ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol.3, No. 1, 2008, *pp*. 39-48

# Extended Watchdog Mechanism for Wireless Sensor Networks

Lei Huang [+], Lixiang Liu

Institute of Software, Chinese Academy of Sciences

(*Received February 28 2007, accepted December 20 2007*)

**Abstract.** Watchdog is a kind of behavior monitoring mechanism which is the base of many trust systems in ad hoc and wireless sensor networks. Current watchdog mechanism only evaluates its next-hop's behavior and propagates the evaluation result to other nodes by broadcasting, which is neither energy efficient nor attack resilient. An extended watchdog mechanism is proposed in this paper. Besides the next-hop, node with extended watchdog will monitor all its neighbors' behavior on the base of information collected from MAC layer. By overhearing the CTS packets, subsequent RTS or data packets and intervals between them, a node can judge whether its neighbor forwards the packet or not despite the location of the neighbor. The misbehaved node is punished by restricting its injected packets. Analysis and simulations on NS2 prove the effectiveness of extended watchdog mechanism. Its low communication overhead (about 1.76%) and attack resilience resulted from its fully distributed nature make it a competent solution in constructing a secure wireless sensor network.

**Keywords:** sensor network, watchdog, misbehavior detection.

## 1. Introduction

Due to their severe resource constraints, nodes in wireless sensor networks are susceptible to attacks. Compromised nodes may drop packets or inject false packets. And there is another category of nodes which behaves the same way: selfish nodes. Misbehaviors of these insiders are hard to detect and prevent since they have legitimate keys.

To mitigate the misbehavior of the non-cooperative insiders, two sub-problems must be solved: 1) Paths from a legitimate node to the sink should not go through a malicious or selfish node to prevent the legitimate packets from being dropped. 2) Excessive packets originated by malicious or selfish nodes should not be forwarded to the sink.

Watchdog mechanism proposed in [1] is a monitoring method used widely in ad hoc and sensor networks, and it is the base of a majority of misbehavior detection algorithms and trust or reputation systems. The basic idea of watchdog is that a node monitors whether its next-hop neighbor forwards the packets it just sent by overhearing. If the packet is not forwarded within a certain period, the neighbor is regarded as misbehaving in this transaction. When the misbehaving rate surpasses certain threshold, source is notified and following packets will be forwarded along other route. Then sub-problem 1 is solved.

Current watchdog mechanism can only judge the behavior of its next-hop neighbors, not last-hops. If the data flows are evenly distributed and node behaves consistently, it can make the judgment of its last-hop with reverse flows. But in most WSNs, the dominant traffic pattern is destined to sink. The amount of reverse traffic is much smaller. Node has little information about its last-hop's fidelity and it can't decide whether to forward packets for last-hop or not.

Although literatures proposed some propagating mechanisms to make node aware of last-hop's past behaviors, exchange of trust information also introduces other disadvantages such as communication overhead and potential attacks.

This paper proposes a new direct last-hop neighbor behavior evaluation mechanism (LHDA), which is based on the information from MAC layer when RTS/CTS/DATA/ACK control packets are enabled. Last-hop neighbor's forwarding behavior can be judged by overhearing its CTS packet and following outgoing

[+] Corresponding author. *E-mail address*: huangleiff@yahoo.com.cn

data packet. The node is regarded as misbehaving if it drops a received packet instead of forwarding it. The packets injected to WSN by such nodes should be restricted.

Every node maintains a table of counters for every upstream neighbor. If it finds the upstream node forwards a packet to itself successfully, the counter will be increased by 1+α. If it finds the upstream node drops a packet, the counter for it is decreased by β. α and β are reward and punishment parameters respectively. Whenever a packet is forwarded, the counter of that neighbor is decreased by 1.Only when the counter is greater than 0 will the node forward the packet for the neighbor. The method can effectively decrease the number of injected false data and selfish data. Extended watchdog mechanism is made up of LHDA and traditional watchdog algorithm. It can reduce the influence of malicious node to the lowest degree.

The paper is organized as follows: Related work is briefly discussed in section 2. Section 3 specifies our system model and assumption. Section 4 describes the extended watch-dog mechanism. Its characteristics are analyzed in section 5. Section 6 presents the simulation setup and its results. In section 7, we conclude our paper and discuss the future work.

