A Trust Model Based Cooperation Enforcement Mechanism in Mesh Networks

Zhiming Zhang[1,2], Weirong Jiang[1,2], Yibo Xue[2,3]
[1] Department of Automation, Tsinghua University, Beijing, China
[2] Research Institute of Information Technology, Tsinghua University, Beijing, China
[3] Tsinghua National Lab for Information Science and Technology, Beijing, China
zzm02@mails.tsinghua.edu.cn

***Abstract*—Mesh networks are autonomous, where each node is not only a terminal but also a router. Due to this feature, mesh networks demand nodes to cooperate with each other to improve the availability and the performance, as well as to enforce the security. A cooperation enforcement mechanism based on a node trust model is addressed in this paper in order to evaluate the trustworthiness of neighboring nodes. Simulation results validate that the proposed mechanism can respond rapidly to the behaviors of other nodes, and significantly reduce packet loss rate when malicious or selfish nodes present.**

Keywords-PID; trust model; cooperation enforcement; mesh network

I. INTRODUCTION

As a novel mobile wireless multi-hop network, ad hoc network has evolved into mesh networks recently. The features of mobile, autonomous, multi-hop and wireless make it very different from traditional wired networks, and also introduce many potential security vulnerabilities.

The routers, believed by the end users in wired networks, become disbelieving in the "infrastructureless" mesh networks. A malicious router/node may disrupt, discard, falsify, and eavesdrop the data passing through themselves, or mislead the routing protocol, or even consume the valuable network resources by injecting useless packets into networks. A selfish router/node may refuse to relay data for others to conserve its power and decrease its CPU consumption.

Facing the new security challenges, some solutions have been proposed mostly in two categories: the prevention mechanisms and the detection and reaction mechanisms [2, 3].

A. Prevention mechanisms: The main ideas are to build some cryptography-based mechanisms to prevent any selfish or malicious behavior. Most proposals assume that a group of nodes have shared symmetrical key or each node can acquire others' public keys, then they can ensure security through encryption or hash functions [4, 5, 6].

B. Detection and reaction mechanisms: All or some of the nodes monitor the network and take some reactions according to the corresponding behaviors of other nodes, such as intrusion detection and cooperation enforcement. Some classical proposals are: Watchdog and Pathrater [13], CONFIDANT[14], CORE [15], etc.

A prevention-only strategy will only work when the prevention mechanism is perfect. Otherwise, someone will find out how to get around them. Most of the attacks and vulnerabilities have been the results of bypassing prevention mechanisms [8, 14]. Therefore, it is usually necessary to have a detection and reaction mechanism as a complement to the practical (non-perfect) prevention mechanism. Moreover, in a mesh network, the negative effects of the selfish behaviors shall not be underestimated as they may bring great impacts to the whole mesh network [10].

In order to enforce the cooperation among the nodes, two major types of mechanisms have been proposed [3, 7]:

A. Incentive based mechanism: Based on virtual currency, these methods use additional hardware or central settlement to enforce the cooperation among nodes. But adding additional equipments may be infeasible and the centralized solution loses the robustness. Moreover, this mechanism, like the prevention mechanism, can not be perfect, so the presence of unfairness is inevitable [11, 12].

B. Punishment based mechanism: Mainly composed of trust management systems. In this mechanism, trust model can be adopted as a measure to avoid the misbehavior nodes by using the first hand information, but it does not punish the selfish or malicious nodes [13]. CONFIDANT calculates the first hand trust value by detecting the behaviors of other nodes, and exchanges these trust values as the second hand information. In order to avoid the bad mouthing attacks, the Bayesian statistics is also introduced to solve this problem [14]. CORE combines three reputation values—subjective reputation, indirect reputation and function reputation—to form the final reputation value [15]. Although nodes only exchange positive reputation information, the false praise (good mouthing) makes malicious nodes harder to detect. But the negative effect of the exchange of positive reputation information challenges this mechanism. Using adaptive forgetting factor to trace the behaviors of other nodes rapidly, an entropy-based model and a probability-based model have also been proposed to calculate the trust value and to avoid the bad mouthing attacks [16].

