Reputation Prediction in Mobile Ad Hoc Networks Using RBF Neural Networks

Fredric M. Ham[1], Eyosias Yoseph Imana[1], Attila Ondi[2], Richard Ford[2], William Allen[2], and Matthew Reedy[1]

[1] Florida Institute of Technology, Electrical and Computer Engineering Department
[2] Florida Institute of Technology, Computer Sciences Department
{fmh,eyoseph2007,aondi,mreedy}@fit.edu,
{rford,wallen}@cs.fit.edu

Abstract. Security is one of the major challenges in the design and implementation of protocols for mobile *ad hoc* networks (MANETs). 'Cooperation for corporate well-being' is one of the major principles being followed in current research to formulate various security protocols. In such systems, nodes establish trust-based interactions based on their reputation which is determined by node activities in the past. In this paper we propose the use of a Radial Basis Function-Neural Network (RBF-NN) to estimate the reputation of nodes based on their internal attributes as opposed to their observed activity, e.g., packet traffic. This technique is conducive to prediction of the reputation of a node before it portrays any activities, for example, malicious activities that could be potentially predicted before they actually begin. This renders the technique favorable for application in trust-based MANET defense systems to enhance their performance. In this work we were able to achieve an average prediction performance of approximately 91% using an RBF-NN to predict the reputation of the nodes in the MANET.

Keywords: MANET, attribute vector, reputation, radial basis function neural network, ROC curve.

1 Introduction

A Mobile ad-hoc Network (MANET) is a collection of mobile nodes connected to each other through a wireless medium dynamically forming a network without the use of centralized control or existing infrastructure. It is this type of network that is being researched and is projected to be applied in urgent and critical situations like medical emergency, disaster relief and military combat arenas. The sensitivity of such applications and the possible lack of an alternate communications path make these networks attractive to cyber attacks [1]. According to a recent DARPA BAA [2], among all of the cyber threats that exist, one of the most severe is expected to be worms with arbitrary payload, which can saturate and infect MANETs on the order of seconds. Some examples of the mobile devices that can reside at the nodes in the MANET are workstations, sensors, handheld devices, laptop computers, etc.

Maintaining the integrity of the network and each individual node in a MANET is challenging due to the lack of central authority in the network. Preserving this integrity,

D. Palmer-Brown et al. (Eds.): EANN 2009, CCIS 43, pp. 485–494, 2009.

however, is critical for carrying out missions in a military or disaster relief scenario. Although, for example, a computer at a node can be fully tested and configured with the latest "known good" patches (and therefore assumed to be free from malicious code), this known good state might be compromised in the field when a sudden change in the mission or environment could require the users to install new applications or modify certain settings "on-the-fly."

Assuming the presence of a trusted monitoring component in the kernel of the operating system allows a reliable measurement of the state of the computer; however, in itself such a trusted component might be unable to detect the presence of certain malicious code and stop the activities. Moreover, even if the malicious code can be detected, it is often too late: by the time of the detection, the malicious code could have compromised the system in an unknown manner as well as spread to other systems. Also, detecting attacks from remote hosts is useful as those hosts can be isolated, preventing further damage. The mere fact of detecting an attack, however, does not give insight into the vulnerability that allowed the malicious compromise, thus limiting the system to reactive rather than proactive defense.

Intuitively, it should be possible for these trusted components to cooperate and share information about themselves and each other. Correlating the known state of the computers with their reputation relating to malicious activities can be used to identify vulnerable computers based on their states even before they engage in malicious activity, thereby keeping "a step ahead" of the malicious code. As we will demonstrate in this paper, although some compromise is inevitable before the system learns the vulnerable state, the overall damage can be well contained.

In this work we use a Radial-Basis Function Neural Network (RBF-NN) [3] at each node to perform the "step ahead," prediction of the node's reputation, i.e., proactive defense from malicious activities. As explained above, relying on the detection of malicious activity by other nodes or monitoring systems to assign reputations scores is not adequate since malicious code in compromised nodes might wait only a few time steps before it starts attacking other nodes [4]. Moreover, we want to predict a potential compromise before the malicious activity is well underway.