## 2. Related Work

To ensure the reliability of packet delivery, trust for ad hoc and sensor networks has been investigated in a lot of literatures. The foundation of such dynamic trust system is the node behavior monitoring mechanism. There are many different methods proposed so far, such as data consistency checking in [2], payment variance inference in [3], en-route probing in [4], etc. But the most frequently used one is the watchdog mechanism proposed in [1] and its variations.

The main idea of watchdog in [1] was overhearing. When a node sent a packet to its neighbor, it also cached one locally. Then the node listened to its neighbor's communication. If the neighbor didn't forward the same packet to its next-hop node within a period, it was regarded as misbehaving. By this way, a node could record the successful and failed forwarding history of its next-hop.

On the base of watchdog, various judging algorithms and subsequent handling mechanisms are proposed. [1] judged a node to be misbehaving when failure tally exceeded certain threshold bandwidth and it would send a packet backward to notify the source. Then the source would choose a new route free of misbehaving node with the aid of pathrater.

[5] proposed to measure the next-hop's behavior with the local evaluation record which was defined as a 2-tuple: packet ratio and byte ratio. They were ratios of the packets successfully forwarded by the next-hop to the packets the node sent to this neighbor. Local evaluation records would be broadcast to all neighbors. The trust level of a node was the combination of its local observation and the broadcasted information. Trust level was inserted to the RREQ. Route was selected in the similar way to AODV.

Although many ad hoc trust or reputation systems [6-8] adopted different trust level calculation mechanism, the basic processes were similar to [5], including monitoring, broadcasting local observation, combing the direct and indirect information into the final trust level. The broadcast process not only incurred communication overhead, but also brought risks to the trust or reputation systems. To avoid false accusations, some literatures [6][8] proposed a ageing or timeout mechanism to recover the revoked node which was once below certain trust threshold. But the aging weight [6] and the timeout threshold [8] have not been specified.

## 3. System model

The exte`nded watchdog mechanism is designed for the wireless sensor networks where RTS/CTS/DATA/ACK is enabled in MAC layer, such as S-MAC[13] or 802.11DCF. Although the control sequence introduces some overhead, it is energy-efficient in the cases when the volume of data is big, for example, reports of image sensors. Even in the cases of small data packet, the time duration field in the control packet can also help to schedule sleep to save energy, as the coordinated adaptive sleeping schedule proposed in [13].

To make the description clearer, we will focus on a segment of a path from source to sink, as shown in fig 1, where A forwards the packet to B and B to C. SRC is the source of the packet and SINK is the destination. D is a neighbor of B, but it is not on the route of current packet. $N_i$ is B's other downstream neighbors, i=1…n. C can also be represented by $N_0$.

We will make following assumptions:

- We assume that nodes in WSN are stationary after deployment. The communications between two nodes are symmetric and the node can overhear all its neighbors' communications.
- We assume that whenever a packet is received, it should be forwarded to sink within certain period.
- We assume a hop-by-hop unicast routing is adopted in this model.

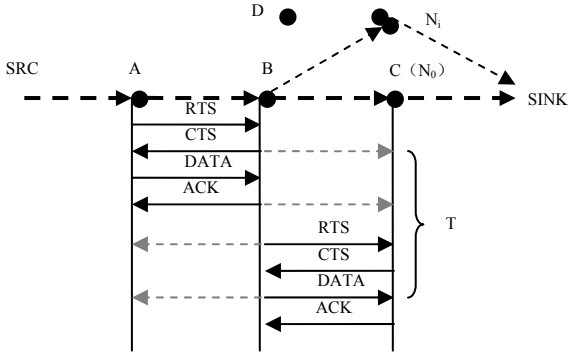



Fig. 1: System Model

# 4. Extended watchdog mechanism

## 4.1. Extended monitoring

From fig 1, B's communications can be overheard by A, C, D, and $N_i$. But in traditional watchdog mechanism, A is the only observer of B. If C and D want to know B's behavior in current forwarding transaction, they must learn it from A. For example, A broadcast its observation to D as in [5-8]. There are two disadvantages of this method: 1) It incurs a lot of communication overhead. 2) It gives A a chance to propagate false observations which may make the honest B distrusted by other nodes.