Most of the trust models include exchanging reputation information within the network. But this mechanism suffers

some significant problems, such as high overhead and false alarms (false negative and false positive). Although some of them only adopt the first hand information, the procedure and configuration are quite complex [1]. Moreover, in an ad hoc network, nodes can have high mobility. When a node is on the move, it may lose neighbors with established trust relations and must build trust relationship with its new neighbors quickly. Therefore, it is rational to attain the trust value rapidly by exchanging information with neighboring nodes while maintain performance at a certain level.

As ad hoc networks evolving into mesh networks, it tends to be more stationary. The relatively static feature of mesh networks makes it possible that the nodes can establish the trust relationship with its neighbors based on the relatively stable detection results. What's more, once there are some selfish or malicious nodes, the bad/good mouthing attacks will lower tracking speed of trust models under the condition of ensuring certain veracity level. Therefore, the mechanism of exchanging information is a tradeoff between robustness and tracking speed. According to these observations, only utilizing the first hand information is reasonable and should be able to decrease the false alarm rate.

In this paper, we introduce a reactive protection mechanism, a trust model based cooperation enforcement mechanism, which only utilizes the first hand information to track the behaviors of others rapidly. Moreover, the parameters of the trust model can be configured easily.

The main contribution of this paper is that a PID like trust model is proposed, based on which the cooperation enforcement mechanism can acquire relatively rapid speed to track the behaviors of the neighbors, reduce the loss rate of mesh networks with low complexity of computing and configuration while only utilizing the first hand information. Moreover, the recommended configuration of trust model parameters brings negligible negative impact to the mesh networks.

The remainder of this paper is organized as follows. Section II elaborates the details of our trust model based cooperation enforcement mechanism. Section III illustrates and analyzes the simulation results. Section IV discusses the parameter selection of the trust model. Lastly section V concludes this paper and provides some open problems to be solved in future.

II. COOPERATION ENFORCEMENT MECHANISM

A. Trust Model

In order to avoid the shortcomings from exchanging information, the reaction mechanism proposed in this paper relies only on the first hand information to protect the mesh network. Enlightened by [9, 17], we designed a PID (Proportional-Integral-Derivative) like trust model into the mesh network.

A continuous system with PID controller [19] is shown in Figure 1 (a frame copied from MATLAB), where $\frac{1}{s}$ is the integration in the form of Laplace transformation, and du/dt denotes the derivative. In this system, the transfer

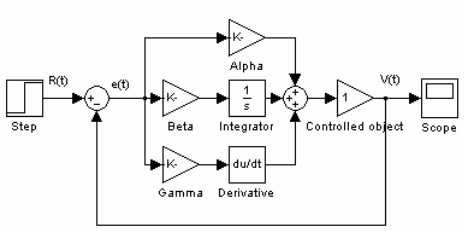


Figure 1. Standard PID controller

function of controlled object can be considered as "1". The corresponding equation is as follows:

$$e(t) = R(t) - V(t)$$
$$V(t) = \alpha * e(t) + \beta * \int_0^t e(t)dt + \gamma * \frac{d}{dt}e(t) \quad (1)$$

Equation 1 resembles a PID controller used in control systems. The input $R(t)$ is the detected result of the neighbor's behavior and $V(t)$ is the corresponding output. The right of the lower equation is composed of three components. The first component indicates the contribution of current difference between input and output. The second component refers to the record of history about difference. The last component reflects the sudden changes of the difference which can also be replaced by output. α, β, and γ represent the weights of the three parts respectively.

When the node find that the neighbor behaves maliciously or selfishly, the input of the corresponding trust model $R(t)$ will be assigned as "0"; otherwise, it will be assigned as "1". That is to say, the input is "0" or "1". Once the input changes between "1" to "0," the output will fluctuate sharply. That is not the desired result.

