Ad Hoc & Sensor Wireless Networks, Vol. 16, pp. 229-242
Reprints available directly from the publisher
Photocopying permitted by license only

Autoregressive Trust Management in Wireless Ad Hoc Networks

Xu Li[1], Zhi Li[2], Milos Stojmenovic[3], Venkat Narasimhan[4] and Amiya Nayak[2]

[1]*Inria Lille - Nord Europe, France*
E-mail: xu.li@inria.fr

[2]*University of Ottawa, Ottawa, Canada*
E-mail: {zli, anayak}@site.uottawa.ca

[3]*Singidunum University, Belgrade, Serbia*
E-mail: mstojmenovic@singidunum.ac.rs

[4]*Les entreprises Norleaf Networks Inc., Gatineau, Canada*
E-mail: narasim@norleaf.ca

Received: September 11, 2011. Accepted: October 31, 2011.

In this paper, we propose a novel trust management scheme for improving routing reliability in wireless ad hoc networks. It is grounded on two classic autoregression models, namely Autoregressive (AR) model and Autoregressive with exogenous inputs (ARX) model. According to this scheme, a node periodically measures the packet forwarding ratio of its every neighbor as the trust observation about that neighbor. These measurements constitute a time series of data. The node has such a time series for each neighbor. By applying an autoregression model to these time series, it predicts the neighbors future packet forwarding ratios as their trust estimates, which in turn facilitate it to make intelligent routing decisions. With an AR model being applied, the node only uses its own observations for prediction; with an ARX model, it will also take into account recommendations from other neighbors. We evaluate the performance of the scheme when AR, ARX or a previously proposed Bayesian model is used. Simulation results indicate that the ARX model is the best choice in terms of accuracy.

Keywords: Trust management; autoregression; wireless ad hoc networks

I INTRODUCTION

A wireless ad hoc network is a dynamic peer-to-peer environment, where nodes interconnect wirelessly through multi-hop routing paths without using any centralized control or fixed infrastructure. Nodes may freely join and leave the environment. They do not have special role assignments such as “client-server”; the interaction between them are thus “peer-to-peer”. The network functionalities rely solely on nodal cooperation on message forwarding. If a node cooperates with a compromised (malicious) node that is bribed by an adversary intending to attack the network, then this malicious node may easily undermine the network performance, for example, by sending incorrect routing information or simply dropping messages. Another risk can be caused by selfish nodes [25, 26, 27]. Since nodes in the network can autonomously change their status, selfish nodes could potentially refuse to provide any cooperation in order to minimize their own power consumption or CUP usage. When malicious or selfish nodes are present, the network is at a high risk of being threatened and attacked by them. In some emerging applications, they may even cause direct damage to the physical world [10, 11]. It is necessary to identify these nodes and isolate them from the network.

Trust is one’s degree of belief about the future behavior of another entity (node). It is based on the one’s past experience with and observation of the other’s actions. Trust management involves formulating evaluation rules and policies, representing trust evidence, and evaluating and managing trust relationships. It was first introduced as a separate component of security in network services and given an overall definition in [2]. After that, considerable research on the topic follows. Thus far, many trust management systems [6–9] have been introduced in literature. They can be divided into two main categories: *credential-based* and *reputation-based*. In most credential-based systems, the trust relation between nodes is established by managing and exchanging credentials which should be verified and restricted by a preset policy. These systems usually make a binary decision, whether to trust or distrust a node. Because of this binary approach, they lack flexibility. On the other hand, reputation-based systems perform better by focusing on the evaluation of trust value. They calculate the trust value of a node by gathering observations of the node’s behavior in the past. A trusted node is one that always normally complete their assigned tasks; an untrustworthy node is one that does not provide desired services or provide abnormal services .