An attribute vector is computed for each network node. The vector contains 10 numeric values related to crucial status indicators of processes and physical characteristics associated with nodal activity. An RBF-NN at each node maps the attribute vector of that particular node to its reputation score so that trustworthiness of each node can be estimated earlier than it would be determined by monitoring the behavior of the node. Figure 1 illustrates a comparison between behavior monitoring systems and our proposed RBF-NN reputation prediction system. In the figure, the node becomes compromised at time step $n = n_k$ and initiates its malicious activities at $n = n_l$ ($n_l > n_k$). Behavior monitoring defense systems detect the compromise at $n=n_l+1$ while the RBF-NN predictor can detect the compromise at $n = n_k + 1$.

The remainder of this paper is organized as follows. Section 2 gives information on related work and Section 3 details how the network and the nodes are modeled in this study. Section 4 gives a brief explanation of the simulation details, including the RBF-NN predictor training. Simulation results are presented in Section 5. Practical considerations that should be considered for implementation of the system that we are proposing is presented in Section 6, and finally, Section 7 concludes the paper with suggestions for future research directions.

Time Step	RBF-NN Predictor System	Behavior Monitoring System
…		
$n = n_k$	Node becomes compromised	Node becomes compromised
$n = n_k + 1$	RBF-NN predicts change in the reputation of the node indicating potential attack	
$n = n_k + 2$	Preventive measures taken to protect the MANET	
…		
…		
…		
$n = n_l$		Node begins malicious activity
$n = n_l + 1$		Other nodes detect malicious activity and reputation of the node changes
$n = n_l + 2$		Preventive measures taken to protect the MANET from further attacks

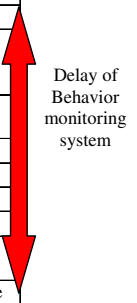

Fig. 1. Comparison of a RPF-NN predictor system and a behavior monitoring system

2 Related Work

Marti et al. [5] proposed the use of trust-based systems in the selection of next hop for routing. To ascertain the reliability of a node, the node monitors neighboring nodes to determine if it has been cooperative in forwarding other nodes' traffic. If a node has been cooperative, then it has a good reputation and its neighbors are inclined to forward packets to this node. In this way, packets are diverted (or directed away) from misbehaving nodes. Moreover, Buchegger and Le Boudec [6], [7] proposed and analyzed the CONFIDANT protocol, which detects and isolates misbehaving nodes. Similarity, they designed their protocol so that trust relationships and routing decisions are made through experience, which can be observed or reported behavior of other nodes. In [8] the authors analyzed the properties of the mechanisms that mobile nodes use to update and agree on the reputation of other mobile nodes. They suggest that mobile nodes can evaluate the reputation of other nodes based on both direct observation and the reputation propagation algorithm that they formulated in their research. The pitfalls of such systems involve relying on the detection of abnormal and possibly malicious activity to activate the necessary countermeasures, that is, operating reactively as opposed to proactively.

In Zheng et al. [9] it is discussed that malicious code is typically some type of coded computer program and under certain conditions within the computer's hardware and software it can outbreak and infect other code, compromise information and even possibly destroy certain devices within the system. These certain conditions can be modeled by defined combinations of states related to different features at the nodes. Saiman [10] lists features at the nodes that determine the reputation of the nodes within the network. They are classified as *performance metrics evaluation* features and *quantitative trust value* features. In the first category, features have different states assigned with certain numerical values. In the second category, the features are assigned values by mathematical evaluation or physical measurements.

This paper proposes a system which can be applied in any reputation-based or trust-based system. It enhances their performance by being able to determine the reputation value faster than it would be determined by simple behavior monitoring.

3 Modeling the Network

Here we present the assumptions made while building the model of the network. Computers in the network are not homogeneous, i.e. computers can have different settings. Moreover, these settings are not static, but can change as a function of time. There are n distinct settings that describe the state of any given computer in the network. Each setting i can be described by an integer value, s_i between 0 and $s_i^{max} > 1$. The particular value of each setting can be interpreted as a specific version or update of an application or hardware that is represented by that particular setting. The value 0 represents the absence of the corresponding setting (e.g. the application or hardware is not installed on the computer); and the value 1 represents a known "clean" state of the setting. Both states imply that the corresponding setting presents no vulnerability on the computer. At the start of the scenario, all settings have value 0 or 1, representing a "factory install" that is established at the beginning of the mission. All other values represent the fact that the application/hardware has been installed or modified on the computer after it has left the base and initially there exists no information about vulnerabilities that might have been introduced. Once a setting has a value other than 1, it cannot ever change back to a value of 1 (representing the fact that the state of "factory install" cannot be restored in the field).

