International Journal of Applied Information Systems (IJAIS) ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 12 - No. 16, October 2018 - www.ijais.org

SoTRMSim: Sociopsychological Trust and Reputation Models Simulator for Wireless Sensor Networks

Henry Nunoo-Mensah
Department of Computer Engineering
Kwame Nkrumah University of Science and Technology

Kwame Osei Boateng
Department of Computer Engineering
Kwame Nkrumah University of Science and Technology

James Dzisi Gadze
Department of Electrical/Electronic Engineering
Kwame Nkrumah University of Science and Technology

Griffith Selorm Klogo
Department of Computer Engineering
Kwame Nkrumah University of Science and Technology

ABSTRACT

The use of trust and reputation models (TRMs) are on the rise due to the increasing complexities of cryptography. The use of cryptography though potent on traditional networks, cannot be supported by the resource constraint wireless sensor network (WSN). Trust evaluations have made use of a number of approaches, such as analytical, bio- and socio-inspired methods, in the design of TRMs. The use of socio-inspired methods for TRM design is given less attention though its a simple and effective method. The paper proposes the first sociopsychological TRM simulator (SoTRMSim). The models implemented in SoTRMSim are the consensus-aware and privacy-aware sociopsychological TRMs by Rathore and Nunoo-Mensah respectively. The proposed simulator provides an objective way of simultaneously evaluating sociopsychological TRMs.

Keywords:

wireless sensor network, security, trust and reputation models, sociopsychology, simulator

1. INTRODUCTION

Most wireless sensor networks (WSNs) are deployed for critical applications but the unattended nature of field nodes make their security paramount. This is because adversaries can easily attack these nodes on network. The captured nodes can be reprogrammed to maliciously disturb the normal operations of the network. Security schemes developed for traditional networks cannot be easily transferred to WSNs due to certain restrictions or constraints that are evident in WSNs alone and not found in traditional networks. Some of the constraints pronounced in WSNs are the limited processing capabilities, memory or space restrictions, unreliable communication, higher communication latency and the unattended operation of the networks. As a result of these constraints, security schemes proposed for WSNs need to be optimised [1].

A survey conducted by [12] presented methods of evaluating or assessing the trustworthiness of sensor nodes. Their work categorised the trust evaluation methods into analytical, bio- and socio-inspired approaches. Analytical methods [15, 16, 3, 5, 18] of evaluating trust is dominantly patronised by many security researchers. Analytical approaches are computationally expensive and thus new ways of evaluating trust need to be explored. The use of bio-inspired methods [9, 14, 20, 17] of assessing a node's trust have been explored by some researchers. These proposed bio-inspired models for WSNs provide very flexible and simple approaches for solving complex security problems facing the network. The issue with such models is predominantly the slow convergence rate.

A solution to the drawbacks of analytical and bio-inspired TRMs is the use of socio-inspired trust evaluation models [13, 11]. These models as discussed in [12] provide the pros and cons of the various trust evaluation methods (i.e., analytical, bio- and socio-inspired). There are only a handful of works involving sociopsychological constructs into the modelling of their models. The integration of psychological antecedents and other socio constructs into the design of trust models for WSNs is given less attention by researchers in the domain. There are only two sociopsychological models as of the time of writing this paper are [13] and [11].

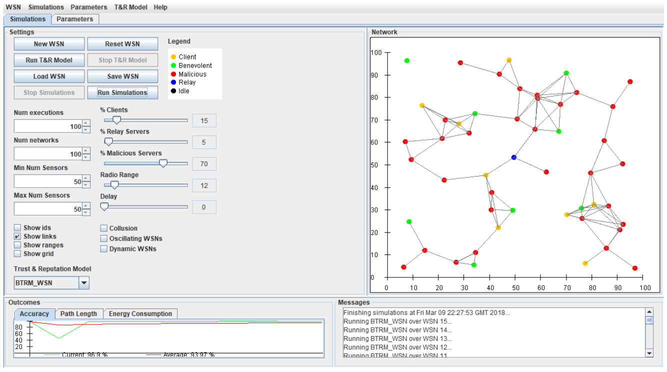
The contribution of this paper is to introduce the first open source Sociopsychological TRM Simulator (SoTRMSim), it is supposed to aid in the design and simulation of sociopsychological trust models meant for WSNs. This simulator is meant to increase the zeal and motivation by other researchers in undertaking sociopsychological trust and reputation model designs. It will provide an avenue to simulate both existing and personally proposed sociopsychological TRMs in order to facilitate the effective evaluation of these models.

