高技术通讯 2009 年 第 19 卷 第 9 期:901 ~ 906

doi:10.3772/j.issn.1002-0470.2009.09.005

OIECE:基于一跳信息转发的自组网节点合作协议①

郭建立② 吴智博 刘宏伟 董 剑 杨孝宗

(哈尔滨工业大学计算机科学与技术学院 哈尔滨 150001)

摘 要 针对移动自组网中自私节点的不合作问题,提出了用于检测并隔离网络中的不合作节点,减小它们对网络性能影响的 OIECE 方法。OIECE 方法工作时,节点周期地向邻居广播自己的检测结果,使得邻居节点能够快速发现周围的不合作节点,对它们进行有效的隔离;在路由发现过程中,源节点和目的节点之间的所有节点,都对经过的路由请求消息和路由应答消息进行过滤,丢弃那些含有不合作节点的路由,有效减小了不合作节点对网络的危害;当节点发送数据时,优先选择含有可信节点的路由,提高了数据发送的成功率。仿真结果显示,当网络中存在不合作节点时,OIECE 方法能够显著提高合作节点的吞吐率,同时还能够对不合作节点进行有效的惩罚,降低它们的吞吐率。

关键词 移动自组网(MANETs),节点合作,一跳信息,看门狗,信誉

0 引 言

近年来,随着硬件技术的不断进步,出现了大量的民用自组网,如临时在大学教室或飞机场中组建的自组网。在这些网络中,各节点分别隶属于不同的个人或组织,它们缺乏共同目的,因此节点间的合作无法得到保证,某些节点可能会为了节省资源(如能量、内存或 CPU 等)而表现出不合作行为,丢弃那些需要转发的数据。在移动自组网(mobile Ad hoc networks,MANETs)[1]中,不合作节点将会严重影响网络性能。文献[2]指出,当网络中存在 10% ~ 40%的不合作节点时,整个网络的性能将会下降 16% ~ 32%。

Marti 等[2]最先提出使用看门狗(watchdog)检测不合作节点,但这种方法对不合作节点的检测速度较慢,不能够及时有效地把不合作节点从网络中隔离开。Bansal 等[3]提出了基于观察的自组网合作(OCEAN)方法,在路由请求消息中增加一个 avoid-list 域,使得路由发现过程尽量避开不合作节点,但该方法效果不够明显,同时还引入了新的安全问题。Buchegger 等[4]提出采用二手信息(second-information)计算节点的信誉值,并通过贝叶斯统计方法[5]减轻流言攻击,提高了合作节点的吞吐率,但该方法存在很多缺点[6,7],如需要为每个节点保存一个信誉值,占用了大量存储空间;传播二手信息占用大量网络带宽;更容易受到攻击,等等。

本文提出了一种适用于 MANETs 的基于一跳信息转发的合作(one-hop information enhanced cooperation enforcement,OIECE)方法,采用一跳信息计算节点的信誉值,能够加快对不合作节点的检测速度。同时,OIECE 方法还对路由控制消息进行了过滤,防止不合作节点出现在路由中。通过 ns2 的仿真实验,我们发现 OIECE 能够显著提高合作节点的吞吐率,同时还能对不合作节点进行有效的惩罚。

1 相关工作

针对移动自组网中节点合作问题,研究者们提出了许多解决方案[8,9]。Michiardi 等[10]提出针对每种网络服务(如转发服务或路由发现服务)建立单独的信誉值,然后通过加权方法计算出节点总的信誉值。Buttyan 等[11]引入一种被称作 Nuglets 的虚拟货币,源节点向转发节点支付一定数量的 Nuglets 作为报酬,激励节点对数据进行转发,但这种方法需要特殊硬件的支持。Anderegg 等[12]提出了 Ad hoc-VCG 方法,使用了机制设计原理,把数据转发过程看作一个二级密封价格拍卖活动。Wang 等[13]提出的 LOTTO 方法是对 Ad hoc-VCG 方法的改进,使路由发现过程的消息负载从 $O(n^3)$ 降低到 $O(n^2)$。Eidenbenz

① 863 计划(2006AA01A103)和国家自然科学基金(60503015)资助项目。
② 男,1980 年生,博士生;研究方向:移动自组网及传感器网络;联系人,E-mail: gjl@ftcl.hit.edu.cn
(收稿日期:2008-05-15)