In this paper, in line with the reputation-based approach, we propose a novel trust management scheme for reliable routing in wireless ad hoc networks, by exploiting two classic autoregression models [12], i.e., Autoregressive (AR) model and Autoregressive with exogenous inputs (ARX) model. We present this scheme in the context of Greedy routing [13] and elaborate how AR and ARX are adopted to manage trust. According to the proposed scheme, a node a periodically measures the packet forwarding ratio of its every neighbor b. The measurements are the trust observations about b; they form a time series of data. Then, node a applies an autoregression model to the time series and predicts b’s future packet forwarding ratio as the trust estimate (or simply trust value) of b. In routing, a takes b as next hop if b is the one with the highest trust value among all neighbors closer to destination than *a* itself. With an AR model being applied, *a* only uses its own observations to predict *b*'s trust value; with an ARX model, *a* also takes into account the collective recommendation from neighbors, which is the weighted average of the trust values of *b* in those neighbors opinion. Through extensive simulation, we evaluate the scheme's performance in comparison with an existing trust management scheme based on Bayesian model [1]. Our simulation results indicate that use of the ARX model leads to the best trust estimation accuracy and the largest improvement on routing reliability.

The remainder of the paper is organized as follows. We briefly review some related work in Sec. II and introduce AR model and ARX model in Sec. III. Then, we present autoregression-based trust management for reliable routing in Sec. IV. After that, we report our comparative simulation study in Sec. V. We finally conclude the paper in Sec. VI.

II RELATED WORK

Recently, many mathematical techniques, such as graph theory, information theory, Bayesian theory and Markov chain, have been introduced to model the trust relation between nodes in computer networks. Due to space limitation, in this section we introduce only a few of them. Readers are referred to [6–9] for a comprehensive survey.

In [14], trust evaluation is viewed as a shortest path problem on a weighted directed graph, where vertices represent nodes. Each edge is associated with a weight indicating the opinions that one end node has about the other in the edge direction. Opinions are assigned to edges by nodes on the basis of their local observation and on their own criteria. Every opinion consists of two values, trust value and confidence value. The former is a node's trust estimate while the latter reflects the accuracy of the trust value assignment. An indirect trust relation without previous immediate experience is established by the semiring theory. The problem of this approach is that maintaining such a dynamic trust graph on each node is expensive. It requires frequent message exchange in the network for graph update.

In [15], the authors presented four axioms for establishing trust relationship, and suggested that trust is a measure of uncertainty, which is measured by entropy in information theory. The value of trust can be in the range [-1, 1]. Recommendations from other nodes and direct-observation-based trust are combined in such a way that the four axioms are satisfied, using entropy or probability. Recommendations are propagated by flooding. Hop-count constraint is used in flooding for reducing communication overhead. However, it is difficult to pre-define a proper hop count so that recommendations can be delivered to all interested nodes. This trust evaluation is based on incomplete information and thus inaccurate.

In [16], the authors addressed the issue of ignorance (resulted from lack of knowledge or mobility) on a node as well as the security problems of using explicit recommendation packets/headers, and proposed a novel subjective logic based trust model, where ignorance is modeled as uncertainty. A number of subjective logic operators including evidence-to-opinion mapping, discounting, and consensus, and fading operators are used to evaluate a node's trustworthiness. A node uses its previous-hop's intention to forward or discard a packet from the source node as the previous-hop's opinion about the source, and incorporate it as recommendation in evaluating the trustworthiness of the source. This method enables recommendation dissemination without message transmission and avoids false or fraudulent recommendations. However, the recommendations considered are from local nodes only. They do not fully reflect the nature of the source, resulting in inaccurate trust evaluation.

The authors in [17] proposed a semantic web trust model using vector autoregression. The authors gave an expression of the model and discussed the parameters estimation. However, they failed to provide enough details and results on how their model works for a semantic web. Compared with the insufficient description in [17], a linear hidden Markov (LHM) model is presented in [18] with a more complete analysis. LHM is designed for trust evaluation and avoids malicious feedbacks so that the trust prediction variance can be obtained. The autoregressive function is used to define the trust state evolution, and the Markov process is used to assess the prediction variance. However, even with this model it is difficult to achieve a better prediction because the feedbacks that it relies upon are subjective and insufficient. Additionally, the Markov process takes a long time to run into a constant precise state.