Some pre-defined (but unknown) value of a particular setting (or a combination of such settings) represents a vulnerable state. When a computer is in a vulnerable state, it might become compromised at a later point in time by malicious code. When such a compromise happens, the machine will eventually behave in a way that is identified as malicious by its peers (i.e. it attacks its peers, causing them damage). Each machine has a reputation value between 0 (fully trusted) and 1 (untrusted) that is assigned to it by its peers. This value is initialized to 0. Once a machine begins attacking its peers, its reputation value (as derived by its peers) begins to increase, until the attack stops (for example, because of a new patch), or the value reaches 1. When a computer is no longer attacking its peers, its reputation value begins to decrease towards 0, representing the "forgiveness" of its peers towards its past behavior.

In order to model the changes in the states of the nodes, the following variables were used. The number of steps for a change on any given computer is determined by a truncated Poisson distribution with mean λ_c and maximum λ_c^{max}; for each change there is a probability, p_v, with a uniform random distribution that reflects a change that introduces a vulnerability. A computer is considered to have changed if at least one of its setting changes. If the change introduces vulnerability, a Poisson distribution with mean λ_m determines the number of time steps it takes for the malicious code to compromise the computer. A compromised computer has probability of $P_f = 0.5$ to improve its reputation (i.e., be "forgiven").

The particular choices for the parameters described above were: n=10, for all i: s_i^{max}=10, λ_c=5, λ_c^{max}=10, p_v=0.06, λ_m=3. There were 50 computers (nodes) in the network, the simulation ran for 1000 time steps, and the reputation values for each compromised machine changed by $\Delta r = 0.05$ per time step. The number of attribute settings set to 1 (as opposed to 0) at the beginning of the simulation was determined by a truncated Poisson distribution with mean 4 and maximum of 10. A heuristic approach was used to select the values of the parameters. There parameters are summarized in Table 1.

Table 1. Simulation Parameters and assumed values

Name	Description	Value
n	Number of distinct settings that describes the state of a node	10
s_i	Possible integer values assigned to setting i	0 - 10
s_i^{max}	Maximum possible value of integer to be assigned to settings	10
λ_c	Mean number of time steps before there will be a change in a computer (Poisson Distribution)	5
λ_m	The mean number of time steps it takes a malicious code to compromise the computer	4
λ_c^{max}	Maximum number of time steps before there will be a change in the computer	10
p_v	Probability that a change is to a vulnerable state	0.06
P_f	Probability that a malicious node moves from a compromised state to a secure one	0.5
Δr	The change in the reputation value of a compromised machine	0.05

4 Simulation Details

From the 50 nodes that were simulated according to the model presented in Section 3, 14 of them were compromised and displayed some level of malicious activity, i.e., their reputation fluctuated between 0 and 1. All of the remaining nodes had reputation of 0(trusted) during the simulation run. Each node had 10 attributes (settings) that can assume any integer from 0 to 10. As previously mentioned, the vector containing the numeric values that represent the "state" of the node is referred to as the attribute vector. The RBF-NN predictor is used to estimate the reputation of a node given the attribute vector at each time step. MATLAB® was used to develop the simulation. The steps involved in training and testing the RBF-NN is now explained. Detailed explanation of RBF-NNs is given in [3].

Let the matrix $\mathbf{A}_i \in Z^{p\times q}$ (a matrix of integers), where p=1000 and q=10, contain the attribute row vectors generated for i^{th} node for 1000 time steps. The vector $\mathbf{R}_i \in \Re^{p\times 1}$ (p=1000) contains the assigned reputation values for each 1000 times steps.

The following steps were followed to train, test and calibrate the neural network:

Step 1: Mean center and variance scale $\mathbf{A}_i$ [3].

Step 2: Select 60% of the rows in $\mathbf{A}_i$ to be $\mathbf{A}_{i,train}$ and 40% to be $\mathbf{A}_{i,test}$. Similarity, $\mathbf{R}_i$ was decomposed into $\mathbf{R}_{n,train}$ and $\mathbf{R}_{i,test}$.

Step 3: Train RBF-NN using $\mathbf{A}_{i,train}$ with a spread parameter of 300 (determined experimentally).