In practice, the PID controller can (or must) be tuned to fit the practical application. Analyzing the Equation 1, we can find out that the proportional component brings a jump when the input changes, and the derivative ingredient is sensitive to the fluctuation of input. We need a trust value changing in a smooth manner. In order to achieve that, α and β must be relatively small enough values compared with that of β, or removed from equation 1 completely.

According to the control theory, the input is zero order signal, and the controller only with one integration can track the input in time and get no static difference. Moreover, our goal is to trace the behaviors of the neighbors at a higher speed and attain a smooth change of trust value. Therefore, we take off the proportional and derivative components while preserve the integral one to conclude a relative simple but usable model. Now, the trust model is shown in Figure 2.

The corresponding equation is:

$$e(t) = R(t) - V(t)$$
$$V(t) = \beta * \int_0^t e(t)dt \quad (2)$$

Under the conditions:

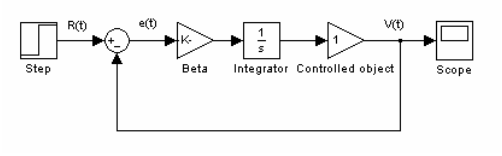
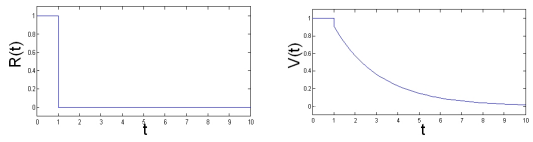
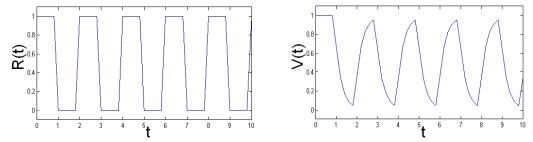


Figure 2. Reduced PID as trust model



A. Illustration of tracking speed



B. Dynamics of this trust model

Figure 3. Simulation of trust model in MATLAB

$$R(t)=\begin{cases}1, t<0\\0, t\geq 0\end{cases}$$

$$V(t)=1, t<0$$

We can resolve out

$$V(t)=e^{-\beta t}, t\geq 0 \qquad (3)$$

Equation 3 can be used to calculate the parameter β.

In order to utilize this trust model, the Equation 2 is discretized as a discrete equation, which is shown in Equation 4:

$$\begin{aligned} &e_B^A(k+1)=R_B^A(k+1)-V_B^A(k)\\ &V_B^A(k+1)=\beta*\sum_{n=0}^{k+1}e_B^A(n)\\ &\beta=\begin{cases}\beta_+, e_B^A(k+1)\geq 0\\ \beta_-, e_B^A(k+1)<0\end{cases}\end{aligned} \qquad (4)$$

Where $V_B^A(k+1)$ denotes the trust value of node B in time k+1 recorded in node A. The others have the similar meanings corresponding to those in equation 2. In addition, we set $\beta_+<\beta_-$ to give higher punishment for selfish and malicious behaviors so as to discover abnormal nodes as soon as possible.

Figure 3 presents some simulation results of this trust model in MATLAB. In Figure 3A, the left figures denote the inputs, i.e. the neighbor's behaviors, and the right figures are the corresponding outputs—the corresponding trust values. Different output speed to trace the input can be acquired by adjusting β. The results indicate that this trust model can discover the abnormal node in time.

The left of Figure 3B stands for that the node has 50 percent chances to behave selfishly and maliciously, and the corresponding right figure is the output *V(t)*. The result of simulation shows that the output *V(t)*, the trust value, can trace the input, the node's behaviors, very well.