The Bayesian system was first proposed in [19] for handling trust in the field of e-commerce. It implements the probabilistic trust ratings based on the Beta probability density function, $Beta(\alpha, \beta)$, where $\alpha > 0$ and $\beta > 0$ respectively represent positive and negative ratings, as the binary inputs of the Bayesian system. The prediction of trust is the expectation value. In the recent variant [1], second-hand information (recommendations) is combined into the inputs. This extended scheme also provides aging parameters to give less weight to historical information. However it cannot achieve a satisfied fading rating because it simply assigns a less than 1 constant to the weighting factor.

III AUTOREGRESSION PRIMITIVES

Autoregression [12] is a way to understand and predict a time series of data. In this section, we briefly introduce two classic autoregression modes: Autoregressive (AR) model and Autoregressive with exogenous inputs (ARX) model. Later, in Sec. IV, these two models will be exploited for modeling the trust relation between nodes in wireless ad hoc networks.

An AR model is usually designed for analyzing and forecasting a time series of data such as average daily temperature and earthquake magnitude over the years. Recently, it has been adopted to predict neighborhood change for improving networking performance in wireless ad hoc networks [23, 24]. The model predicts the output value $y(t)$ of the series based on its p previous values $y(t-1), y(t-2), \ldots, y(t-p)$, called samples. A p-order autoregressive model, denoted as AR(p), can be expressed as follows:

$$y(t) = c + \sum_{i=1}^{p} \varphi_i y(t-i) + \varepsilon(t) \tag{1}$$

where c is a constant, $\varepsilon(t)$ is a white noise term with zero mean, $y(t)$ represents the output at time t, $\varphi_1, \ldots, \varphi_p$ are the coefficients of the model, p is the order of the model. The constant term c can be omitted for simplicity.

By the autoregression formula Eqn. 1, the output at time t can be estimated knowing φ_i and p. One problem in AR modeling is to find proper φ_i so as to represent the model best. There are different algorithms for computing AR model coefficients. The Least-Square (LS) approach and the Yule-Walker (YW) approach are the basic ones [20]. The model order p defines the size of the sample set. There is often a misunderstanding: the larger the model order, the more accurate the model. A large model order implies that a large number of samples from the past are used. However, when the samples are too old, they can be an interference to the prediction process as they may not be able to reflect the current value of the time series. Thus another problem is to select the optimal model order p. Two well-known selection criteria [21] are Final Prediction Error (FPE) and Akaike Information Criterion (AIC).

FPE is the mean-square error of the one-step-ahead prediction based on the LS estimated parameters of the AR model [21]. It is defined as follows:

$$FPE(p) = \frac{N+p}{N-p} P_p, \tag{2}$$

where P_p is the residual squared error for a p-order AR model, and N is the number of the estimate data samples. We take the order p when $FPE(p)$ is minimized.

AIC is a measure of goodness of fit of the model. In general, it is represented by:

$$AIC(p) = \ln P_p + \frac{2p}{N} \tag{3}$$

We select the order p that minimizes $AIC(p)$.

An ARX model is an extension of the AR model with input parts which can store external information. An ARX model with p outputs and b inputs, denoted as $ARX(p, b)$, can be written as follows:

$$y(t) = \varepsilon(t) + \sum_{i=1}^{p} \varphi_i y(t-i) + \sum_{i=1}^{b} \eta_i d(t-i) \tag{4}$$

where $\varphi_1, \varphi_2, \ldots, \varphi_p$ are the parameters of outputs and $\eta_1, \eta_2, \ldots, \eta_b$ the parameters of inputs. To determine the model order, we can still use the AIC method. The LS method can be used to estimate parameters.

