This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2008 proceedings.

Worm Propagation Dynamics in Wireless Sensor Networks

Bo Sun*
Dept. of Computer Science
Lamar University
Beaumont, TX 77710
bsun@my.lamar.edu

Guanhua Yan
Information Sciences (CCS-3)
Los Alamos National Laboratory
Los Alamos, NM 87545
ghyan@lanl.gov

Yang Xiao*
Dept. of Computer Science
University of Alabama
Tuscaloosa, AL, 35487, USA
yangxiao@ieee.org

***Abstract*—Worms have become an emergent threat towards information confidentiality, integrity, and service availability. While playing an important role for people to interact with surrounding environments, wireless sensor networks suffer from growing security concerns posed by worms because of sensor networks' low physical security, lack of resilience and robustness of underlying operating systems, and the ever increased complexity of deployed applications.**

In this paper, we study worm propagation in 802.15.4 based wireless sensor networks. First we present a baseline worm model in the context of wireless sensor networks. Then we describe a preliminary study of the impact of various protocol parameters and network scenarios on worm propagation dynamics. Our simulation study can provide insight into deriving a suitable model to characterize worm propagation in sensor networks.

I. Introduction

Wireless Sensor Networks (WSNs) have become a promising technology which has the potential to change our everyday life. However, because of low physical security, lack of resilience and robustness of underlying operating systems [1], and the ever increased complexity of deployed applications, new system vulnerabilities keep reported about WSNs. Worms, which have become an emergent threat towards information confidentiality, integrity, and service availability, have posed imminent attacks towards normal functions of WSNs.

New security concerns about WSN worms are hard to exaggerate. The low physical security makes sensor nodes easily compromised, thus rendering embedded secrets open to attackers and various keying mechanisms helpless. Recently emerged viruses, like *Cabir* [2], have demonstrated the ability to spread over the air interfaces, making it possible to develop worms replicating between sensor nodes. Lack of resilience and robustness of underlying operating systems [1], inadequacy of basic building blocks for reliable WSN software systems [3], and the ever increased complexity of deployed applications make it difficult to develop safe applications all the time. All these make it feasible for attackers to develop self-replicated malicious codes in WSNs. While no proof-of-concept worms have caused any serious damages to WSNs so far, it is anticipated that WSNs will be plagued by worms designed in the future. This is especially true when WSNs are increasingly deployed in critical infrastructures. Once worms start spreading, manual human intervention is hardly effective based on the experience gained from past Internet worms [4].

Motivated by these concerns, we perform a simulation based study of worm propagation dynamics in 802.15.4 [5] WSNs. First we present a baseline worm model in the context of 802.15.4 WSNs. We illustrate how a compromised node can obtain an unfair bandwidth share by not adhering to 802.15.4, thus facilitating the fast spread of worms. Then we describe a preliminary study of the impact of various protocol parameters and network scenarios on worm propagation dynamics. Our simulation study can provide insight into deriving a suitable model to characterize worm propagation in WSNs.

II. An WSN Worm Model

In 802.15.4 [5], a node which has a packet ready to send first backs off for a random number of time between 0 and $2^{BE}-1$. If the channel is found to be busy again after the random backoff, BE increases by 1. This process is repeated until either BE equals *aMaxBE*, at which point BE is frozen at *aMaxBE*, or until a certain maximum number of permitted random backoff stages, denoted as *macMaxCSMABackoffs*, is reached, at which point an access failure is declared.

In this section, we present the design of a baseline WSN worm. We assume that one sensor node is compromised by a worm and this infected sensor node attempts to replicate this worm to those sensor nodes within its transmission range. We focus on Medium Access Control (MAC) layer propagation dynamics and thus ignore infecting sensor nodes multiple hops away. The single-hop worm propagation in WSNs represents one of the key differences from Internet worms.

We focus on the worm study in an *nonbeacon-enabled* mode and illustrate how a compromised node can obtain an unfair bandwidth share by not adhering to 802.15.4, thus facilitating the fast spread of worms. Our proposed worm model can also be applied to *beacon-enabled* mode after slight modifications.