等[14]提出的 COMMIT 方法使用了拓扑控制方法，并对 Ad hoc-VCG 的支付方案进行了修改，使源节点满足个人理性约束。Dorsey[15]提出的 XPM 方法，把路由选择过程看作一个组合交换（combinatorial exchanges）过程，利用博弈论方法寻找最优路径，同时 XPM 还考虑了节能问题。

2 OIECE 方法

OIECE 方法基于动态源路由（dynamic source routing，DSR）协议，位于网络层和媒体接入层（MAC）层之间，如图 1 所示，包含三个组件：看门狗（watchdog）、过滤器（filter）和信誉管理器（reputation manager）。

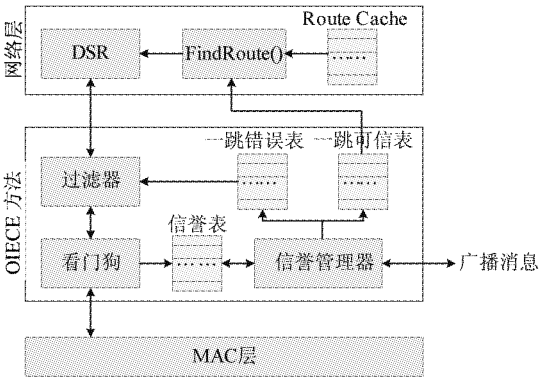

图 1 OIECE 结构

2.1 看门狗

看门狗在混杂模式下对邻居节点进行监听，观察它们是否对数据进行转发，以及在转发过程中是否修改了数据的内容，并对信誉表（reputation table）进行更新。信誉表中存放每个邻居所对应的信誉值，初始为 0，取值范围为[－1，1]。定义两个阀值 $Thre_{Trust}=0.8$ 和 $Thre_{Faulty}=-0.5$，并规定信誉值大于 $Thre_{Trust}$ 的节点为可信节点，小于 $Thre_{Faulty}$ 的节点为不合作节点，介于二者之间的为中立节点。

对于路由层发出的数据包，只要其下一跳节点不是目的节点（目的节点不对数据包转发，因此无法对其进行监听），就被看门狗放入监听缓存中。每当看门狗捕获一个数据包，就去监听缓存中查找，如果找到并且没有被恶意篡改，就增加该转发节点的信誉值，并根据公式

$$newRV=\begin{cases}oldRV\times a_1+b_1 & \text{if } oldRV<Thre_{Faulty}\\ oldRV\times c_1+d_1 & \text{if } oldRV\geqslant Thre_{Faulty}\end{cases} \tag{1}$$

计算节点新的信誉值（实验中取 $a_1=0.99$，$b_1=0.01$，$c_1=0.9$，$d_1=0.1$）。

如果直到监听缓存中的数据包超时，看门狗都没能观察到对该数据包的转发，或者看门狗发现转发节点对数据进行了恶意修改，就减小转发节点的信誉值，并根据公式

$$newRV=\begin{cases}oldRV\times a_2-b_2 & \text{if } oldRV\geqslant Thre_{Trust}\\ oldRV\times c_2-d_2 & \text{if } oldRV<Thre_{Trust}\end{cases} \tag{2}$$

计算节点新的信誉值（实验中取 $a_2=0.98$，$b_2=0.02$，$c_2=0.9$，$d_2=0.1$）。

公式（1）和（2）使节点的信誉值具有如下性质：1）信誉值能够反映节点过去的行为；2）节点最近的行为对信誉值的影响更大；3）不合作行为使节点的信誉值快速地低于 $Thre_{Faulty}$；4）不合作节点的信誉值增加缓慢，要经过较长时间的合作行为才能使信誉值大于 $Thre_{Faulty}$；5）偶然连续丢失 2～3 个数据包并不会使可信节点的信誉值降到 $Thre_{Trust}$ 以下。

为使看门狗建立的信誉值更精确，引入了共同邻居监听技术。看门狗每捕获一个数据包，只要其当前转发节点和下一跳转发节点都是自己的邻居，就把它放入监听缓存中，对其进行监听。如图 2 所示，节点 S 通过 A、B、C 向节点 D 发送数据，节点 M 位于 B 和 C 的传输范围内。当 B 向 C 转发数据时，M 能够捕获这个数据包，并发现当前转发节点（节点 B）和下一跳转发节点（节点 C）都是自己的邻居，M 就把该数据包放入监听缓存中，监听节点 C 对该数据包的转发情况，并相应更新 C 的信誉值。