Step 4: Sort the test data such that $\mathbf{R}_{i,test}$ increases monotonically ($\mathbf{A}_{i,test}$ should be ordered accordingly). Generate the reputation vector estimate $\hat{\mathbf{R}}_{n,test}$ obtained by presenting $\mathbf{A}_{i,test}$ to the RBF-NN from step 3.

Step 5: Threshold $\mathbf{R}_{i,test}$ so that any value greater than 0.5 is rounded to 1, otherwise set to 0.

Step 6: Use a Receiver Operating Characteristic (ROC) curve [11], developed for each node and based on the simulated data to compute an optimal threshold, TR_i. Specifically, the thresholded reputation test vector $\mathbf{R}_{i,test}$ and its estimate, $\hat{\mathbf{R}}_{i,test}$, are used in the ROC curve analysis to determine the optimal threshold TR_i associated with the RBF-NN output (this is the calibration step referred to above) in step 3.

Step 7: Threshold $\hat{\mathbf{R}}_{i,test}$ such that any value greater than TR_i becomes 1, otherwise it is set to 0.

Step 8: Determine the predictive performance of the RBF-NN by comparing the thresholded vector $\hat{\mathbf{R}}_{i,test}$ from step 7 with the thresholded vector $\mathbf{R}_{i,test}$ from step 5. The neural network's predictive performance is computed as the percent of the ratio of the correct predictions to the total number of predictions.

Step 9: After performing steps 1 – 8 for all active nodes, the overall Reputation Prediction Performance (RPP) of the MANET is computed as the average of the individual performances of all the nodes.

Assuming a stationary environment, once the neural network is trained on the attributes/reputations of a particular node then a proactive approach can be taken using the RBF-NN to perform a "step ahead" prediction of the node's reputation. This significantly enhances the performance of the wireless network's defense system since it allows prediction of malicious activities before they occur.

For a non-stationary environment, training must be performed periodically "on-the-fly" to update the RBF-NN weights based on the changes in the statistical nature of the environment. The frequency of "re-training" the RBF-NN is determined by how the statistics change within the environment.

5 Simulation Results

As previously mentioned, of the 50 nodes defined in the simulation, only 14 were active in the MANET. From the simulation results, the average RPP achieved was 90.82% using an RBF-NN predictor at each node. The (i) actual node reputation, (ii) the RBF-NN reputation estimate and (iii) the thresholded reputation estimate for nodes 0 and 43 are given in Fig. 2 and 3.

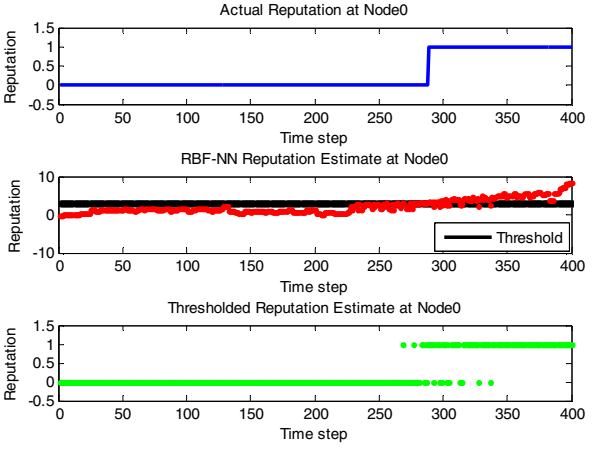

Fig. 2. RBF Estimation for Node 0

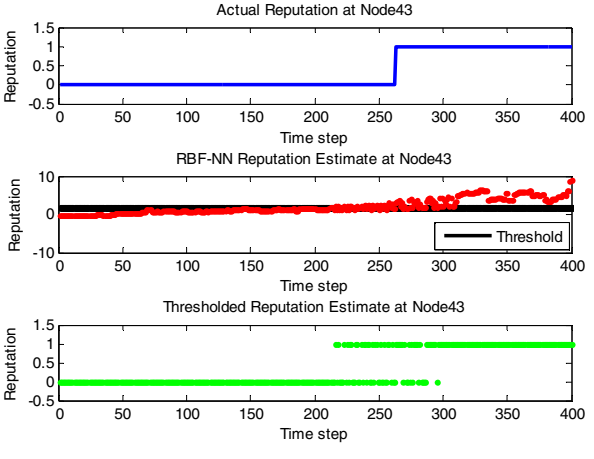

Fig. 3. RBF Estimation for Node 0

Figure 4 shows the ROC curves for nodes 0 and 43. The optimal threshold at the output of the RBF-NN for each node is determined by computing the Euclidean distance from the (0,1) point on the graph to the "knee" of the ROC curve.