IV AUTOREGRESSION-BASED TRUST MANAGEMENT

In this section, we introduce AR and ARX models into trust management for reliable ad hoc routing. Clearly, the adoption of the models is irrelevant to the routing protocol used. Here we choose to use Greedy routing [13] for simplicity. We assume that each node has a unique identity so that it can be distinguished from other nodes during communication.

A AR-based trust management

At initiation, nodes are unfamiliar with each other, and their trust rating for each other is set to default value ‘1’, which indicates a relationship of total trust. In this case, original Greedy routing is engaged for data communication.

Each node i locally slots the time. At the end of a time slot t, it measures the ratio of the number of the packets correctly forwarded by each neighbor j to the total number of packets it transmitted to j for forwarding during that time slot. We call the measurement *trust observation* of i about j in time slot t and denote them by $T_{i,j}(t)$. If there is not packet transmissions from i to j in time slot t, node i considers that $T_{i,j}(t)$ is equal to the average of its previous trust observations about j.

Having gathered sufficient trust observations about each neighbor j, node i no longer chooses next hop following the original Greedy routing rules. Instead, based the existing trust observations it predicts what its trust observation about j is going to be in the immediate next time slot, say slot $t+1$, if j is selected. We call the predicated trust observation *trust estimate* (or simply trust value) and denote it by $\hat{T}_{i,j}(t+1)$. Node i selects j only if the trust value of j implies that j is trustworthy. Notice that trust observations constitute a time series of data. Node i may do the predication by an AR model.

To judge whether or not node j is trustworthy, we use a trust threshold δ. If $\hat{T}_{i,j}(t+1) \geq \delta$, i will consider j trustworthy, and untrustworthy otherwise. The threshold has direct impact on the decision whether a node can be taken as a routing next hop. If it is too small, a malicious node may be chosen; if the value is too large, many good nodes may be excluded from the networking process, and the network performance will degrade. It is crucial to give the threshold a proper value. Here, we set the trust threshold δ to the mean $\overline{T}_i(t)$ of trust observations of i about all neighbors in current time slot t. Apparently, for each node i, the value of δ changes overtime according to the node's neighbors behavior dynamics. Note that it is possible that a node with a low trust value is never taken for cooperation by other nodes and thus never has the chance to increase its trust value. This can be mitigated by considering the frequency of nodes activity when making routing decisions.

B ARX-based trust management

The AR model only focus on the direct trust observations of a node, it is subjective and insufficient. When using the AR model, one also loses the supervision function of the trust management process because a node will never know a neighbor's behavior in the past until routing through it. If the neighbor is a good node, the communication will do benefit. If it is a malicious or selfish node, the communication will risk disruption. To solve this problem of lack of supervision, we propose to use an ARX model instead of the AR model.

The ARX model brings an input part to establish a supervision function to the trust management process, which are the recommendations provided by the neighboring nodes who have communicated with the evaluated node. Therefore, it calculates trust based on not only direct observations but also indirect observations.

When node i at local time slot t wants to predicts neighbor j's behavior in the next time slot $t+1$, it asks the other neighbors k_z $(1 \leq z \leq m)$ for their recommendations $R_{k_z,j}(t)$ (current trust estimates) about j based on their own trust observations. However, not every recommendation is trustworthy. Malicious or selfish nodes may provide false recommendations. To prevent these kinds of nodes influence on trust calculation, the recommendations should be weighted. Here we weigh the recommendation by each recommender's own trust value. Then the collective recommendation $R_i^j(t)$ to node i about its neighbor j by all the other neighbors $k_1, \ldots, k_m$ is defined as the weighted average of individual recommendations of these neighbors. Specifically,

$$R_i^j(t) = \frac{1}{m} \sum_{z=1}^{m} \hat{T}_{i,k_z}(t) \hat{T}_{k_z,j}(t). \tag{5}$$

With the trust values that node i has for neighbors $k_1, \ldots, k_m$ who give the recommendations, we can give trustworthy recommendations high weight and let them have more influence on the computing, and give untrustworthy recommendations low weight and minimize their effect. $R_i^j(t)$ is the final recommendation value that we are going to use as the exogenous inputs in the ARX model.