B. Trust Model Based Cooperation Enforcement Mechanism

How this mechanism starts up is a very important problem. In the practical scenarios, if there is an authentication and management center, this problem will be solved easily. If there is no such kind of center, we assign a value between threshold and 1 as the starting trust value (V_i) (the detailed illumination will be shown in the later sections)

In addition, we define a threshold of trust value V_T to indicate whether this node should believe its neighbors.

REGULATION I:

$$R_B^A(k)=\begin{cases}0, & \textit{If node B behaviors maliciously or selfishly at time k.}\\ 1, & \textit{If node B behaviors normally at time k.}\end{cases}$$

REGULATION II:

1) If $V_B^A(k)\geq V_T$, A believes B

2) If $V_B^A(k)<V_T$, A does not believes B, and B will be put in A's black list and excluded from A's neighbors.

Using V_T, N to substitute $V(t)$ and t in equation 3 respectively, we can deduce the parameter β:

$$\beta=-\frac{\ln V_T}{N} \qquad (5)$$

Where, N is the number of steps in which the trust value changes from 1 to V_T when the input is always 0 after certain time point, defined as the speed to trace the behaviors of neighbors.

An example of the PID trust model based cooperation enforcement mechanism is shown in Figure 4. Every node monitors the behaviors of its neighbors. Based on the monitored result, the input of Equation 4 is figured out according to REGULATION I, and the corresponding output (i.e. trust value) is computed. Then the node will take the "believe" or "disbelieve" action according to the REGULATION II. For example, when the monitor of node 1 detects that the node 3 behaves selfishly at time k+1, $R_3^1(k+1)$ is set as "0" according to REGULATION I. Then the $V_3^1(k+1)=0.496381$ is computed based on the current system state. Because $V_3^1(k+1)=0.496381<V_T=0.5$, the node 3 is put into the black list of node 1 and removed from its neighbor list. The node 1 will not select the path through node 3 and do not relay any packets for it.

Certainly, we cannot guarantee that the detection result is always absolutely right, so we give a chance to recover for the nodes in the blacklist (*BNTimeout*) (The detailed illumination will be shown in the later sections).

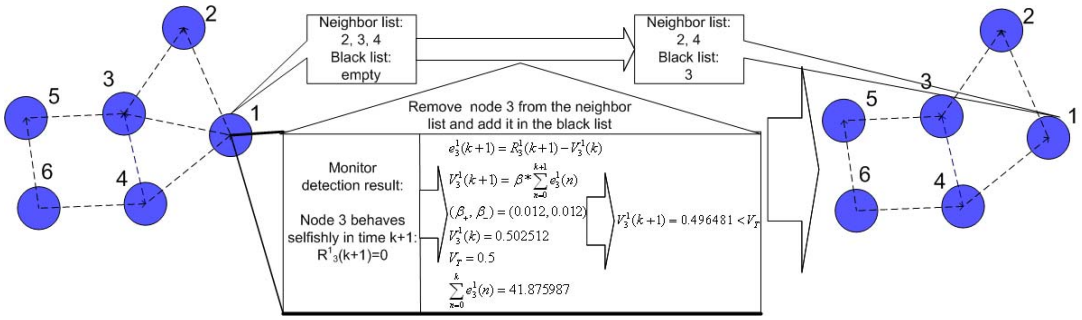


Figure 4. Illustration of trust model based cooperation enforcement mechanism

III. SIMULATION EXPERIMENTS

A. Overview

In order to further understand and validate this trust model based mechanism, some simulation experiments are implemented in NS-2. In these simulations, the selfish behavior is defined as that the node does not relay data for others except routing messages but requires other nodes to forward data for itself.

There are no straightforward ways to monitor the behavior of neighboring nodes. In our simulation, we suppose that all the nodes in the mesh network use the same channel. The example of monitoring process is shown in Figure 5, when node A sends data to destination node C, the data must be relayed by node B (or D). According to the properties of MAC layer protocol and the OLSR [18] routing protocol, the links on the way from A to C must be symmetrical. Therefore, when the node B (or D) relays the data for A to C, A should be able to monitor that transmission process. If node A does not monitor that the node B (or D) relays data within a time limitation (*PTimeout*), node A will record a selfish behavior of node B (or D) to its reputation system. Otherwise, the result of normal behavior will be recorded. Here, the time "*k*" in equation 4 denotes the *k*th time detection result.