Actually, C and D can monitor B's forwarding behavior themselves. With control packets enabled, C and D can know when A has sent a packet to B although they cannot hear A. Since whenever B tries to receive a packet from A, it must send out CTS first. This can be heard by all its neighbors. If B behaved normally, it should forward the packet to C within certain period, which can also be heard by B's neighbors. If the packet is not sent out after that period, B must have dropped it. So C and D can get a direct observation of B's every forwarding behavior, and there is no need for A to broadcast.

But there are four problems in above judgment criteria. First, although C can hear the CTS packet, it can't ascertain whether the source of the CTS is B or other C's neighbors, because there is no source field in current 802.11 CTS packet. To solve this problem, we add a source field to CTS packet .The length of source field is the same as that of destination field.

Second, if the destination of the packets A forwards to B is B itself, B will not send out any data packet. B's neighbor will think that B has dropped the packet. In WSN, collected data must be sent to the sink. The packets destined to intermediate nodes are mostly up-layer control packets, such as routing messages. Since in most cases, the length of control packet is different from that of data packet, we can differentiate them from the duration field in CTS packet. Only data packet is used in judging last-hop's behavior.

Third, if A is a malicious node, it can make C or $N_i$ to mistake B for misbehaving by not sending DATA after it received CTS from B. So B must monitor A's behavior. If it always fails to send DATA, B will refuse to send CTS to it after its RTS packet received. Then A can't mislead the judgment of C or $N_i$ towards B.

The forth problem is how to differentiate legitimate dropping caused by congestion and collision from malicious dropping? Basically, the extended monitoring can not differentiate them. But the major difference between them is that legitimate dropping occurs randomly while malicious dropping will last for long time at the same level. With the algorithm in next section, mistakes caused by random dropping can be recovered after some time.

## 4.2. Last-hop malicious node detection and avoidance

When a node's dropping rate reaches a threshold, it's regarded as malicious. A traditional method to deal with malicious node is to exclude it from the network and recover it after certain timeout as in [8]. But the timeout mechanism was not specified in [8]. We think the node should be recovered after it has been

punished enough for its dropping behavior.

We design a last-hop malicious node detection and avoidance (LHDA) algorithm. Every node maintains a local counter for every neighbor, which is the up-limit number of the packets the node will forward for that neighbor. Whenever the node forwards a packet successfully which is received from a neighbor, the counter for that neighbor is decreased by 1.When the counter is decreased to zero, the node will refuse to receive packets from that neighbor by not sending CTS packets after receiving RTS from that neighbor.

The counter is made up of two parts: $C_o$, which represents the number of packets originated by the neighbor, and $C_f$, which means the number of packets forwarded by the neighbor. The former part is closely related to WSN application. For periodical reporting and query-based applications, $C_o$ will be refreshed with the predefined reporting speed $S$ every time interval. For event driven application, the precise speed can not be determined. But we can estimate an up-limit of it.

The initial value of $C_f$ is set to 0. If C receives a packet from B within certain period $T$ after it hears B's CTS packet, B's $C_f$ will be increased by $1+\alpha$. $\alpha$ is a reward parameter for B's forwarding behavior. If C hears that the B forwards a packet to a node other than itself within the period, B's $C_f$ remains unchanged. If C finds B fails to forward a packet, which means B's misbehavior, B's $C_f$ is decreased by a punishment parameter $\beta$. The larger $\beta$ is, the fewer packets will be forwarded as the punishment for dropping packets.

The algorithm is list in table 1.