C Model validation and adjustment

Whether an AR model or an ARX model is used, the model coefficients and the model order can not remain the same all the time and have to be updated in accordance to nodal behavior dynamics. The model needs to be validated periodically. If the validation result meets a minimum requirement, for example, a data threshold, the model is considered accurate and can stay unchanged; otherwise, it needs to be updated based on recent trust observations (and recommendations in the case that ARX mode is used).

To validate the model, we use the coefficient of determination or R^2 presented in [22]. R^2 is used to measure how well a model predicts a future performance based on comparing the predictions with the corresponding outcomes. In linear regression, R^2 can be simply defined as the square of the differences between predictions and outcomes, represented as

$$R^2 = 1 - \frac{SS_{err}}{SS_{tot}}, \tag{6}$$

where SS_{err} is the sum of squares of residuals which is the difference between outcomes and predictions, SS_{tot} is the total sum of squares of difference between outcomes and the average of the outcomes.

Suppose that each validation interval spans q time slots. For ease of presentation, we index the q time slots from 1 to q. In correspondence to our trust management context, the representation of R^2 is then revised as follows:

$$R^2 = 1 - \frac{\sum_{t=1}^{q} \varepsilon_{i,j}(t)^2}{\sum_{t=1}^{q} (T_{i,j}(t) - \overline{T}_{i,j}(t))^2} \tag{7}$$

where $\varepsilon_{i,j}(t) = T_{i,j}(t) - \hat{T}_{i,j}(t)$ and $\overline{T}_{i,j}(t)$ is the mean of trust observations in the validation interval.

R^2 reflects how trust estimates are close to trust observations. The close ones come with high R^2 values. According to [22], when $R^2 < 0.85$, for example, the model coefficients and the model order normally should be re-estimated using the trust observations (and recommendations). Here 0.85 is a predefined threshold representing validation minimum requirement.

V PERFORMANCE EVALUATION

We simulated the two autoregression models and the Bayesian model [1] using MATLAB and evaluate their performance for reliable Greedy routing [13]. We considered *model accuracy* and *message drop rate* as two main criteria for comparing these models. Model accuracy is measured by predication error, which is defined as the distance between observed trust value and predicted value. The smaller predication error, the more accurate the model. For analyzing which model could do a better prediction, we choose two neighboring nodes at random and compare one's trust observations and estimates about the other. In addition, we also use AIC and FPE parameters to assess which model is a better choice for trust management. When considering drop rate, there are two causes for which a node drops a message: (1) no neighbor with sufficiently high trust value and (2) lack of delivery guarantee of greedy routing itself.

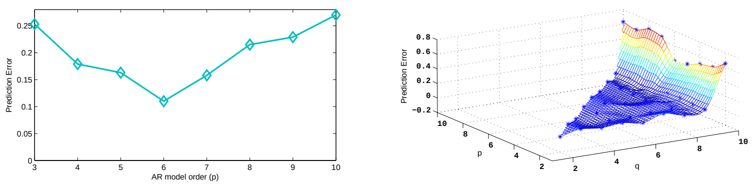
A Simulation setup

We randomly deploy a connected wireless ad hoc network of 100 nodes in a rectangular area of 100×100. Nodes have the same communication range 10, and network connectivity check is by Dijkstra's algorithm. We simulated 600 data communication sessions started at random in time, each between a randomly selected pair of nodes and involves 100 data packet transmissions at constant rate of 1 packet per simulated time unit. Fifteen nodes (non source or destination) are randomly selected as malicious nodes. They drop every received packet with a constant probability of 0.7. Proper nodes drop packets with a small probability of 0.15, simulating unreliable wireless communication. Each (trust observation) time slot is composed of 5 consecutive simulated time units, in other words, each data communication session spans 20 time slots.