OLSR [18] is used as the underlying routing protocol. The objective of the simulation is to validate the trust model based cooperation enforcement mechanism and not to consider the selfish or malicious usage of routing protocol.

The uniform scenario parameters of simulations are used. The range of wireless channels is 250m. The size of space is 1000m by 1000m with 30 nodes. Because most of the nodes should be static and only few of them move occasionally, what's more, even though all the nodes are static, the connectivity of links may change due to all kinds of reasons. So we let the nodes move with a maximum speed of 0.5m/s to simulate the practical scenarios. Every node whose initial position is arranged randomly selects a destination position with a speed less than 0.5m/s, when it arrives at the destination, it will stay there for 30s. And each node launches a connection with a random destination node. The interval of UDP packets is 1s with a packet size of 512 bytes. The reason why we set these parameters is that we want to keep the packet loss rate in a low range; otherwise, the mesh network will lose its usability. Simulation time is 2000s.

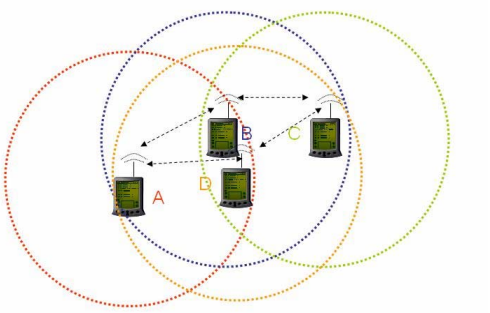


Figure 5. Example of monitoring selfish behaviors

B. Analysis of Simulation Results (Analysis of typical scenarios)

A typical comparison is shown in Figure 6. The parameters of trust model are

$V_T = 0.5$

$(\beta_+, \beta_-) = (0.01, 0.01).i.e. N_+ = N_- = 70$

$PTimeout = 5s$

$BNTimeout = 2000s$

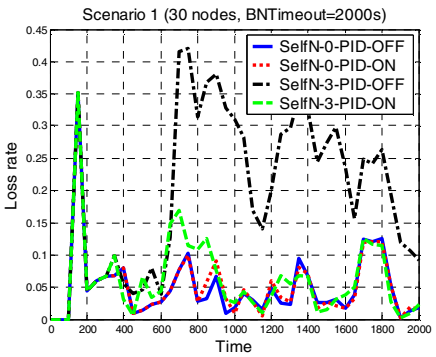
(Do not give the black node any chance to recover)

$$V_i = 0.6 \; i.e. \; N_i = \left\lceil (-\frac{\ln V_T}{\beta_-}) - (-\frac{\ln V_i}{\beta_-}) \right\rceil = 18$$

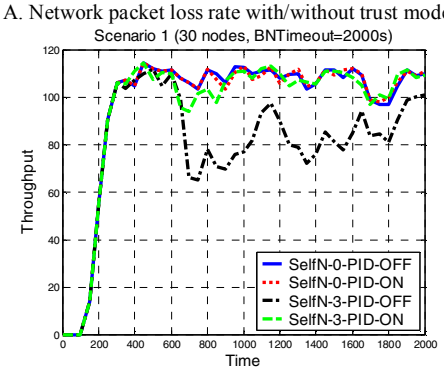
Where N_i refers to how many packets will be lost if trust value decreases from V_i to V_T continuously.