Table 1 Algorithm of last-hop malicious node detection and avoidance(LHDA)

```
On receiving a packet P
{
  if ( MAC_ta(P) falls into the last-hop set of this node){
  //MAC_ta(P) is the transmitter address of P.
   if(typeof(P)==CTS){
            if(MAC_ra(P)!=this node){
            //MAC_ra() is receiver address.
                  Num_CTS[MAC_ta(P)]++;
                  CTS_Timer[MAC_ta(P)].reschedule(T);
                  //T is the predefined period
            }
            802.11_handling(P);
      } else if(typeof(P)==RTS){
            if(MAC_ra(P)==this node){
            //The RTS is destined for this node
                  if(Num_CTS[MAC_ta(P)]>0){
                        Num_CTS[MAC_ta(P)]--;
                       Cf[MAC_ta(P)]+= 1+α;
                  }
                 if(Co[MAC_ta(P)]+Cf[MAC_ta(P)]<=0)
                        does not send CTS to MAC_ta(P) ;
                  else  802.11_handling(P);
            } else {
            //The RTS is destined for other node
                  if(Num_CTS[MAC_ta(P)]>0)
                        Num_CTS[MAC_ta(P)]--;
                  802.11_handling (P);
            }
      } else if(typeof(P)==DATA){
            if(Num_CTS[MAC_ta(P)]==0)
            //The packet is originated by last-hop
                        Co[MAC_ta(P)]--;
            else if(MAC_ra(P)==this node )
                        Cf[MAC_ta(P)]--;
             802.11_handling(P);
      }else 802.11_handling(P);
  }
}
```

```
On CTS_Timer[n]. timeout
{
  Cf[n] = Cf[n]-β*Num_CTS[n];
  Num_CTS[n]=0;
}
At the beginning of each interval
{
 For(i=1;i<number of last-hop;i++) Co[i] +=S;
}
```

$C_f$ represents the forwarding history of a node. So we can adjust node's injecting speed limit $S$ according to $C_f$ if needed. If the node forwards packets successfully in most cases, which can be deduced from a high counter at the beginning of every interval, its $S$ can be increased. Otherwise, $S$ should be decreased. So, the injecting speed of leaf node will remain the same because it doesn't forward packet for others and can not be rewarded for forwarding. This policy will stimulate nodes to participate in routing and relay packets for others.

Extended watchdog mechanism (EWD) is the combination of above last-hop malicious node detection and avoidance algorithm (LHDA) and traditional watchdog algorithm (WD).

# 5. Analysis of EWD

## 5.1. Effectiveness

Then we will analysis the effectiveness of above algorithm. The following analysis is limited to within a time interval. Suppose A sends $K_a$ packets to B, among which B drops $K_d$ and forwards $K_i$ to next-hop neighbors $N_i$. Then $K_a = K_d + \sum K_i$. The packets originated by B is also forwarded by its next-hop neighbors, where $S = \sum S_i$. The packet the $i^{th}$ neighbor will forward for B is $\min(S_i+K_i,\ S_i+(1+\alpha)\cdot K_i-\beta\cdot K_d)$. When B works normally, i.e, $K_d \approx 0$, the packets forwards by next-hop neighbors is $S+K_a$. If B is a malicious node, the number of B's packets forwarded is $\min(S+K_a-K_d,\ S+(1+\alpha)\cdot K_a-(1+\alpha+(n+1)\cdot\beta)K_d)$. Node's dropping behavior is punished by restricting its injected packet.

If B is smart and it just modifies the content of the packet instead of dropping them, the algorithm cannot detect it out. But with the WD proposed in [1], its modification can be easily detected by B's last-hop node A, which will choose another node as its next-hop instead of B. So, after several rounds, $K_a=0$, and the up limit of false packets B can inject to the network is $S$.

## 5.2. Recovery from mistakes caused by legitimate dropping

Sometimes, packets are dropped because of congestion or collision. The major difference between legitimate dropping and malicious dropping is that the former occurs randomly while the latter will last for long time at the same level. EWD can recover from the random mistakes after some time.

Suppose that when B's threshold in A is surpassed, there are $m$ packets being dropped or blocked due to congestion or collision. A will stop to forward packet to B as the result of the traditional WD mechanism. The counters for B in B's all next-hop neighbors are decreased by $m\cdot\beta$. Suppose that there are $q$ packets being blocked in B's queue, $q<m$ and B has successfully forward $f_i$ packets to the next-hops neighbors before the moment. Then the $i^{th}$ next-hop will forward $f_i\cdot\alpha - m\cdot\beta$ packets for B. The recovery speed is closely related to $f_i$ .So with different next-hop node chosen, the time needed for recovery is different. Our analysis in following paragraph will be based on the expectation of $f_i$.