B Simulation results

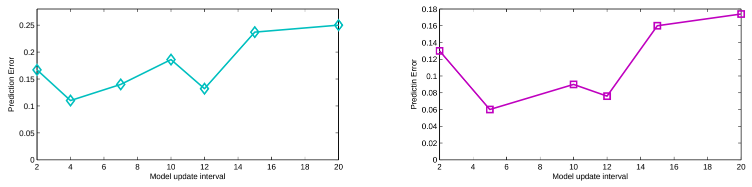
We first study model accuracy in relation with model order. For AR model, the model order p defines the number of previous observations used for predication. It must be carefully selected. A large order value causes the prediction to mostly rely on the previous reputations, which may lead to a prediction lag. On the other hand, a small order value cannot provide enough information for a precise prediction. Figure 1(a) unveils that the prediction error decreases before p reaches 6 and increases afterwards. The lowest predication error is obtained when $p = 6$.

For ARX model, the model order is a pair of values (p, q), where p is the AR model order and q is the exogenous input order. Figure 1(b) reports that when



(a) AR model (b) ARX model

FIGURE 1
AR model predication error vs. model order



(a) AR model (b) ARX model

FIGURE 2
ARX model predication error vs. update interval

the ARX order is (4, 3), the prediction error is the lowest. This order indicates that the prediction is the most accurate when it relies also on the proper number of objective recommendations instead of only on the subjective direct observations. If we increase p, the prediction becomes inaccurate because the information is too subjective. If we increase q, the prediction is not precise because it leans too much on the recommendations from other nodes, which may not be trustworthy.

In addition to model order, the accuracy of AR/ARX models are also subject to the model update interval, which determines how often the models (coefficients) are corrected. If we update it too often (i.e. update period is too small), we calculate the coefficient using only a little piece of information, which may bring forth an inaccurate prediction. Furthermore, updating the model too often costs the system a large amount of computational effort and storage memory. On the contrary, if we set a long update period, the model will quickly turn outdated given that it will not fit the newly updated observation values, thus leading to a decline in prediction accuracy.

Figure 2(a) discloses the relationship between prediction error and model update interval for AR model. The curve reaches the lowest point when the update interval is 5 time slots. Similarly, the best update interval for ARX model is also 5 time slots as indicated by Figure 2(b). In the following, we compare the AR and ARX models with the Bayesian model in terms of predication error. They are constructed using the best parameter settings. That is, their orders are set to 6 and (4, 3) respectively. The model update interval is set to 5 for both of them. That is, model validation is performed every 5 time slots, and model update is conducted immediately as needed according to the validation result.

Figure 3 shows trust observations about a randomly selected malicious node in comparison with the trust estimates by all three models during the initial 70 data sessions. At the beginning, the estimates by the AR and the ARX model are much higher than the observations because predication is not possible due to lack of information and trust estimate is set to the default value of 1. For the Bayesian model, the estimate is around 0. 5 at the beginning which means that both trusted behavior and untrusted behavior have the same

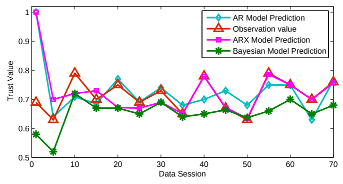


FIGURE 3
Trust value over the first 70 data sessions

probability of happening. With the increase in the number of date sessions, the estimates by all three models get closer to the observations as expected. There is noticeable difference in the case of the ARX model as its trust estimates are very close to the observations, which indicates that it is the best model among all three models considered.