After a worm has established a presence, one of its basic goals is to spread quickly to other vulnerable nodes. Therefore, an infected node may use selfish strategies to obtain an unfair share of the channel. Taking IEEE 802.15.4 unslotted CSMA-CA mechanisms as an example, this may include:

*This research was supported in part by the Texas Advanced Research Program under grant 003581-0006-2006 and the US National Science Foundation (NSF) under grants DUE-0633445, CNS-0716211, and CNS-0737325.

978-1-4244-2075-9/08/$25.00 ©2008 IEEE

- Selecting a smaller average backoff value, as specified by the parameter *wormBE*,
- Adopting a different retransmission strategy that does not increase BE after collision,
- Increasing the maximum possible value for NB, *worm_macMaxCSMABackoff*, so that a node may have more chances to compete for channels.

The modified unslotted CSMA-CA for one possible approach of propagation of WSN worms is illustrated in Fig. 1.

Depending on application requirements, we propose two types of worm models: *unicast* WSN worm model and *broadcast* WSN worm model. For both types of worms, when a WSN worm is activated, the worm starts spreading itself in its vicinity. In the unicast worm model, the infected sensor node can only unicast the worm to one of its neighbors at a time. For example, a neighbor discovery protocol enables the compromised node A to keep the list of single-hop neighbors. When node A wants to spread infection, A picks up one neighbor, constructs the worm, and sends it out. In the broadcast worm model, the infected sensor node A just constructs a broadcast packet, and send it out.

Extensive research work has been devoted to pairwise key establishment mechanisms in WSNs. Communications between nodes are secured by keying mechanisms. A unicast model is therefore necessary in order for a worm to spread in this environment. For applications which are protected by group key schemes, a broadcast model can be used.

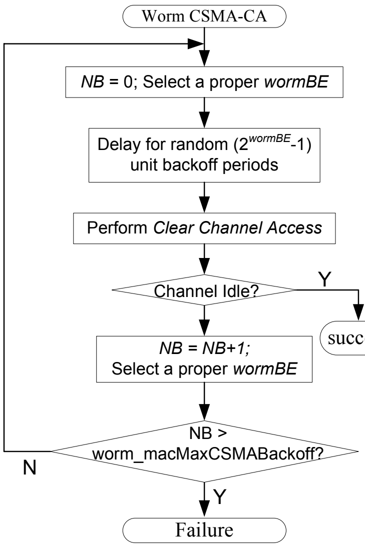
Worm CSMA-CA
NB = 0; Select a proper *wormBE*
Delay for random (2^{wormBE}-1) unit backoff periods
Perform *Clear Channel Access*
Channel Idle?
Y
success
NB = NB+1; Select a proper *wormBE*
NB > worm_macMaxCSMABackoff?
N
Y
Failure

Fig. 1. Unslotted CSMA-CA for WSN Worms in 802.15.4 *nonbeacon-enabled* mode.

Each WSN node can be in one of the following three *states*:

- *Invulnerable*: An *invulnerable* sensor node A is not vulnerable to propagated worms. When A receives a worm packet, A silently drops the data packet that contains the worm code.
- *Vulnerable and Infected*: An *vulnerable and infected* sensor node A is vulnerable and has been infected with propagated worms. When node A receives a worm packet from node B, node A may be infected and send back an INFECTED packet to node B, informing node B that node A has already been infected. In this case, node B can update its neighbor information correspondingly.
- *Vulerable and Uninfected*: An *vulerable and uninfected* sensor node A is vulnerable to propagated worms and has not been infected. When node A receives a worm packet from node B, node A sends back a SUCCESS packet to node B, informing node B that node A is now infected with the worm.

This is illustrated in Fig. 2.

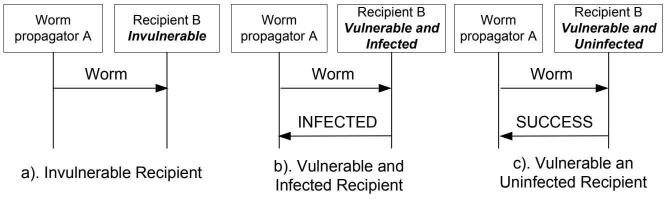
Worm propagator A
Recipient B *Invulnerable*
Worm
a). Invulnerable Recipient
Worm propagator A
Recipient B *Vulnerable and Infected*
Worm
INFECTED
b). Vulnerable and Infected Recipient
Worm propagator A
Recipient B *Vulnerable and Uninfected*
Worm
SUCCESS
c). Vulnerable an Uninfected Recipient

Fig. 2. Neighbor Feedback.