In this simulation, just like [16], four scenarios are compared: (1) the system that does not utilize trust model and no attackers (2) the system with trust model and no attackers (3) the system with 3 attackers and no trust model (4) the system that utilizes trust model with 3 selfish nodes. We can see that utilizing trust model nearly do not influence the performance of mesh networks when there are not any selfish nodes. When the trust model is disabled, the selfish nodes can make the loss rate very high and the corresponding throughput jumping down. (One node begins discarding the data relayed for others at 300s and the other two selfish nodes do that at 500s). Because every node sends packets in a constant bit rate, the loss rate has an exact relationship with the corresponding throughput. So only the loss rate will be talked about in the following discussion.

When all nodes turn on the trust model, at the beginning phase, the selfish nodes discard data transmitted for others, and

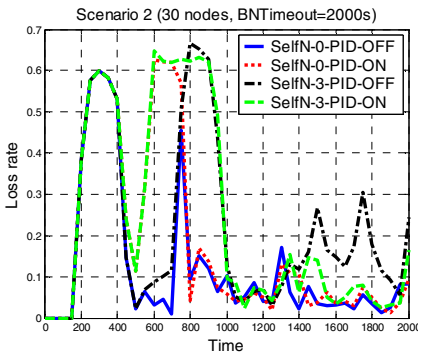


A. Network packet loss rate with/without trust model

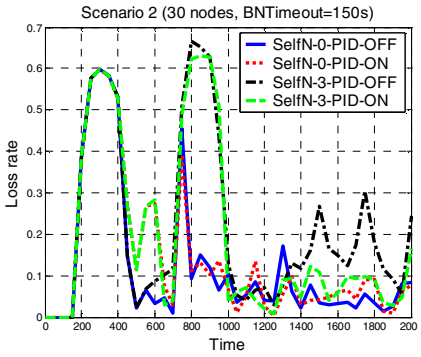


B. Network throughput with/without trust model

Figure 6. Simulation results of typical scenario 1



A. Network loss rate with/without trust model with *BNTimeout*=2000s



B. Network loss rate with/without trust model with *BNTimeout*=150s

Figure 7. Performance comparison between different configurations of parameter *BNTimeout* in typical scenario 2

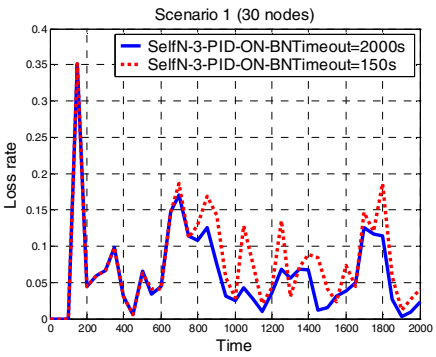


Figure 8. Performance comparison between different configurations of parameter *BNTimeout* in scenario 1

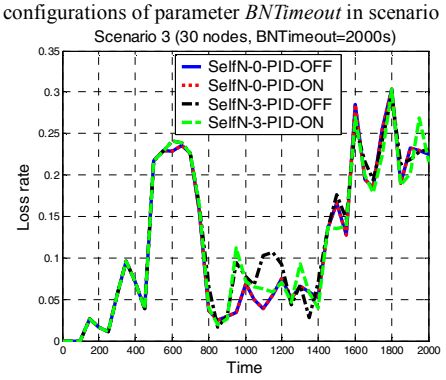


Figure 9. Network loss rate with/without trust model in scenario 3

the loss rate is the same with the situation without trust model. About tens of seconds later, the other nodes find out the selfish nodes, exclude them from their own neighbors, and do not forward data for them, which make the loss rate getting back to the level without selfish nodes. Because this operation must invoke the increase of loss rate of the selfish nodes, we only calculate the loss rate of connections with normal nodes.

Another typical scenario is shown in Figure 7. We can see that the loss rate with trust model is worse than that without trust model. We adjust all kinds of parameters to remove this exception and find out the reason. There must be some nodes added into the black node list by others wrongly, and at the same time, these nodes have no any other path to send their data to the destination. This would happen when the nodes have asymmetric information about the network topology or the relaying process is often heavily interfered. In order to deal with this problem, *BNTimeout* is set to 150s, which means when the time of node in black list is beyond 150s, it will be removed from the black list and reconsidered as a normal node. This mechanism gives the black node a chance to resume and decrease false positive.