The paper is organised into the following sections. Section 2 summarises related work in the area of simulators for trust modelling and evaluation in WSNs. Section 3 presents the proposed MATLAB based simulator for simulating sociopsychological TRMs designed for WSNs. A simulation scenario is outlined in section 4 for evaluating the currently implemented models in the simulator (i.e, Rathore et al. [13] and Privacy-aware Sociopsychological TRM [11]). This section also presents some results from the simulations carried out using the simulation scenario. Section 5 concludes the work and presents future directions for the proposed SoTRMSim.

2. RELATED WORKS

A number of simulators have been proposed for WSNs to help facilitate the testing and evaluation of new communication protocols. This is in order to check the correctness, robustness or accuracy of these communication protocols. A survey presented by [10] sort to discuss as many publicly available simulators they could find. Their work classified the various simulators available for WSNs into: emulators and code level simulator, topology control simulators, environment and wireless medium simulators, network and application level simulators, cross level simulators, NS-2 based simulators, OMNeT++ based simulators, and Ptolemy II based simulators.

The scope of this paper limits the review carried out to security oriented network and application level simulators. Some existing state-of-the-art security centric network and application level simulators include Sensor Security Simulator ($S3$) and TRMSim-WSN [6].



Num executions: 100
Num networks: 100
Min Num Sensors: 50
Max Num Sensors: 50
% Clients: 15
% Relay Servers: 5
% Malicious Servers: 70
Radio Range: 12
Delay: 0

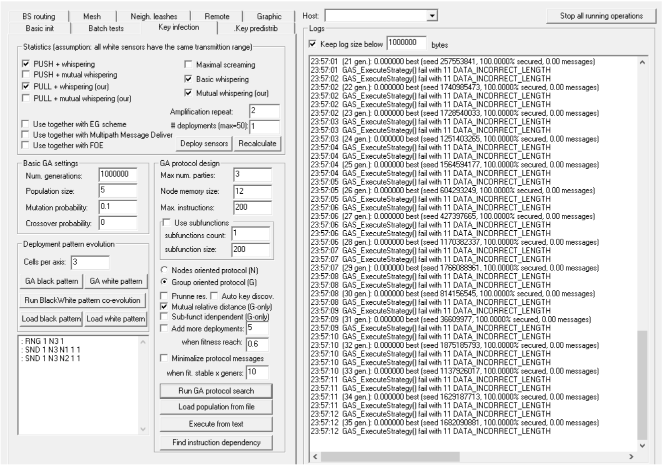
□ Show ids
☑ Show links
□ Show ranges
□ Show grid
□ Collusion
□ Oscillating WSNs
□ Dynamic WSNs

Trust & Reputation Model: BTRM_WSN

```
Finishing simulations at Fri Mar 09 22:27:53 GMT 2018...
Running BTRM_WSN over WSN 15...
Running BTRM_WSN over WSN 14...
Running BTRM_WSN over WSN 13...
Running BTRM_WSN over WSN 12...
Running BTRM_WSN over WSN 11...
```

Fig. 1: TRMSim-WSN

TRMSim-WSN is a Java based simulator which was designed to simulate trust and reputation models for WSNs. The authors set out to provide an objective means of analysing existing and future peer-to-peer (P2P) trust models. TRMSim-WSN currently has the following models implemented: BTRM-WSN [7], Eigen-Trust [4], PeerTrust [19], PowerTrust [21], LFTM [2], TRIP [8]. The simulator measures the accuracy, path length and energy consumption of the network. The authors also made provision for users to simulate their own models by implementing the subclasses: $TRModel\_WSN$, $Service$ and $TRMParameters$. The models that have been implemented are simulated individually for the generated network. This makes comparison with other models difficult since the models have to be run individually. The results of simulations carried out cannot be exported thus making it difficult to control data outputs from the simulator. Figure 1 shows the simulations interface where the network and sensor properties can be set. The simulation tab interface allows the user to select the TRM that should be run for that particular simulation instance over the already deployed network. The parameters tabs allows the user to load custom or predefined model parameters into the simulator. These parameters are defined in a text file.