Because the congested packets exceed the forwarding threshold, they are mistaken by the neighbors as originated by B. So if $E(f_i)\cdot\alpha - m\cdot\beta > q$ , the congested packets will be forwarded successfully. Otherwise, all the blocked packets will be sent out after $t$ rounds, $E(f_i+\Delta f_i)\cdot\alpha - m\cdot\beta + t\cdot S > q$, where $\Delta f_i$ is the packets forwarded by B during these $t$ rounds. Then B is recovered from mistakes caused by legal dropping.

**Features**

The extended watchdog mechanism proposed in the paper is a totally distributed algorithm. The evaluation is based only on direct observations, and the node can get as much information as [5-8] did to

judge a node's behavior. So there is no need to exchange opinions in the neighborhood. The independent nature of extended watchdog mechanism brings a lot of advantages:

- Low communication overhead

The only additional communication overhead incurred by extended watchdog is the source field added to CTS. Take the parameters when the physical layer is using direct sequence spread spectrum (DSSS) for example [10], the communication overhead is about 1.76%, when data length is the maximum.

- Resilience to attacks

Exchange of information brings some risks to the system, making it susceptible to bad mouth attacks, sibyl attacks [11], etc. The trust or reputation mechanisms which include second-hand information must spend more resources on preventing these attacks. There is no need to do so with extended watchdog mechanism, because the node will make the judgment totally based on its direct observations.

# 6. Simulations and results

To evaluate the effectiveness of extended watchdog algorithms, we set up several experiments on the platform of NS2. A shim layer where the new algorithm was implemented was inserted between MAC and NetIF components in the structure of MobileNode. To focus on the extended watchdog mechanism, we adopted a very simple location-based routing algorithm in our experiments which would choose the neighbors nearest to the base station as the next-hop. At the same time, other potential next-hop candidates were also kept in the routing table in case that the main next-hop might fail to forward.

In the experiment network, nodes were scattered randomly in a 200m*200m topology. The base station located in (0,100). Every node sent 1 report to the sink every 5 minutes. But the malicious node would send 1 data packet every 1 minute. The RTS threshold remained the default 0. The communication range was set to be 50m. Simulation lasted for 100 minutes, i.e., 19 rounds.

## 6.1. Recovery from mistakes caused by legitimate dropping

We first evaluated the influence of congestion and collision to extended watchdog mechanism. We made all nodes send their reports within 0.2s, and set the number of nodes to 40, 50 and 60, corresponding to different degrees of collision. The throughput with and without EWD is listed in table 1

Table 2 Throughput when collision occured

| Number of nodes | | 40 | 50 | 60 |
|---|---|---|---|---|
| Packet received(No EWD) | | 713 | 901 | 616 |
| Packet received (EWD enabled) | $\alpha$=0,$\beta$=1 | 710 | 898 | 587 |
| | $\alpha$=0.5,$\beta$=1 | 724 | 905 | 621 |
| | $\alpha$=0,$\beta$=0.5 | 718 | 886 | 593 |

If there was no collision, the packet received at the base station should be the number of node multiplied the number of rounds. In the "clean" network with no EWD, the number of received packets was below the product. So there must be collisions. And the data of first row indicated that more collisions occurred with the increase of the number of nodes. From table 1, we found EWD could recover from the mistakes caused by collisions as we analyzed in last section. It might seem strange that throughput with $\alpha$=0.5,$\beta$=1 was higher than that of the "clean" case. The reason was that there was a hotspot where congestion occurred in most rounds. With EWD enabled, the route could circumvent that spot. The influence of mistakes caused by legitimate dropping was closely related to the reward and punishment parameter. The larger the reward parameter and the smaller the punishment parameter was, the less sensitive EWD would be to the legitimate dropping.