We further computed the ACI and FPE indexes [21] of the three modes, as shown in Tab. 1. The model with lowest AIC value best represents the time series (trust evolution) with a minimum number of parameters (i.e., minimum model order); the most accurate model has the smallest FPE value. As stated in Tab. 1, the AIC and FPE indexes of the ARX model is lower than those of the other two models, which means that the ARX model could best represent the trust evolution of a node with a fewer output parameters than the AR and Bayesian models, confirming our previous simulation analysis.

We finally turn our attention to the influence of these trust models on routing. Figure 4 shows the message drop rate of Greedy routing (without

	AIC	FPE
AR	−11.2660	0.0019
ARX	−17.7852	0.0016
Bayesian	−11.5207	0.0077

TABLE 1
Index for Comparison Between the Models

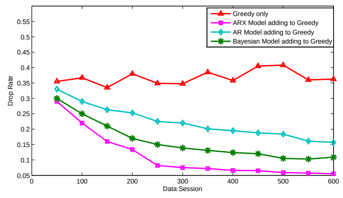


FIGURE 4
Message Drop Rate

engaging trust management), Greedy with Bayesian model [1], Greedy with AR model, and Greedy with ARX model. Greedy with built-in trust models have a lower drop rate than Greedy alone. We observe that the drop rates under different trust models have a process from closing to each other at the beginning to falling separately after several data sessions. This is because nodes can predict each other's trust values based on their pass behavior after a few interactions. Around 250 data sessions or so, the drop rates become stabilized. ARX model has the best performance as the drop rate falls most quickly among all three models. The Bayesian model leads to a lower drop rate than the AR model because it takes into account second-hand information (recommendations) when computing nodal trust, which the AR model does not include. Under the same condition, the ARX model does a better job than the Bayesian model, because its data source is based on a more direct numerical expression (weighting observations and recommendations) as opposed to the indirect expression of the Bayesian model. In Bayesian model, the data source needs mathematical transformation before prediction, which brings more errors than ARX model.

VI CONCLUSIONS

We proposed to use an autoregressive model (AR) and an autoregressive with exogenous inputs model (ARX) in wireless ad hoc networks for establishing trust between nodes on the basis of their prior interactions. We use Greedy routing [13] to demonstrate their applicability and effectiveness. The AR model is efficient in establishing trust relations since it only counts on the direct interactions; however, it is lacking supervisions by other nodes. The ARX mode with an addition of recommendation input part makes the supervisions by other nodes available to build trust in a more reliable way but less efficient (requiring extra local space for storing the recommendations) compared to the AR model. Our simulations result show that the ARX model performs best in term of accuracy and improvement brought to Greedy routing.

ACKNOWLEDGMENTS

This work was partially supported by the following grants: "Reliable and secure QoS routing and transport protocols for mobile ad hoc networks", NSERC CRDPJ 386874-09 and "Digital signal processing, and the synthesis of an information security system", TR32054, Serbian Ministry of Science and Education.

REFERENCES

[1] S. Ganeriwal, L. Bolzano, M. B. Srivastava, "Reputation based Framework for High Integrity Sensor Networks". ACM Trans. Sensor Networks, 4(3), 2008.

[2] M. Blaze, J. Feigenbaum, and J. Lacy, “Decentralized Trust Management”. Proc. IEEE SP, pp. 164-173, 1996.

[3] J. Weise, “Public Key Infrastructure Overview”. Sun BluePrints, 2001.

[4] S. Garfinkel, PGP: Pretty Good Privacy, O’Reilly & Associates, 1995.

[5] K. Ren, T. Li, Z. Wan, F. Bao, R.H. Deng, and K. Kim, “Highly reliable trust establishment scheme in ad hoc networks”. Journal of Computer Networks, pp. 687-699, 2004.

[6] S. Ruohomaa and L. Kutvonen, “Trust management survey”. Proc. ITRUST, LNCS 3477, pp. 77-92, 2005.

[7] J. Sabater and C. Sierra, “Review on computational trust and reputation models”. Artificial Intelligence Review, 24(1): 33-60, 2005.