Statistics (assumption: all white sensors have the same transmittion range)

☑ PUSH + whispering
□ PUSH + mutual whispering
☑ PULL + whispering (our)
□ PULL + mutual whispering (our)
□ Maximal screaming
☑ Basic whispering
☑ Mutual whispering (our)
Amplification repeat: 2
# deployments (max=50): 1
□ Use together with EG scheme
□ Use together with Multipath Message Deliver
□ Use together with FOE

Basic GA settings
Num. generations: 1000000
Population size: 5
Mutation probability: 0.1
Crossover probability: 0

Deployment pattern evolution
Cells per axis: 3

```
: RNG 1 N3 1
: SND 1 N3 N1 1 1
: SND 1 N3 N2 1 1
```

GA protocol design
Max num. parties: 3
Node memory size: 12
Max. instructions: 200
□ Use subfunctions
subfunctions count: 1
subfunction size: 200
○ Nodes oriented protocol (N)
● Group oriented protocol (G)
□ Prunne res. □ Auto key discov.
☑ Mutual relative distance (G-only)
□ Sub-funct idenpendent (G-only)
□ Add more deployments: 5
when fitness reach: 0.6
□ Minimalize protocol messages
when fit. stable x geners: 10

Host:
Logs
☑ Keep log size below 1000000 bytes

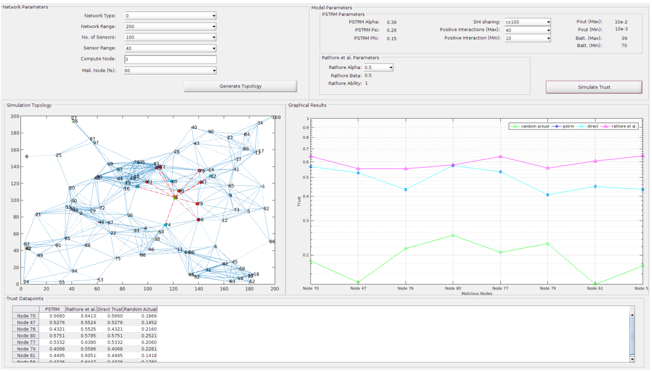
```
23:57:01  (21 gen.): 0.000000 best (seed 257553841, 100.0000% secured, 0.00 messages)
23:57:01  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:02  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:02  (22 gen.): 0.000000 best (seed 1740985473, 100.0000% secured, 0.00 messages)
23:57:02  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:02  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:02  (23 gen.): 0.000000 best (seed 1728540033, 100.0000% secured, 0.00 messages)
23:57:03  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:03  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:03  (24 gen.): 0.000000 best (seed 1251403265, 100.0000% secured, 0.00 messages)
23:57:04  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:04  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:04  (25 gen.): 0.000000 best (seed 1564594177, 100.0000% secured, 0.00 messages)
23:57:04  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:05  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:05  (26 gen.): 0.000000 best (seed 604293249, 100.0000% secured, 0.00 messages)
23:57:05  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:06  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:06  (27 gen.): 0.000000 best (seed 427397665, 100.0000% secured, 0.00 messages)
23:57:06  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:06  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:06  (28 gen.): 0.000000 best (seed 1170382337, 100.0000% secured, 0.00 messages)
23:57:07  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:07  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:07  (29 gen.): 0.000000 best (seed 1766088961, 100.0000% secured, 0.00 messages)
23:57:08  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:08  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:08  (30 gen.): 0.000000 best (seed 814156545, 100.0000% secured, 0.00 messages)
23:57:08  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:09  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:09  (31 gen.): 0.000000 best (seed 36609977, 100.0000% secured, 0.00 messages)
23:57:09  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:10  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:10  (32 gen.): 0.000000 best (seed 1875185793, 100.0000% secured, 0.00 messages)
23:57:10  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:10  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:10  (33 gen.): 0.000000 best (seed 1137926017, 100.0000% secured, 0.00 messages)
23:57:11  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:11  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:11  (34 gen.): 0.000000 best (seed 1629187713, 100.0000% secured, 0.00 messages)
23:57:11  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:12  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
23:57:12  (35 gen.): 0.000000 best (seed 1682090881, 100.0000% secured, 0.00 messages)
23:57:12  GAS_ExecuteStrategy() fail with 11 DATA_INCORRECT_LENGTH
```