From Figure 7B we can see that this trust model nearly does not bring any negative influence with this improvement. But it also does not reduce the loss rate aroused by the selfish nodes. The reason must be that the selfish nodes are the exclusive and key nodes in most of the connections. Under that situation, whatever mechanisms will not work. Figure 8 compares the results between *BNTimeout*=2000s and *BNTimeout*=150s in the last scenario. This improvement only brings negligible loss rate.

Because every node depends on whether the neighbor relays data for it to determine whether the neighbor is selfish, when the selfish nodes just stay at the edge of mesh network all the time, that will be impossible to transmit data for others, and the selfish node also do not harm this mesh network. Therefore, in these scenarios the mesh network can not acquire any benefits from this mechanism; simultaneously that nearly does not bring any negative influence. Just like Figure 9. We do not talk about these scenarios in detail.

IV. ANALYSIS OF PARAMETERS

In this trust model, there are six parameters to setting: V_T, (β_+, β_-), (N_+, N_-), *PTimeout*, *BNTimeout* and V_i.

The former three parameters have a strict relationship, and the purpose is to set the speed parameter N of trust model. Generally, V_T=0.5 is set. Then β can be calculated with the selected N. Actually, the N is based on how many packets discarded by the selfish nodes that can be put up with. The N_+ is the increase speed of trust value while N_- is the negative one.

PTimeout, which actually is related to the specific detection mean and not the nuclear parameter of this trust model, is the maximum time for relaying a packet (MTR) by the neighbor. *PTimeout*=5s is set here, and that's enough to be set at most

10s. In our simulations, the maximum delay of packets is tens even hundreds of seconds, and that will be impossible in the real mesh networks. If this parameter is assigned less than MTR, some nodes may be treated falsely. How many seconds *PTimeout* will be assigned needs to be tested in the practical mesh networks.

BNTimeout should be at least *BNTimeout*>3*N_i so as to give the selfish nodes enough punishment. But it should be not too large in case of high ratio of false positive.

Initial Trust Value V_i can be designated arbitrarily in theory. But the ratio of false negative will be higher especially in a high load network if the V_i is relative little. On the contrary, V_i can be set a higher value, such as 0.9 or 1, to decrease the ratio of false negative due to rapid response feature of this trust model, but the loss rate may be higher. Therefore, this is a trade off between false positive and loss rate. Consequently, a higher V_i should be configured when the node becomes neighbor in the first time, and a lower V_i after that node is removed from the black node list.

From these analyses, we can see that N is the key parameter and the configuration of parameters is very simple. That depends on your application scenario. In the simulation environment, the recommended setting of parameters is as follows:

Parameters	Values
V_T	0.5
$(\beta_+, \beta_-)\&(N_+, N_-)$	(0.01, 0.01)&(70, 70)
PTimeout	5s
BNTimeout	150s
V_i (N_i)	0.6 (19)

Table 1. Recommended parameters

V. CONCLUSION AND FUTURE WORK

In this paper we introduce a simplified PID like trust model, which is a punishment based mechanism to strengthen cooperation. This mechanism leverages the rapid judgment of selfish behaviors and mistaking the nodes. Because of the drawbacks of exchanging the trust value among nodes, we do not combine this mechanism with our trust model. According to the analysis and simulation results, we conclude that this trust model can reinforce the cooperation effectively and nearly does not produce any adverse effects to the mesh networks with appropriate parameters. By excluding the selfish nodes from the mesh network the loss rate was depressed obviously. Moreover, all over the parameters have explicit implications, and the configuration of parameters is very simple.