A. Unicast Worm Model

In the unicast worm model, we assume that a neighbor discoverery protocol is used for one node to discover its one-hop neighbors. This is a reasonable assumption given that most applications and upper layer protocols in WSNs require neighbor information. Therefore, each node can collect a list of neighbors within its radio range. Each node may start replicating worms based on the state of its neighbors.

For situations illustrated in Fig. 2b and Fig. 2c, on the arrival of an INFECTED or SUCCESS response packet, node A removes the victim node B from the neighbor list and attempts to infect the next one.

The sending node A may also need to set up a timer T_U, which expires after T_U time units if no response is received. In situations illustrated in Fig. 2a, node A may not receive any feedback from node B. The worm packet and feedback packet from node B may also be lost in Fig. 2b and Fig. 2c. When T_U expires, node A may update the information about node B in its neighbor list based on the response packet from node B. A proper T_U should give node A sufficient time to wait for the feedback from node B. If no response is received, node A may mark that node B as *invulnerable* and attempt to infect the next neighbor.

Considering all these, the infection cycle of an unicast worm is illustrated in Fig. 3.

B. Broadcast Worm Model

A corresponding modification is the broadcast worm model, in which the worm can broadcast in one hop to its neighbors. Every neighbor may generate a corresponding reply based on schemes illustrated in Fig. 2.

The broadcast worm model is illustrated in Fig. 4. In Fig. 4, timer T_B gives node A sufficient time to wait for feedback

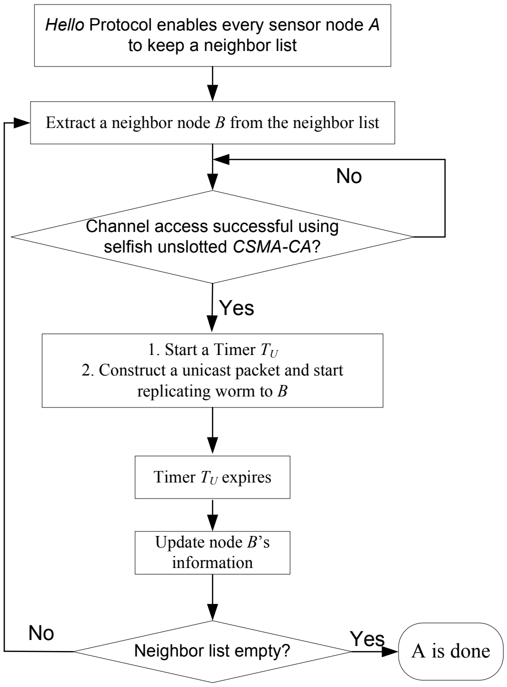
Hello Protocol enables every sensor node *A* to keep a neighbor list
Extract a neighbor node *B* from the neighbor list
No
Channel access successful using selfish unslotted *CSMA-CA*?
Yes
1. Start a Timer T_U
2. Construct a unicast packet and start replicating worm to *B*
Timer T_U expires
Update node *B*'s information
No
Neighbor list empty?
Yes
A is done

Fig. 3. Unicast Worm Infection.

from all neighbors. Based on Fig. 2, node A may modify the status of its neighbors correspondingly.

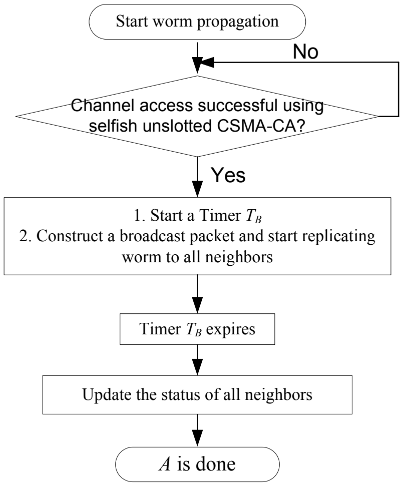
Start worm propagation
No
Channel access successful using selfish unslotted CSMA-CA?
Yes
1. Start a Timer T_B
2. Construct a broadcast packet and start replicating worm to all neighbors
Timer T_B expires
Update the status of all neighbors
A is done

Fig. 4. Broadcast Worm Infection.