Fig. 2: Sensor Security Simulator (S3) graphical user interface

In $S3$, simulations are based on pseudo-random number generators with specified seed defined for each experiment. This process makes all simulation steps fully deterministic. The deterministic characteristic of each simulation can yet be complemented by random operations when required. Batch sequential execution of several simulations is supported and parameters of network topologies can be iterated over destined ranges. The simulator also provides a graphical user interface as shown in Figure 2, where simulation parameters can be set and simulations started. The simulator was developed using the C++ programming language. The data gathered during each simulation is stored in files which are formatted to facilitate further processing, data visualisation and graph plotting in MATLAB. $S3$ is predominantly designed to be used for routing protocol security. Trust simulations for WSNs have received far less attention, the only simulator that considers some level of TRM inclusion only focuses on bio-inspired models and not socio based TRMs. In order to encourage the design of socio-inspired TRMs by researchers, a sociopsychological simulator is proposed and presented in the next section.

3. SOTRMSIM

This section presents the proposed Sociopsychological Trust and Reputation Models Simulator (SoTRMSim) and its operations. The simulator is MATLAB based and provides an objective means of computing and evaluating sociopsychological TRMs for WSNs. The graphical user interface of SoTRMSim showing the network and TRM parameters, network deployment, plotting and plotted data points sections, is illustrated in Figure 3. The main classes within SoTRMSim are the $GUINetwork$ and $PSTRMGui$. The $GUINetwork$ provides the following subroutines: *generateNet*, *highlightComputeNode*, *getNeighbours*, *placeMaliciousNodes*, *generateMaliParam*, *computeAbility*, *computeIndirectInfo*, *averageIndirectValues*, *computeBenevolence*, *generateConsistencyParam*, *computeConsistency*, *computePstrmTrust*, *makeTrustPlot*, and *computeRathoreTrust*.



Network Type: 0
Network Range: 200
No. of Sensors: 100
Sensor Range: 40
Compute Node: 3
Mali. Node (%): 60

PSTRM Alpha: 0.39
PSTRM Psi: 0.26
PSTRM Phi: 0.15
SHI sharing: co100
Positive Interactions (Max): 40
Positive Interaction (Min): 10
Pout (Max): 10e-2
Pout (Min): 10e-3
Batt. (Max): 99
Batt. (Min): 70

Rathore Alpha: 0.5
Rathore Beta: 0.5
Rathore Ability: 1

	PSTRM	Rathore et al.	Direct Trust	Random Actual
Node 70	0.5660	0.6413	0.5660	0.1866
Node 47	0.5276	0.5524	0.5276	0.1452
Node 76	0.4321	0.5525	0.4321	0.2160
Node 80	0.5751	0.5785	0.5751	0.2521
Node 77	0.5332	0.6380	0.5332	0.2060
Node 79	0.4068	0.5586	0.4068	0.2281
Node 61	0.4495	0.6051	0.4495	0.1418
Node 56	0.4326	0.6447	0.4326	0.1760

Fig. 3: Graphical user interface of SoTRMSim

The *generateNet* subroutine generates the WSN topology based on pseudo-random $X - Y$ coordinates for the sensor nodes. The number of nodes on comprising the network, the network size, and the radio range of deployed sensor nodes are supplied as arguments to the subroutine. The subroutine returns a WSN based on the parameters supplied as arguments. The *highlightComputeNode* subroutine receives the identity (ID) of the compute node i together with its X and Y coordinates and returns a plot of the compute node's position on the deployed network. The *getNeighbours* subroutine selects the neighbours of the compute node's radio range. The *placeMaliciousNodes* subroutine pseudo-randomly selects number of nodes that commensurate with the percentage of malicious neighbours supplied to the subroutine. The battery levels and outage probabilities are also pseudo-randomly generated using the *generateMaliParam* subroutine. The ability of the malicious nodes is computed using *computeAbility* and the indirect information by *friends* are processed using *computeIndirectInfo* subroutine.