There are still some works we need to do in the future, such as enhancing the robustness of this trust model, exploring more new measures to detect whether or not the other nodes are selfish and malicious and combining this trust model with routing protocol and so on. One more idea is to integrate this trust model with other security mechanism. A better security mechanism should be a combination of proactive and reactive mechanism. Last but not least, this trust model including what we mentioned in section I needs to be validated in the practical applications.

ACKNOWLEDGMENT

The work is supported by NEC Laboratories China.

REFERENCES

[1] E. Huang, J. Crowcroft, and I. Wassell, “Rethinking incentives for mobile ad hoc networks,” Proceedings of the ACM SIGCOMM Workshop on Practice and Theory of Incentives in Networked Systems, Portland, Oregon, USA,.pp. 191-196, September 2004.

[2] S. Buchegger and J. Le Boudec, "Cooperative routing in mobile ad-hoc networks: Current efforts against malice and selfishness," In Lecture Notes on Informatics, Mobile Internet Workshop, Informatik 2002, Dortmund, Germany, October 2002.

[3] P. Yi, Y. Jiang, S. Zhang, and Y. Zhong, "A survey of security for mobile ad hoc networks," Acta Electronica Sinica, Vol. 33, No. 5, May 2005.

[4] L. Zhou and Z. J. Haas, "Securing ad hoc networks," IEEE Networks Special Issue on Network Security, 13, No. 6, pp. 24-30, November/December 1999.

[5] S. Yi and R. Kravets, "MOCA: Mobile certificate authority for wireless ad hoc networks," Proceedings of 2nd Annual PKI Research Workshop Program (PKI 03), Gaithersburg, Maryland, USA, pp. 65-79, April 2003.

[6] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad-hoc networks," IEEE 9th International Conference on Network Protocols (ICNP' 01), Riverside, California, pp. 251-260, November 2001.

[7] S. Buchegger and J.-Y. Le Boudec, "A robust reputation system for mobile ad-hoc networks," Technical Report IC/2003/50, EPFL-IC-LCA, 2003.

[8] B. Schneier, "Secrets and lies: Digital security in a networked world," J. Wiley & Sons, Inc, 1 edition, 2000.

[9] M. Srivatsa, L. Xiong, and L. Liu, "TrustGuard: Countering vulnerabilities in reputation management for decentralized overlay networks," Proceedings of the 14th International Conference on World Wide Web, May 2005.

[10] P. Michiardi and R. Molva, "Simulation based analysis of security exposures in mobile ad hoc networks," Proceedings of European Wireless Conference, Firenze, Italy, February 2002.

[11] L. Buttyan and J. P. Hubaux, "Enforcing service availability in mobile ad hoc WANs," Proceedings of the IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC), Boston, MA, USA, pp. 87-96, August 2000.

[12] L. Buttyán and J-P Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," Mobile Networks and Applications, pp. 579-592, August 2003.

[13] S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," Proceedings of the Sixth International Conference on Mobile Computing and Networking (Mobicom 2000), Boston, pp. 255-265, August 2000.

[14] S. Buchegger, J-Y Le Boudec, "Performance analysis of the CONFIDANT protocol: Cooperation of nodes fairness in distributed ad hoc networks," Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing (MobiHOC2002), EPFL Lausanne, Switzerland, pp. 226-236, June 2002.

[15] P. Michiardi and R. Molva, "CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," Sixth IFIP Conference on Security Communications and Multimedia (CMS2002), Portoroz, Slovenia, pp. 107-121, September 2002.

[16] Y. L. Sun, Z. Han, W. Yu, and K. J. Ray Liiu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks," INFOCOM 2006, Barcelona, Catalunya, Spain, pp. 23-29, April 2006.

[17] X. Li, M. R. Lyu, and J. Liu, "A trust model based routing protocol for secure ad hoc networks," Proceedings of IEEE Aerospace Conference, Vol. 2, March 2004.

[18] RFC3626. http://www.ietf.org.

[19] H. Ozbay, "Introduction to feedback control theory," CRC Press Inc.