Because most secure WSN applications involve the deployment of pairwise key schemes, therefore, we simulate unicast worm models and study its propagation dynamics.

III. Simulation Study

We use a simulation model based on NS2-2.31 [7]. NS2-2.31 provides an implementation of 802.15.4, based on which we implement the presented unicast worm model specified in Fig. 3.

In our simulation, we randomly pick up one node A as an *infected* node. All other nodes are set to *vulnerable and uninfected*. Node A begins worm propagation at simulation time 5sec. This allows every node to collect neighbor information. All nodes are randomly deployed in the simulation area.

For all the parameters used in simulations, we use default values recommended by 802.15.4, unless otherwise specified. The relevant parameters are *macMinBE*, which is set to 3, and *macMaxCSMABackoff*, which is set to 4.

For every set of simulation parameters, we adopt 10 different node deployments and simulate the worm propagation for 10 runs. We present the average result of these 10 runs.

The radio propagation model we use is the two-ray ground reflection. In this paper, we focus on nonbeacon model, therefore, the *beaconenabled* is set to false.

We do not discuss the impact of T_U on worm propagation dynamics. We simply set T_U to 0 and present the analysis.

A. Impact of BE

In this simulation, we use a moderate network density and simulate 100 sensor nodes in square areas with side 70 meters. The sensor nodes are randomly deployed in the square area. We change the parameter *BE* to 2, 3, 4 and depict the propagation dynamics in Fig. 5.

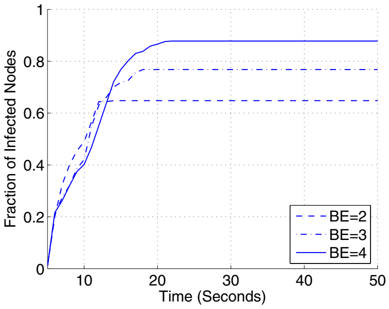

Fig. 5. Propagation Curves under Different BE. 100 nodes, 70 x 70 area.

First, we observe that given a reasonable network density, it is difficult to infect all the vulnerable nodes in the network using the model presented in Fig. 1. This is because the infected node tries to infect its neighbors as fast as possible, which leads to the interference of worm packets. Furthermore, because 802.15.4 does not employ RTS/CTS exchange in order to achieve higher channel utilization, it suffers from hidden terminal problems. This also contributes to the loss of packets.

Second, with the increase of BE, we observe that more nodes are infected. This is also what we expect. With the increase of BE, infected nodes transmitting worm packets need to backoff a relatively longer time to compete for channels. This leads to less packet interference. Therefore, worm packets have less chance of collision. This contributes to more worm infection instances.

Third, for different BEs, we observe that when BE is smaller, for example, when BE is 2, it propagates faster than other situations. This is because when BE is smaller, an infected node can obtain an unfair share of the channel, thus speeding up the spread of worms.

B. Impace of NB

In this simulation, we simulate 100 sensor nodes in square areas with side 70 meters. The sensor nodes are randomly deployed. We change the parameter NB to 1, 2, 3 and and depict the propagation dynamics in Fig. 6.

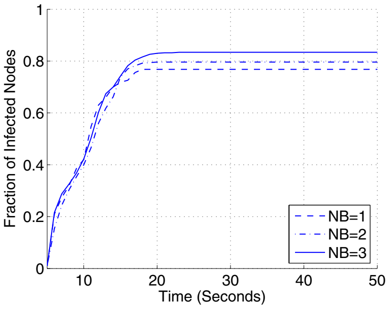

Fig. 6. Propagation Curves under Different NB.

First, we observe that when NB is larger, more nodes are infected. Based on 802.15.4, a larger NB gives an infected node more opportunities to retransmit the worm packet after sensing a busy channel. Therefore, worm packets have a higher probability to propagate.

Second, for different NB, we do not observe the difference in their propagation speed in the first phase. This is also what we expect. At first, there are not many worm packet collisions. Therefore, in the first phase of worm propagation, NB does not have dramatic impacts on worm propagation. When the number of infected nodes reaches a certain level, NB starts to demonstrate its impact on accelerating worm propagation.

C. Impact of Node Density

In this set of simulation runs, we inspect the impact of node density on worm propagation. We set NB to 1 and BE to 3, as specified in 802.15.4. We fix the square area with side length 70 meters, and vary the number of nodes by using 50 nodes, 100 nodes, and 200 nodes. We plot the fraction of infected nodes versus simulation time in Fig. 8.