The indirect information received from *friends* are averaged with the *averageIndirectValues* subroutine and the returned value supplied together with the direct information observed on the *trustor* and the benevolence computation using the *computeBenevolence* subroutine. The consistency parameters needed to compute the consistency of malicious nodes are generated using the *generateConsistencyParam* subroutine. The returned values from the called *generateConsistencyParam* subroutine are supplied to the *computeConsistency* subroutine in order to calculate the consistency of each malicious node. The PSTRM and Rathore models are computed using *computePstrmTrust* and *computeRathoreTrust* respectively. The results from all the computations are plotted with the help of the *makeTrustPlot* subroutine.

4. TYPICAL SIMULATION SCENARIO

This section showcases the models that have been implemented in the current version of SoTRMSim. The implemented models are Rathore et al. [13] and PSTRM [11]. A scenario illustrating the operations of the simulator is simulated. The parameters utilised in this simulation are defined in table 1. There are 5 network types derived due to 5 different seeds for the random number generator routine in MATLAB.

Table 1. : Wireless sensor network deployment parameters

Parameters	Specifications
Network type	$type$ 2
Network range	100 m x 100 m
Number of sensors	100
Sensor range	10 m
Trustor node	$Node$ 6
% of malicious nodes	80%

The network type used for this particular simulation is type 2. The created network spans an area defined by the dimensions $100m\,x\,100m$. 100 sensor nodes are deployed pseudo-randomly within the network area. Each sensor node in the area is equipped with a radio with a range of $10m$. $Node 6$ is selected as the compute node and 80% of the neighbours of $Node$ 6 are considered to be malicious. The deployed network utilised in simulating the implemented models is shown in Figure 4.

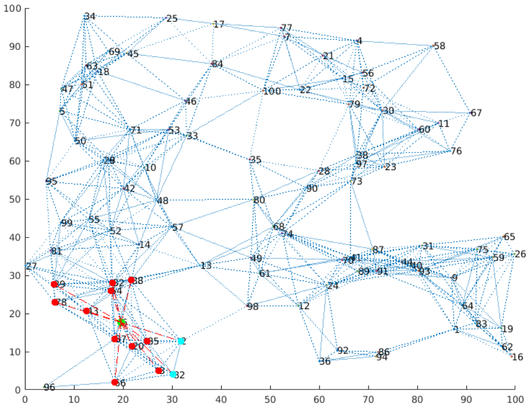


Fig. 4: Extracted network deployment from SoTRMSim

Figure 5 shows the simulated results from the simulator. It is juxtaposing all the implemented models for each comparison and inference. The data points can be exported and used with any plotting package such as R software package etc. This ensures flexibility and allows researchers to use their preferred plotting software packaging other than MATLAB.

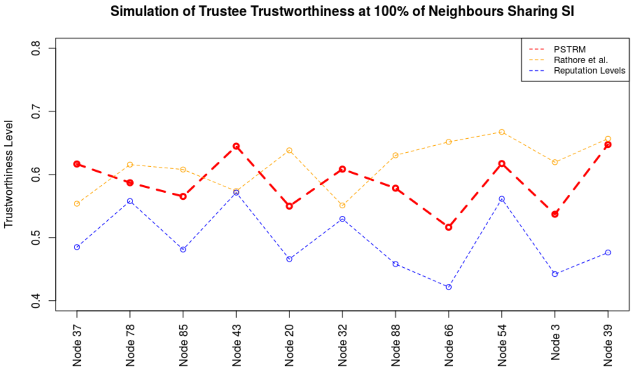


Fig. 5: Extracted data plot from SoTRMSim

The results in figure 5 show the reputation levels generated for the various malicious nodes as well as the computed trust values for both Rathore *et al.* [13] and PSTRM [11].