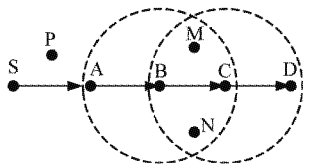

图 2 共同邻居监听

作为对不合作节点的惩罚，过滤器会丢弃所有来自不合作节点的数据。如图 2 所示，假定源节点 S 是一个不合作节点，并且被节点 A 发现了，作为惩罚，节点 A 会丢弃所有来自 S 的数据。为了防止看门狗把这种惩罚行为误判为不合作行为，要求看门狗不对路由中第一个转发节点进行监听，如图 2 中，节点 P 不对节点 A 进行监听。

2.2 信誉管理器

信誉管理器周期地检查信誉表，把可信节点和

不合作节点放入广播消息中,周期地向邻居进行广播。同时,信誉管理器还接收来自邻居的广播消息,将其放入缓存中,并对其进行处理。

信誉管理器对广播消息的处理过程如下:首先,认为来自不合作节点的广播消息是不可信的,丢掉它们;其次,认为自己的检测结果最可信,所有与自己的检测结果相矛盾的广播消息都被认为是不可信的,丢掉它们;接着,两个来自同级别的广播消息(如都来自可信节点,或都来自中立结点),如其内容相矛盾,则同时丢掉它们;最后,认为来自可信节点的广播消息是较可信的,丢掉所有与之相矛盾的、来自中立节点的广播消息。

经过上述过程的处理,剩下的广播消息被认为是有用的。信誉管理器把这些广播消息中的可信节点合并起来,对一跳可信表(one-hop trusted table, OHTT)进行更新;把不合作节点合并起来,对一跳错误表(one-hop faulty table, OHFT)进行更新。

2.3 过滤器

过滤器主要根据 OHFT 对经过的路由控制消息,如路由请求消息(route request, RREQ)和路由应答消息(route reply, RREP),进行选择性过滤,具体算法如图 3 和图 4 所示。对于 RREQ,如果其源节点是可信节点或中立节点,并且在已发现的路由中含有不合作节点,过滤器将其丢掉。对于 RREP,如果其目的节点是可信节点或中立节点,并且含有不合作节点,过滤器将其丢弃,因为这个路由是不可用的,使用它发送的数据将不会到达目的节点。

```
1  收到一个 RREQ
2  If 源节点是可信或中立节点,并且含有不合作节点 then
3      丢弃该 RREQ 并返回
4  end if
5  把这个 RREQ 交给路由层处理
```

图 3 转发 RREQ 的算法

```
1  收到一个 RREP
2  If 目的是可信或中立节点,并且路由中含有不合作节点 then
3      丢弃该 RREP 并返回
4  Else if 目的是不合作节点,并且路由中全部是合作节点 then
5      丢弃该 RREP 并返回
6  End if
7  把这个 RREP 交给路由层处理
```

图 4 转发 RREP 的算法

在路由发现过程中,位于源和目的节点之间的所有节点都在进行路由过滤,在这些节点的帮助下,可信节点或中立节点所发现的路由最大程度地绕开了不合作节点,从而增加了数据发送的成功率。与之相反,在由不合作节点所发起的路由发现过程中,那些绕开不合作节点的可用路由会在返回的途中被丢弃,这将降低不合作节点发送数据的成功率,起到了对不合作节点惩罚的作用。

2.4 对 DSR 的修改

在 DSR 协议中,当有数据要发送时,就去路由缓存(route cache)中查找最短路由,然后用这个最短路由发送数据。如果路由缓存中没有可用路由,DSR 就会启动路由发现过程,重新搜索路由。为进一步提高节点发送数据的成功率,尽可能避免使用含不合作节点的路由,OIECE 记录附近的可信节点,把它们保存在 OHTT 中,同时修改 DSR 协议的 findRoute ()函数,使其优先返回包含可信节点的路由。

2.5 第二次机会机制

多种原因(如信号冲突、网络拥塞等)会影响看门狗的检测结果,把合作节点错误地标记为不合作节点。另外,那些被检测出的不合作节点,也有可能在后期乐意进行往来合作,为了给它们一个“改过自新”的机会,引入了第二次机会机制。规定所有被放入 OHFT 中的节点,经过一段时间后,将被放出来,但它们的 rating 值不会复位到 0,而是保持当前值不变。这样做的目的是一旦这些节点继续表现出不合作行为,能够快速地被重新发现。

3 仿真实现及结果分析

本节采用 ns-2[16]对 OIECE 方法进行验证,观察其对网络性能的影响,同时对比 pathrater[2] 和 OCEAN[3]方法进行分析,仿真基本参数如表 1 所示。