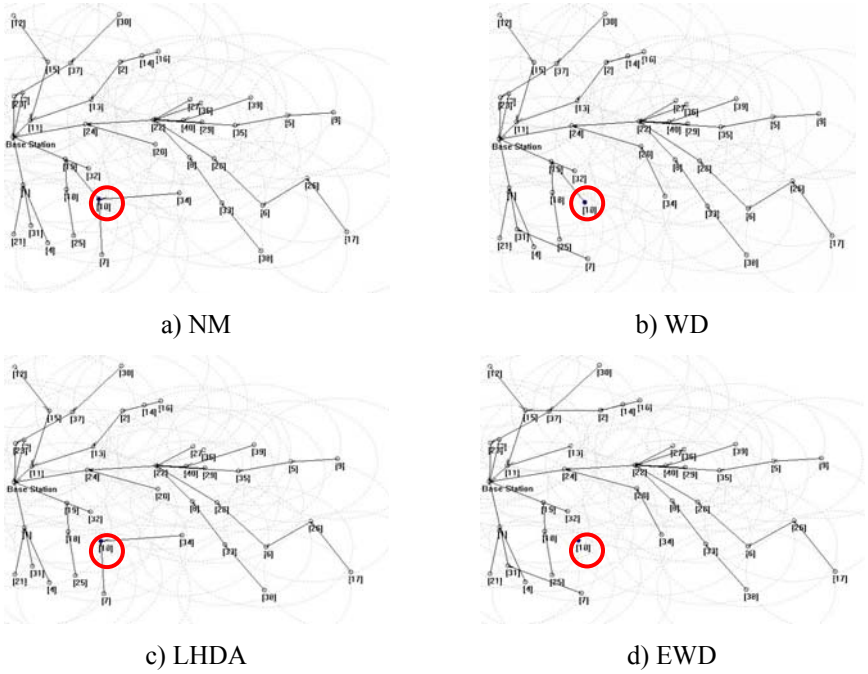
## 6.2. Effectiveness of EWD

Then we evaluated the effectiveness of EWD. We set nodes number to 40 and set node 10 as the malicious node which would drop packets forwarded to it continuously. We checked the routing graph and throughput for four cases:

No measure was taken (NM). The node couldn't differentiate the malicious node from the legitimate ones.

1. Only watchdog mechanism was adopted (WD). We implemented a simple version of WD. The node recorded its next-hop's forwarding history. If it found that its next-hop's dropping rate surpassed a threshold, it would choose another node as its next-hop whenever possible instead of notifying the source.
2. Only last-hop malicious node detection and avoidance algorithm was applied (LHDA). We would vary the reward and punishment parameter in the simulation.
3. Extended watchdog mechanism which was made up of WD and LHDA was implemented (EWD).

The resulted routing graphs, which were snapshots of round three, are depicted in figure 2.



a) NM b) WD

c) LHDA d) EWD

Fig 2 Routing Snapshots

In case 1, no measure was taken. The malicious node 10 took part in routing as normal nods. Packets forwarded to it were dropped, and the false data originated by it were sent to base station by other legitimate nodes. When watchdog mechanism was adopted, as in fig 2 b), node 7 and 34 found node 10 to be malicious and circumvented it. But false packets were still forwarded. In fig 2 c), only LHDA was enabled. Node 19 found node 10 always dropped the packet forwarded to it, and refused to relay packet for node 10. In fig 2 d), node 10's behavior was detected by both its upstream neighbor and downstream neighbor. Then it was isolated from the network. But as time went by, there was no packet sent to node 10 and its next-hop node could not judge its forwarding behavior. It was allowed to inject packet at a minimum speed.

The throughput of four cases with different reward and punishment parameter is listed in table 2.

Table 3 Throughput of different cases

| | NM | WD | LHDA | EWD | Parameters |
|---|---|---|---|---|---|
| Legitimate packets received | 655 | 703 | 639 | 682 | α=0,β=1 |
| | | | 655 | 713 | α=0.5,β=1 |
| | | | 636 | 699 | α=0,β=0.5 |
| False packets received | 95 | 95 | 0 | 15 | α=0,β=1 |
| | | | 0 | 15 | α=0.5,β=1 |
| | | | 0 | 17 | α=0,β=0.5 |

The data of table 2 validates the analysis of its routing graph. The traditional WD could prevent the

legitimate packets from being dropped. LHDA could prevent false data from being injected into the network. EWD, as the combination of both, improved the good-put while decreased the bad-put to the minimum.