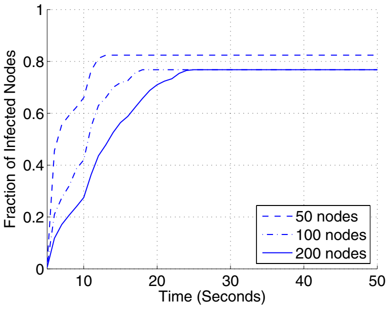

Fig. 7. Propagation Curves under Different Densities.

First, we observe that the denser the network, the slower the propagation is. As the network becomes denser, it is more likely that a large number of nodes transmit packets simultaneously and packet collisions occur.

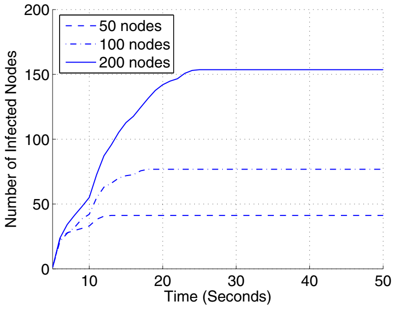

Fig. 8. Propagation Curves under Different Densities.

Second, we observe that the denser the network, the larger the number of nodes are infected. This is also what we expect.

D. Packet Loss Ratio

To further understand the impact of packet loss on worm propagation, we perform another set of simulations. In this set of simulation runs, 100 sensor nodes are randomly deployed in a 70 × 70 m^2 square area. We set NB to 1 and BE to 3 and measure MAC layers packet collisions. We disable the MAC layer packet loss functions, and plot the curve in Fig. 9. This can help us further understand the propagation dynamics illustrated in previous sections.

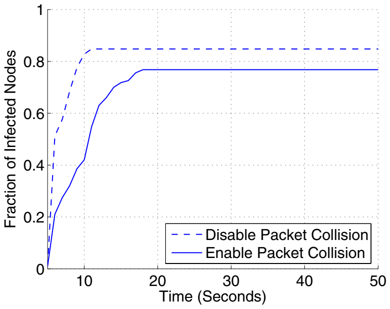

Fig. 9. Propagation Curve Considering Packet Collision.

As we expect, after we disable the packet collision functionality in the MAC layer, more nodes are compromised. However, it is still difficult to achieve 100 percent inflection. This is because we only disable packet collisions at MAC layer. The complexities of low-power wireless networking may make WSNs suffer from a variety of other reasons for packet losses, such as asymmetric loss, grey region, etc. [6].

E. Modeling Considerations

There have been many research efforts devoted to modeling Internet worms. An accurate model can provide insight for detection and defense. In this section, we answer this question: whether existing Internet worm propagation models can be directly applied for WSN worm propagation.

In [8], a logistic model is proposed as a general model to analyze the propagation of worms, such as *Code Red I*:

$$N(t) = N_{dev} * \frac{e^{K(t-T)}}{1 + e^{K(t-T)}}, \qquad (1)$$

where $N(t)$ is the number of infected sensor nodes at time t. N_{dev} is the total number of devices in the network. K and T are two parameters for the model.

Because in our case, only neighbor list information is available, an infected node can only infect its neighbors. Therefore, we set the initial point $N(5)$ to 1 (recall that the first worm instance starts spreading itself at simulation time 5) and derive the inflection point from the simulation results [9]. We assume that the model takes the same amount of time to infect half of the vulnerable population as in the simulation results. We derive K and T based on these two points. The figure illustrating the comparison between the derived logistic model and the simulated curve is plotted in Fig. 10.

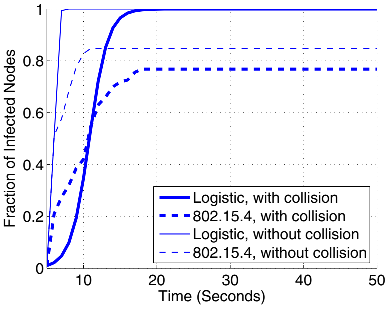

Fig. 10. Compare with Logistic Model.