[8] H. Li and M. Singhal, “Trust management in distributed systems”. IEEE Computer, 40(2): 45-53, 2007.

[9] A. Jøsang, R. Ismail, and C. Boyd, “A survey of trust and reputation systems for online service provision”. Decision Support Systems, 43(2): 618-644, 2007.

[10] X. Li, H. Frey, N. Santoro, and I. Stojmenovic, "Strictly Localized Sensor Self-Deployment for Optimal Focused Coverage". IEEE Trans. Mobile Computing, 10(11): 1520-1533, 2011.

[11] X. Li, X. Liang, R. Lu, S. He, J. Chen, and X. Shen, "Toward Reliable Actor Services in Wireless Sensor and Actor Networks". Proc. IEEE MASS, 2011.

[12] G., Box, G.M. Jenkins, and G.C. Reinsel, Time Series Analysis: Forecasting and Control, 4th edn. Wiley, Chichester, 2008.

[13] G.G. Finn, “Routing and Addressing Problems in Large Metropolitan-Scale Internetworks”. ISI Research Report, University of Southern California, Marina Del Rey, pp. 65-66, 1987.

[14] G. Theodorakopoulos, J.S. Baras, “Trust evaluation in ad-hoc networks”. Proc. ACM WiSe, pp. 1-10, 2004.

[15] Y.L. Sun, W. Yu, Z. Han, and K.J.R. Liu, “Information theoretic framework of trust modeling and evaluation for ad hoc networks”. IEEE Journal on Selected Areas in Communications, 2006.

[16] V. Balakrishnan, V. Varadharajan, and U. Tupakula, “Subjective Logic Based Trust Model for Mobile Ad hoc Networks”. Proc. Securecomm, 2008.

[17] J. Qin and L. Chen, “Trust Management for Semantec Web”. Proc. ICCSSE, pp. 778-781, 2008.

[18] X. Wang, W. Ou, and J. Su, “A reputation inference model based on linear hidden markov process”. Proc. CCCM, pp. 354-357, 2009.

[19] A. Josang and R. Ismail, “The Beta Reputation System”. Proc. Bled eConference, pp. 324-337, 2002.

[20] P. Stoica, B. Friedlander, and T. Soderstrom, “Least-squares, Yule-Walker, and overdetermined Yule-Walker estimation of AR parameters: a Monte Carlo analysis of finite-sample properties”. International Journal of Control, 43(1):13-27, 1986.

[21] H.Akaike, “A new look at statistical model identification”. IEEE Trans. Automatic Control, 19(6):716-723, 1974.

[22] D.C. Montgomery, G.C. Runger, and N.F. Hubele, Engineering Statistics, John Wiley and Sons Inc., 2004.

[23] X. Li, N. Mitton, and D. Simplot-Ryl, “Mobility Prediction Based Neighborhood Discovery for Mobile Ad Hoc Networks”. Proc. IFIP NETWORKING, pp. 138-151, 2011.

[24] X. Liang, X. Li, J.Q. Shen, R. Lu, X. Lin, X. Shen, and W. Zhang, “Exploiting Prediction to Enable Secure and Reliable Routing in Wireless Body Area Networks”. Proc. IEEE INFOCOM, 2012.

[25] X. Liang, X. Li, R. Lu, X. Lin and X. Shen, “SEER: A Secure and Efficient Service Review System for Service-oriented Mobile Social Networks”. Proc. IEEE ICDCS, 2012.

[26] R. Lu, X. Li, X. Liang, X. Lin, and X. Shen, “GRS: The Green, Reliability, and Security of Emerging Machine to Machine Communications”. IEEE Communications Magazine, 49(4): 28-35, 2011.

[27] R. Lu, X. Lin, X. Shen, “SPRING: A Social-based Privacy-preserving Packet Forwarding Protocol for Vehicular Delay Tolerant Networks”. Proc. IEEE INFOCOM, pp. 632-640, 2010.