Then we evaluated the mechanism in the environment of multiple independent malicious nodes. The random selected malicious node were node 1,15,28,40 in 4 malicious nodes case and 1,8,10,13,15,26,35,40 in 8 malicious nodes case. The throughput is listed in table 3.

Table 4 Throughput of cases with multiple independent malicious nodes

| | | NM | WD | LHDA | EWD |
|---|---|---|---|---|---|
| 4 malicious nodes | Legitimate packets | 501 | 578 | 499 | 578 |
| | False packets | 380 | 380 | 111 | 130 |
| 8 malicious nodes | Legitimate packets | 316 | 440 | 311 | 434 |
| | False packets | 760 | 760 | 133 | 162 |

Then we set the malicious mode from continuous dropping to selective dropping in the one selfish node case. We set the selective dropping rate of node 10 to 50%, and the threshold bandwidth of watchdog to 0.3 packets/round and 0.7 packets/round respectively, $\alpha=0.5,\beta=1$. To reflect the randomness, we prolonged the simulation to 1000s, i.e., 199 rounds. The throughput is listed in table 4.

Table 5 Throughput of cases with selective dropping mode

| | | NM | WD | LHDA | EWD |
|---|---|---|---|---|---|
| 0.3 packets /round | Legitimate packets | 7280 | 7280 | 7137 | 7221 |
| | False packets | 995 | 995 | 101 | 98 |
| 0.7 packets /round | Legitimate packets | 7280 | 7494 | 7137 | 7378 |
| | False packets | 995 | 995 | 101 | 194 |

For traditional watchdog mechanism, its effectiveness is closely related to the threshold. When the threshold is below the dropping rate, it cannot detect the malicious node. So when the threshold was set to be 0.3 packets /round, the throughput of WD was the same as that in NM case. In LHDA case, node 10 was punished for its dropping. Its injected packet was restricted. But since there were some legitimate droppings, the throughput of legitimate packets was also decreased. When the threshold is 0.7 packets /round, watchdog mechanism worked. The malicious node was excluded from the forwarding path. The throughput of legitimate packets was increased. In the EWD case, since the nodes circumvent malicious node 10 after several rounds, its next-hop node 19 could not judge its behavior. The injected false packets remained at the predefined speed, 1 packet/round.

## 7. Discussions

In above sections, we proposed a mechanism to detect misbehaviors of last–hop by extending the watchdog mechanism and evaluated its soundness with simulations. Since the scheme relies on overhearing during a time interval, it is very important to feasibility that it can coexist with sleep schedule. Current sleeping schedule can be divided into two category: coordinate sleeping schedule which calls for the cooperation of nodes and random sleeping schedule which is decided by every node independently.

We'll first analyze it coexistence with a control-sequence based coordinated adaptive sleeping scheme(S-MAC) proposed in [13].The most important components in S-MAC are: Coordinate sleeping and overhearing avoidance :

Coordinate sleeping is the major energy-saving measure in S-MAC. Nodes try to schedule their sleeping in accordance with their neighbors'. Most nodes in a neighbor observe the same sleeping schedule. Border nodes which receive multiple schedules will keep awake in the wake time of both schedules. SYNC packet is used as a coordinating message. Because SYNC here is a MAC-layer broadcasting control packet, it doesn't affect the evaluation result in our scheme which is deduced from the relations between the original control

sequences. Since nodes in a neighborhood are awake during the same period, our scheme can be carried out with no difficulty.

With overhearing avoidance enabled, node turns to sleep after it hears RTS or CTS whose destination is not itself. Such handling has no effect on LHDA whose deduction is based on RTS and CTS, but it does affect WD, because WD must listen to the data sent out by its next-hop. We can make a small modification to S-MAC that node should keep awake for a certain time after it sends out a packet.

So EWD can coexist with S-MAC while only minor modification to it. And the extra energy consumption is not large. Then we'll analyze the coexistence of EWD with random sleep schedule[14].With random sleep schedule, every node chooses its schedule independently. So when sleep cycle begins, the node may be still waiting for other nodes' control or data packet. To save energy, node should give up current listening and clear up the variables except the forwarding counter which represents the forwarding history of its neighbors. Since EWD aims to detect node's consistent behavior, the random sleep will not affect it much.