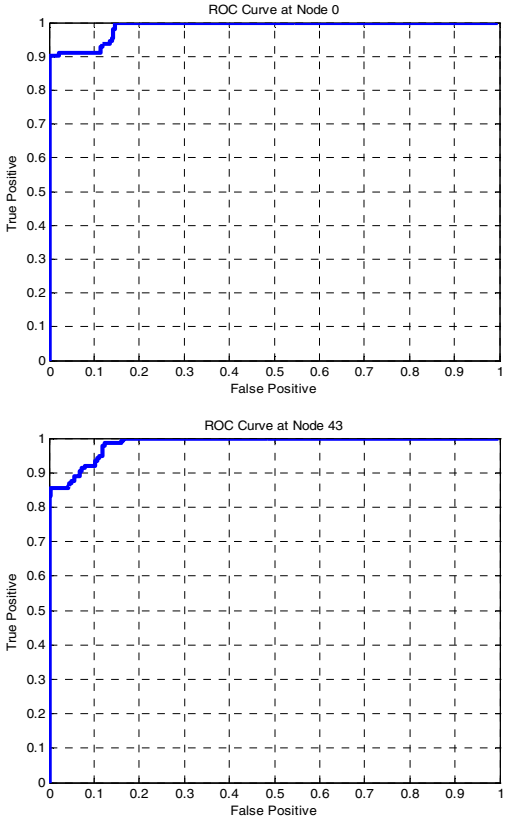

Fig. 4. ROC curves for MANET of node 0 and 43

Figure 5 illustrates the performance of the RBF-NN predictor at the "active" nodes.

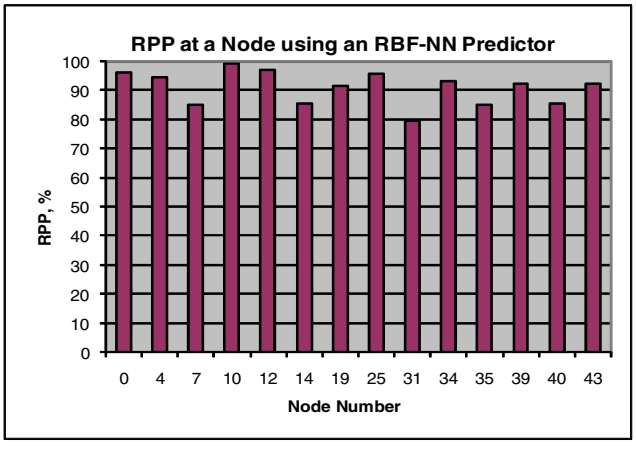

Fig. 5. RBF-NN predictive performance at each node

6 Practical Considerations

There are some practical considerations that must be considered for the successful operation of the proposed system. The first one is the existence of a trusted component in the kernel of the operating system to regularly report the state of the node. This component should be properly secured such that its functionality is not interfered with or obstructed by any software or physical compromise.

The attributes associated with a particular node are definable critical processes and physical characteristics associated with network (MANET) security. Examples are operating system patches, encryption keys, hardware configurations, exposure and location of the node [10]. The numerical assignment of these attributes should be made in such a manner to avoid singularity problems so that the RBF-NN algorithm can perform efficiently. Due to the absence of a central authority, the MANET defense system can be executed in a dynamically assigned node or even a specialized node like a cluster head.

The other issue is reputation scoring. There should be a defined benchmark known at all the nodes to facilitate the assignment of the reputation values for different types of nodal behavior (or activity). In our study, activity either results in an increase in the reputation score or a decrease. A few examples of activities that are normally expected from malicious, misbehaving or suspicious nodes are:

- Packet dropping [10]
- Flooding the MANET with a large number of Route Request (RREQ) [12]
- Sending out incorrectly addressed packets [13]
- A fake Route Reply (RREP) as in a Black Hole attack [14].

It should also be noted that the RBF-NN predictor cannot be operational before it learns the activity and state dynamics of the nodes in the MANET. Moreover, the initial RBF-NN training does not necessarily need to be performed as an integral part of the actual operation of the MANET.