表 1 仿真基本参数

节点移动模型	Random waypoint
仿真持续时间	1000s
传输范围	250m
接收范围	250m
载波监听范围	550m
传输速率	2Mbit/s
传输类型	CBR
数据包大小	512byte
发送速率	5pkt/s

采用以下两个参数对协议进行评价：

(1) 吞吐率(throughput)：到达目的节点的数据包个数与节点发出的数据包个数的比值。

(2) 协议开销(protocol overhead)：OIECE 方法产生的广播消息的总数与网络中发送和转发的数据包总数的比值。

3.1 吞吐率

通过设计静态、动态和大规模三种仿真场景对吞吐率进行比较分析，每个场景的具体参数设置如表 2 所示。

表 2 三个场景的具体参数

场景	仿真区域(m^2)	节点数(个)	CBR 连接(对)	移动速度(m/s)	停留时间(s)
静态网络	670×670	50	50	0	0
动态网络	670×670	50	50	10	100
大规模网络	945×945	100	100	10	100

首先观察静态网络的吞吐率。仿真结果如图 5 所示，其中 Coop 表示合作节点，N-Coop 表示不合作节点，DSR 表示不使用节点合作方法。从图 5 中可以看出，OIECE 方法显著提高了合作节点的吞吐率，使其增加 10% ~ 32.5%，而不合作节点的吞吐率也有了明显降低。在 Pathrater 方法中，不合作节点的吞吐率较高，这是因为 Pathrater 方法缺少惩罚机制。另外，OCEAN 方法中不合作节点的吞吐率也较高，主要原因是：不合作节点只能被少数几个邻居检测到，一般情况下，可以绕开这几个邻居，通过其它邻居把数据发送出去，惩罚效果不明显；另外，在路由发现过程中，OCEAN 方法引入的 avoid-list 有助于不合作节点找到“好”的路由，提高了不合作节点的吞吐率。

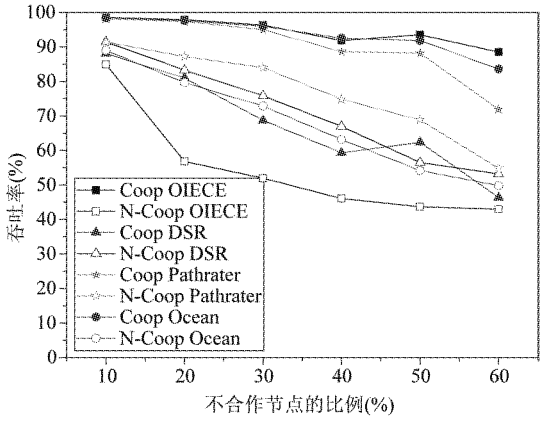

图 5 静态网络

接下来研究动态网络的吞吐率，仿真结果如图 6所示。同样可以看出，OIECE 方法中合作节点的吞吐率最高，不合作节点的吞吐率最低。另外，相比于静态网络，OIECE 方法中合作节点的吞吐率有所下降，而不合作节点的吞吐率却有所升高。这主要是由节点的移动造成的。一方面节点的移动会造成路由中断，使得一些数据包丢失；另一方面，节点的移动还会使它周围的邻居发生变化，而节点对新邻居的行为并不清楚，因而增加了选用含有不合作节点路由的概率，从而导致数据丢失，降低了合作节点的吞吐率。与此相反，对于不合作节点，当其移动到一个新的位置时，附近的节点会把它当成中立节点，不对其进行惩罚，从而提高了不合作节点的吞吐率。