## 8. Conclusions

Node's misbehavior is harmful to the performance of WSN. Malicious dropping and excessive packet injecting is the common misbehaving mode of compromised or selfish nodes, which cannot be detected with traditional authentication method. Current watchdog mechanism can only evaluate the behavior of next-hop. With the near one-direction traffic pattern in WSN, there must be some trust propagating mechanism to tackle the problem.

Following the traditional watchdog mechanism, the extended version makes the node aware of all its neighbors' behavior when MAC control packets are enabled. Node monitors its neighbor's forwarding behavior by observing its CTS and subsequent DATA packets. When the interval exceeds certain threshold, the forwarding behavior is regarded as failed. The misbehavior is punished by restricting the injecting packet of misbehaved node. But the misbehaved node is not excluded from the WSN, instead, it can recover when the punishment is enough for its misbehavior. The extended watchdog mechanism is low energy-cost and resilient to many attacks.

So far we have been focusing on independent misbehaved node. When multiple misbehaved nodes collude, they may escape the extended watchdog mechanism. We will enhance the extended watchdog mechanism to deal with node collusion in our future work.

## 9. Reference

[1] S. Marti, T. J. Giuli, K. Lai, M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, *Proc of the 6th Annual International Conference on Mobile Computing and Networking*, 2000, 255-265.

[2] Junbeom Hur, Younho Lee, et.al., Trust Management for Resilient Wireless Sensor Networks, *LNCS 3935*, 2006, 56-68.

[3] Li Hui，Wei Wei，Chen Ke-fei, A Micro-Payment Based Isolation of Misbehavior Secure Routing for Sensor Network, *Wuhan University Journal of Natural Science*, 2005, **10**(1): 93-97.

[4] Sapon Tanachaiwiwat, Pinalkumar Dave, et.al., Location-centric Isolation of Misbehavior and Trust Routing in Energy-constrained sensor networks, *Proc of the 23rd IEEE International Performance, Computing, and Communications Conference*, 2004, 463-469.

[5] Tirhankar Ghosh, Niki Pissinou, et.al., Towards Designing a Trusted Routing Solution in Mobile Ad Hoc Networks, *Mobile Networks and Applications*, 2005, **10**( 6): 985-995.

[6] Sonja Buchegger, Jean-Yves Le Boudec , Performance analysis of the CONFIDANT protocol: Cooperation of nodes – fairness in dynamic ad hoc networks, *Proc of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2002, 226- 236.

[7] C. Zouridaki, B. L. Mark, et.al., A Quantitative Trust Establishment Framework for Reliable Data Packet Delivery in MANETs, *Proc of ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2005, 1-10.

[8] Saurabh Ganeriwal, Mani B. Srivastava, Reputation-based Framework for High Integrity Sensor Networks, *Proc of ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2004, 66-77.

[9] Tian He, John A. Stankovic, et.al.,A Spatiotemporal Communication Protocol for Wireless Sensor Networks, *IEEE Transaction on Parallel and Distributed Systems*, 2005, **16**(10): 995-1006.

[10] Alvaro A. C´ardenas, Svetlana Radosavac, et.al., Detection and Prevention of MAC Layer Misbehavior in Ad Hoc Networks, *Proc of ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2004, 17-22.

[11] James Newsome, Elaine Shi, et.al.,The Sybil Attack in Sensor Networks: Analysis & Defenses, *Proc of Third International Symposium on Information Processing in Sensor Networks*, IPSN 2004, 2004, 259-268.

[12] Jin Wook Lee, Intrusion detection in wireless embedded sensor networks, *Doctor dissertation*, Arizona State University, 2005.

[13] Wei Ye, John Heidemann, Deborah Estrin, Medium Access Control With Coordinated Adaptive Sleeping for Wireless Sensor Networks, *IEEE/ACM Transactions on Networking*, 2004, **12**(3): 493-506.

[14] Shi Gaotao, Liao Minghong, Stochastic Sleeping for Energy-Conserving in Large Wireless Sensor Networks, *Journal of Computer Research and Development*, 2006, **43**(4): 579-585.