It is clear that in the early phase of propagation, the logistic model underestimates the model. This is because at the early phase, the infected node may pick up a neighbor, and start replicating the worm. While in the model illustrated in Equation (1), there is an infection phase. This may slow down the worm propagation.

At the late phase, the logistic model overestimates the worm propagation. This is because the model in Equation (1) does not consider the congestion and packet loss, which can slow down worm propagation. Also, the logistic model predict a 100% infection ratio, which is not the case in our simulation.

Based on Fig. 10, one natural question is whether disabling packet losses could lead to a better model fitting. To answer this question, we disable packet collision, and plot the fraction of infected nodes and the fitted logistic curve, which is also illustrated in Fig. 10. As we can see, although both curves move left relatively, the logistic curve still overestimates the propagation speed at a late phase. At the early stage, both curves illustrate a very fast propagation.

F. Average Number of Devices Infected per Second

We plot the average number of infected nodes per second in Fig. 11. We can see that as more sensor nodes are compromised, the average number of infected nodes is decreasing. This can slow down the worm propagation. This also helps explain the difference demonstrated in Fig. 10. The model illustrated in Equation (1) assumes the same discovery ratio every time, which is not the case in our situation.

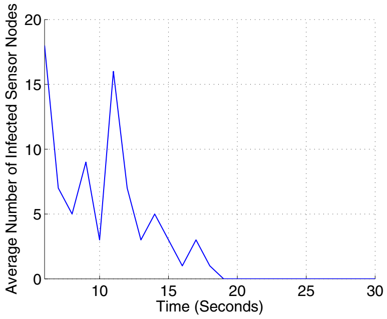

Fig. 11. Average Number of Infected Sensor Nodes per Second as Simulation Advnaces.

IV. Related Work

Yan *et al.* [9] [10] analyze the worm propagation in Bluetooth networks and investigate the impact of mobility patterns on Bluetooth worm propagation. Khayam *et al.* [11] propose a topologically-aware worm propagation model (TWPM) for WSNs. By incorporating MAC and network layer considerations, TWPM captures both time and space propagation dynamics. De *et al.* [12] model the node compromise in WSNs based on epidemic theory. However, their work is not based on 802.15.4, which is an emerging standard for WSNs.

V. Conclusions

In this paper, based on 802.15.4 standard, we present a baseline worm model for wireless sensor networks. We study the propagation dynamics of worms in WSNs and describe a preliminary study of the impact of various protocol parameters and network scenarios. Our simulation study can provide insight into deriving a suitable model to characterize worm propagation in wireless sensor networks.

References

[1] H. Kim and H. Cha, "Towards a Resilient Operating System for Wireless Sensor Networks," *USENIX Annual Tech. Conf.*, Boston, MA, June 2006, pp. 103-108.
[2] http://www.f-secure.com/v-descs/cabir.shtml
[3] J. Regehr, N. Cooprider, W. Archer, and E. Eide, "Memory Safety and Untrusted Extensions for TinyOS," *Tech Report, University of Utah*, 2005.
[4] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the Slammer Worm," *IEEE S & P*, Vol. 1, No. 4, 2003, pp. 33-39.
[5] Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). IEEE Standard, 802.15.4-2003, May 2003.
[6] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis, "Understanding the Causes of Packet Delivery Success and Failure in Dense Wireless Sensor Networks," *ACM Sensys'06*, Boulder, CO, 2006, pp. 419-420.
[7] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/index.html.
[8] S. Staniford, V. Paxson, and N. Weaver, "How to 0wn the Internet in Your Spare Time," *11th USENIX Security Symposium*, August 2002.
[9] G. Yan and S. Eidenbenz, "Bluetooth Worm: Models, Dynamics, and Defense Implications," *ACSAC'06*, Miami, FL, Dec. 2006, pp. 245-256.
[10] G. Yan and S. Eidenbenz, "Modeling Propagation Dynamics of Bluetooth Worms," *IEEE ICDCS'07*, Toronto, Canada, June 2007.
[11] S. Khayam, H. Radha, "A Topologically-Aware Worm Propagation Model for Wireless Sensor Networks," *IEEE SDCS*, 2005, pp. 210-216.
[12] P. De, Y. Liu, and S.K. Das, "Modeling Node Compromise Spread in Wireless Sensor Networks Using Epidemic Theory," International Workshop on Wireless Mobile Multimedia, 2006, pp. 237-243.