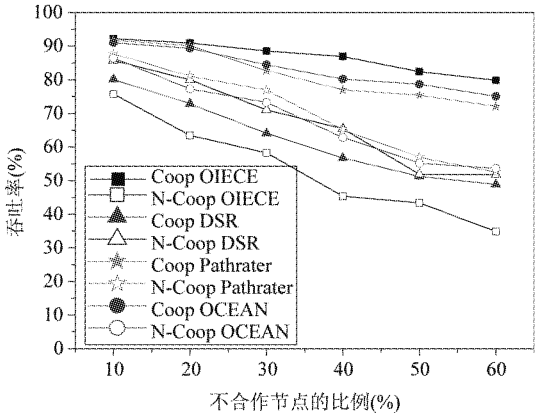

图 6 动态网络

大规模网络的仿真结果如图 7 所示。可以看出，相比于 Pathrater 和 OCEAN 方法，OIECE 方法中

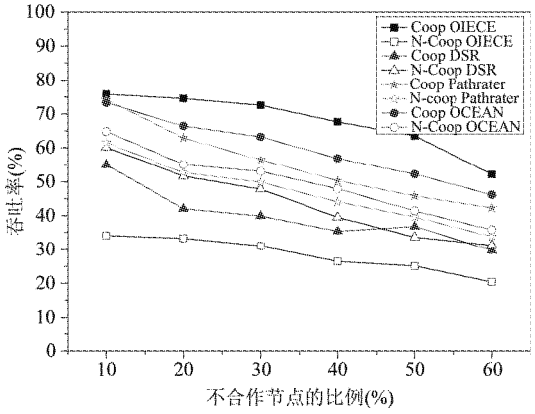

图 7 大规模网络

合作节点的吞吐率有了明显提高。另外,相比于静态和动态网络,OIECE 方法中合作节点的吞吐率降低了大约 20%。主要原因是:随着网络规模的扩大,发送数据时所使用的路由变长了,连接更容易中断,从而丢失数据。

在仿真中,直接传送的数据所占比例较大,最高达到 40%,由于这些直接传送的数据不需要其它节点的转发,因此几乎 100% 发送成功。为了更清楚地观察 OIECE 方法的吞吐率,我们除去结果中直接传送的数据,同时对比三种场景,观察合作节点与不合作节点吞吐率的变化,如表 3 和表 4 所示。可以看出,OIECE 方法中合作节点与不合作节点的吞吐率相差 39.2% ~ 70.7%,如此大的差距完全起到了对不合作节点的惩罚作用,使其放弃不合作行为。

表 3 合作节点的吞吐率(去掉直接传送的数据)

不合作节点比例	静态网络	动态网络	大规模网络
10%	97.3%	88.5%	68.7%
20%	96.5%	81.2%	64.1%
30%	95.4%	80.0%	63.5%
40%	88.7%	76.5%	57.9%
50%	90.0%	73.1%	48.9%
60%	76.6%	57.5%	36.2%

表 4 不合作节点的吞吐率(去掉直接传送的数据)

不合作节点比例	静态网络	动态网络	大规模网络
10%	30.8%	45.7%	24.9%
20%	26.5%	39.0%	22.5%
30%	24.7%	25.9%	19.8%
40%	21.4%	22.5%	12.8%
50%	22.6%	16.4%	9.7%
60%	17.9%	10.1%	6.5%

3.2 协议开销(protocol overhead)

OIECE 需要周期地向邻居发送广播消息,因此引入了协议开销,其计算公式如下:

$$\text{协议开销} = \frac{\text{广播消息总数}}{\text{广播消息总数} + \text{发送和转发的数据包总数}} \times 100\%$$

图 8 给出了动态和大规模网络中,OIECE 方法的协议开销。可以看出,协议开销最高为 15.6%,最低为 8%,大规模网络要比动态网络的开销小,可见 OIECE 引入的开销并不高。另外需要强调的是,仿真时为了避免发生网络拥塞以致影响节点吞吐率的测量,采用的网络负载较小。而 OIECE 方法引入的广播消息数量只与网络中节点的个数和仿真过程持续的时间有关,不随网络负载的增大而变化,因此当网络负载增大后,协议开销会更小。

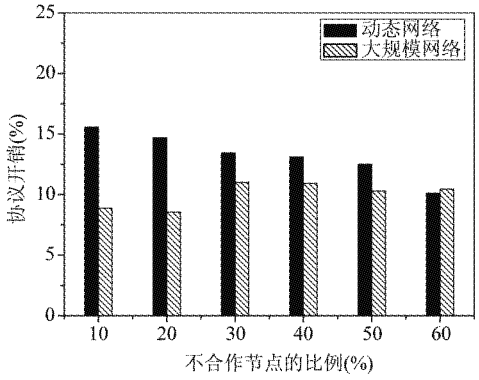

图 8 协议开销

4 结 论

本文提出的 OIECE 方法能够快速检测出移动自组网中的不合作节点,对其进行隔离,减小它们对网络性能的危害。相比于直接信息方法,OIECE 对不合作节点的检测速度更快更准确,既提高了合作节点的吞吐率,还对不合作节点进行了有效的惩罚。一跳信息的使用并没有过多增加系统实现的复杂度,相比于二手信息方法,OIECE 只需要向邻居广播消息,减少了网络中控制消息的数量,使得网络具有很好的可扩展性,同时 OIECE 丢弃来自不合作节点的广播消息,减小了系统受恶意节点攻击的可能性,提高了系统的可靠性和可用性。