As previously mentioned, a proactive approach to MANET defense as opposed to a reactive approach has the potential to better protect the MANET. For this reason, training of the RBF-NN should be carried out in a laboratory, or possibly during field testing, but in either case before the MANET is deployed for actual operation. It should be noted that the RBF-NN could be updated "on-the-fly" by re-training it adaptively once the actual operation has been initiated.

7 Conclusions and Future Work

We have demonstrated that it is possible to use an RBF-NN to proactively defend the nodes in a MANET by predicting the reputation of the nodes. Accurate prediction can allow time to perform preventative measures so that compromised nodes will not infect or disrupt other nodes in the MANET. Using the simulation approach presented in this paper, an average RPP of 90.82% was achieved using an RBF-NN predictor at each active node.

Our plans for future work include building a more realistic MANET simulator to further test the defense system suggested in this paper. In addition, we plan to investigate

the use of a Kalman filter as an N-step predictor capable of estimating the reputation of a node into the future. This would be carried out by predicting the attribute vector out to N time steps, and then using this vector estimate to predict the node's reputation.

Acknowledgments

This research is part of a multi-institutional effort, supported by the Army Research Laboratory via Cooperative Agreement No. W911NF-08-2-0023.

References

1. Abdelhafez, M., Riley, G., Cole, R.G., Phamado, N.: Modeling and Simulations of TCP MANET Worms. In: 21st International Workshop on Principles of Advanced and Distributed Simulation (PADS 2007), pp. 123–130 (2007)
2. TPOC: Ghosh, A.K.: Defense Against Cyber Attacks on Mobile ad hoc Network Systems (MANETs). In: BAA04-18 Proposer Information Pamphlet (PIP), Defense Advanced Research Projects Agency (DARPA) Advanced Technology Office (ATO) (April 2004)
3. Ham, F.M., Kostanic, I.: Principles of Neurocomputing for Science and Engineering. McGraw-Hill, New York (2001)
4. Gordon, S., Howard, F.: Antivirus Software Testing for the new Millennium. In: 23rd National Information Systems security Conference (2000)
5. Marti, S., Giuli, T., Lai, K., Baker, M.: Mitigating routing Misbehavior in Mobile Ad Hoc Networks. In: Proceedings of the Sixth International Conference on Mobile Computeing and Networking, pp. 255–265 (August 2000)
6. Buchegger, S., Le Boudec, J.Y.: Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile ad hoc Networks. In: 10th Euromicro Workshop and Parallel, Distributed and Network-Based Processing, pp. 403–410 (2002)
7. Buchegger, S., Le Boudec, J.Y.: Performance Analysis of the CONFIDANT Protocol: Cooperation of Nodes–Fairness in Dynamic ad-hoc Networks. In: Proceeding of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing, pp. 226–236 (June 2002)
8. Liu, Y., Yang, T.R.: Reputation Propagation and Agreement in Mobile Ad-Hoc Networks. IEEE Wireless Communication and Networking 3, 1510–1515 (2003)
9. Zheng, Z., Yi, L., Jian, L., Chang-xiang, S.: A New Computer Self-immune Model against Malicious Codes. In: First International Symposium on Data, Privacy and E-Commerce, pp. 456–458 (2007)
10. Samian, N., Maarof, M.A., Abd Razac, S.: Towards Identifying Features of Trust in Mobile Ad Hoc Networks. In: Second Asia International Conference on Modeling & Simulation, pp. 271–276 (2008)
11. McDonough, R.N., Whalen, A.D.: Detection of Signals in Noise, 2nd edn. Academic Press, New York (1995)
12. Balakrishnan, V., Varadharajan, V., Tupakula, U., Lucs, P.: TEAM: Trust Enhanced Security Architecture of Mobile Ad-hoc Networks. In: 15th IEEE International Conference on Networks, pp. 182–187 (2007)
13. Stopel, D., Boger, Z., Moskovitch, R., Shahar, Y., Elovici, Y.: Application of Artificial Neural Networks Techniques to Computer Worm Detection. In: International Joint Conference on Neural Networks, pp. 2362–2369 (2006)
14. Deng, H., Li, W., Agrawal, D.P.: Routing Security in Wireless Ad Hoc Networks. IEEE Communications Magazine, 70–75 (October 2002)