增加导致了网络中节点报文转发率的异常数据增多，干扰了统计分析过程。图 5 表示了不同恶意节点数量下，全局感知算法对恶意节点的误测率。从图 5 可以看出，无论是否存在虚假报文丢弃攻击，算法的误检率都较低，虽然随恶意节点数量的增大而有小幅上升，但最高不超过 3%。

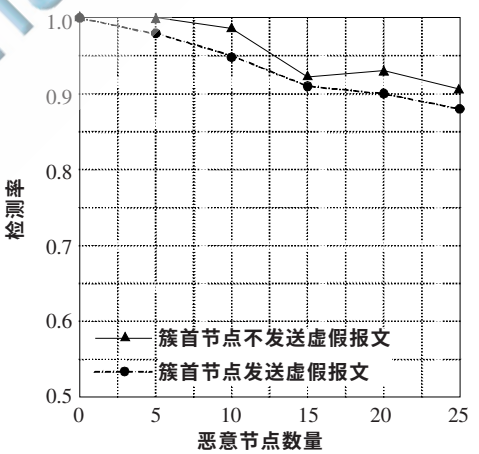

图 4 不同恶意节点数量下的检测率

实验表明，本文算法的恶意节点检测率和误检率性能良好。在恶意节点所占比例较低的情况下，检测率接近 100%，误检率接近 0%。随着恶意节点所占比例的提高和恶意节点发送虚假的邻居节点报文转发率信息，恶意节点的检测率有所下降，但仍保持在 90%以上；误检率有所上升，但仍保持在 3%范围内。

恶意簇首节点发送虚假的簇内节点，节点报文转发率信息对检测率造成了一定的影响，这主要是由于实验

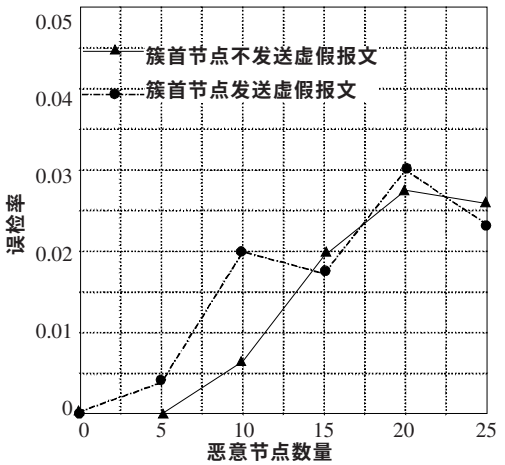

图5 不同恶意节点数量下的误检率

时指定的恶意节点并不一定会被选举为簇首。本算法对于发送虚假报文丢弃攻击，需要在节点两次当选为监测簇首才做出判定，但这并不会影响网络的性能。因为算法是基于簇结构的，恶意节点只有连续两次当选簇首才能对一个正常节点发动虚假报文诬陷攻击，而算法在成簇阶段已经阻止了此类攻击发生，所以节点无法发动虚假报文丢弃攻击，因此表现为检测率降低。在虚假报文丢弃攻击恶意节点比例较高的情况下，检测率平均降低了2%左右。

本文分析了现有依据邻居节点检测算法对报文丢弃攻击和虚假报文丢弃攻击进行检测算法存在的不足，在此基础上，设计了一种基于簇首协作的报文丢弃攻击全局感知算法。仿真实验表明，该算法通过对报文丢弃攻击节点进行全局监测，减轻了网络各节点资源和网络带宽的消耗；利用狄克逊准则进行恶意节点判定，提高了报文丢弃攻击的检测精确度；同时，由于使用了基于簇首的监测方式，恶意节点在作为簇成员的时候无法发动虚假报文丢弃攻击，很大程度上抑制了虚拟报文丢弃攻击在网络中发生的频率。由于算法基于全局监测周期进行消息交换，在周期结束的时候进行恶意节点的检测，因此，在提高入侵检测响应速率和改进Hello消息传播方式节省网络带宽方面需要进一步优化。

参考文献

[1] PAUL K, WESTHOFF D. Context aware detection of selfish Nodes in DSR based Ad hoc networks [C]. IEEE Global Telecommunications Conference, 2002: 178-182.

[2] LEE S, CHOI Y. A resilient packet-forwarding scheme against maliciously packet-dropping nodes in sensor networks [C]. Proceedings of the 4th ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'06). Alexandria, USA, 2006:59-70.

[3] MARTI S, GIULI T, LAI K, et al. Mitigating routing misbehavior in mobile Ad Hoc networks [C]. Proceedings of the 6th International Conference on Mobile Computing and Networking (Mobicom'00). Boston, USA, 2000: 255-265.

[4] ZHONG S, CHEN J, YANG Y R. Sprite: a simple cheat-proof credit-Based system for mobile ad hoc networks [C]. INFOCOM 2003, twenty Second Annual Joint Conference of the IEEE Computer and Communications. San Francisco, CA, USA, April, 2003:1987-1997.

[5] 王芳，易平，吴越，等.基于规范的移动Ad Hoc网络分布式入侵检测[J].计算机科学，2010，37(10)：118-122.

[6] JAIKAEO C, SHEN C C. Qualnet Tutorial [R]. Scalable Network Technologies, University of Delaware, 2007:21-46.

(收稿日期：2011-04-13)

作者简介：

靳倩慧，女，1986年生，硕士研究生，主要研究方向：计算机网络与安全。