参考文献

[1] Chlamtac I, Conti M, Liu J. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks*, 2003, 1(1): 13-64

[2] Marti S, Giuli T, Lai K, et al. Mitigating routing misbehavior in mobile ad hoc networks. In: Proceedings of the 6th Annual IEEE/ACM International Conference on Mobile Computing and Networking, Boston, USA, 2000. 255-265

[3] Bansal S, Baker M. Observation-based cooperation enforcement in ad hoc networks: [technical report]. California: Stanford University, 2003

[4] Buchegger S. Coping With Misbehavior in Mobile Ad-hoc Networks: [Ph.D dissertation]. Zurich Switzerland: Swiss Federal Institute of Technology (EPFL), 2004

[5] Buchegger S, Boudec J L. The effect of rumor spreading in reputation systems for mobile ad-hoc networks. In: Proceed-

ings of Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, Sophia-Antipolis, France, 2003. 66-75

[6] Hu J Y. Cooperation in mobile ad hoc networks. http://www.cs.fsu.edu/research/reports/TR-050111.pdf : Florida State University, 2005

[7] Laniepce S, Demerjian J, Mokhtari A. Cooperation monitoring issues in ad hoc networks. In: Proceedings of International Wireless Communications and Mobile Computing Conference, Vancouver, Canada, 2006. 695-700

[8] Marias G F, Georgiadis P, Flitzanis D, et al. Cooperation enforcement schemes for MANETs: a survey. *Wireless Communications and Mobile Computing*, 2006, 6(3): 319-332

[9] Yoo Y, Agrawal D P. Why does it pay to be selfish in a MANET? *IEEE Wireless Communications*, 2006, 13(6): 87-97

[10] Michiardi P, Molva R. CORE: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In: Proceedings of the 6th IFIP Communication and Multimedia Security Conference, Portoroz, Slovenia, 2002. 107-121

[11] Buttyan L, Hubaux J P. Stimulating cooperation in self organizing mobile ad hoc networks. *ACM/Kluwer Mobile Networks and Applications*, 2003,8(5):576-592

[12] Anderegg L, Eidenbenz S. Ad-hoc-VCG: a truthful and cost efficient routing protocol for mobile ad-hoc networks with selfish agents. In: Proceedings of the 9th Annual IEEE/ACM International conference on Mobile Computing and Networking, San Diego, USA, 2003. 245-259

[13] Wang Y W, Singhal M. On improving the efficiency of truthful routing in MANETs with selfish nodes. *Pervasive and Mobile Computing*, 2007, 3(5):537-559

[14] Eidenbenz S, Resta G, Santi P. The COMMIT protocol for truthful and cost-efficient routing in ad hoc networks with selfish nodes. *IEEE Transactions on Mobile Computing*, 2008, 7(1):19-33

[15] Dorsey J G. Game-Theoretic Power Management in Mobile Ad Hoc Networks: [Ph.D dissertation]. Pittsburgh, Pennsylvania: Carnegie Mellon University, 2004

[16] Fall K, Varadhan K. The ns manual (formerly ns notes and documentation). http://www.isi.edu/nsnam/ns/doc/index.html: The VINT Project, 2009

OIECE: one-hop information enhanced cooperation enforcement scheme for MANETs

Guo Jianli, Wu Zhibo, Liu Hongwei, Dong Jian, Yang Xiaozong

(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001)

Abstract

To cope with the selfish nodes in mobile ad hoc networks (MANETs), the paper proposes a new cooperation enforcement scheme, called the OIECE, to quickly detect and isolate the non-cooperation nodes, decreasing their effect on the networks' performance. In OIECE, each node broadcasts its detecting result to neighbors periodically, making its neighbors find the non-cooperative nodes around it quickly and isolate them from the networks. In the route discovery phase, all the nodes between the source node and the destination node filter the route requests and the route replies, suppressing the routes containing non-cooperative nodes. In the forwarding phase, the source node gives priority to the routes include trust nodes, making the packets more possibly arrive at the destination node. The simulation results show that OIECE can highly improve the throughput of cooperative nodes and severely punish the non-cooperative nodes.

Key words: mobile Ad hoc networks (MANETs), node cooperation, one-hop information, watchdog, reputation