5. CONCLUSION AND FUTURE WORK

The paper proposes SoTRMSim, a simulator for conducting simulations involving sociopsychological WSN TRMs. The simulator currently implements the only two existing sociopsychological TRMs found in literature and also provides an objective way of assessing the existing proposed sociopsychological models. The proposed simulator also seeks to encourage and motivate researchers into adopting and advancing the design of TRMs using sociopsychological antecedents for WSNs.
Since this is the first iteration of the simulator, work is already under way to increase the modularity and flexibility of the simulator in subsequent releases. The aim is to expand the already implemented models by continuously supporting the simulator and integrating future models. SoTRMSim is free for download on the Connected Devices (CoDe) Lab website (https://connecteddeviceslab.org/sotrmsim-sociopsychological-trust-and-reputation-models-simulator-for-wireless-sensor-networks/).

6. REFERENCES

[1] David W Carman, Peter S Kruus, and Brian J Matt. Constraints and approaches for distributed sensor network security (final). Technical Report 1, Trusted Information System, NAI Labs, 2000.

[2] Félix Gómez Mármol, Javier G Marín-Blázquez, and Gregorio Martínez Pérez. Lftm, linguistic fuzzy trust mechanism for distributed networks. *Concurrency and Computation: Practice and Experience*, 24(17):2007–2027, 2012.

[3] Farruh Ishmanov, Sung Won Kim, and Seung Yeob Nam. A robust trust establishment scheme for wireless sensor networks. *Sensors*, 15(3):7040–7061, 2015.

[4] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651. ACM, 2003.

[5] Lianggui Liu, Li Chen, and Huiling Jia. Social milieu oriented routing: a new dimension to enhance network security in wsns. *Sensors*, 16(2):247, 2016.

[6] F. G. Marmol and G. M. Perez. Trmsim-wsn, trust and reputation models simulator for wireless sensor networks. In *2009 IEEE International Conference on Communications*, pages 1–5, June 2009.

[7] Félix Gómez Mármol and Gregorio Martínez Pérez. Providing trust in wireless sensor networks using a bio-inspired technique. *Telecommunication systems*, 46(2):163–180, 2011.

[8] Félix Gómez Mármol and Gregorio Martínez Pérez. Trip, a trust and reputation infrastructure-based proposal for vehicular ad hoc networks. *Journal of Network and Computer Applications*, 35(3):934–941, 2012.

[9] Hosein Marzi and Mengdu Li. An enhanced bio-inspired trust and reputation model for wireless sensor network. *Procedia Computer Science*, 19:1159–1166, 2013.

[10] Bartosz Musznicki and Piotr Zwierzykowski. Survey of simulators for wireless sensor networks. *International Journal of Grid and Distributed Computing*, 5(3):23–50, 2012.

[11] Henry Nunoo-Mensah. Privacy-aware sociopsychological trust and reputation model for wireless sensor networks, 2018.

[12] Henry Nunoo-Mensah, Kwame Osei Boateng, and James Dzisi Gadze. The adoption of socio- and bio-inspired algorithms for trust models in wireless sensor networks: A survey. *International Journal of Communication Systems*, pages e3444–n/a, 2017. e3444 dac.3444.

[13] Heena Rathore, Venkataramana Badarla, and KJ George. Sociopsychological trust model for wireless sensor networks. *Journal of Network and Computer Applications*, 62:75–87, 2016.

[14] Heena Rathore and Sushmita Jha. Bio-inspired machine learning based wireless sensor network security. In *Nature and Biologically Inspired Computing (NaBIC), 2013 World Congress on*, pages 140–146, Fargo, ND, USA, 2013. IEEE.

[15] Jiaojiao Song, Xiaohong Li, Jing Hu, Guangquan Xu, and Zhiyong Feng. Dynamic trust evaluation of wireless sensor networks based on multi-factor. In *Trustcom/BigDataSE/ISPA, 2015 IEEE*, volume 1, pages 33–40, Helsinki, Finland, 2015. IEEE.

[16] Ayman Tajeddine, Ayman Kayssi, Ali Chehab, Imad Elhajj, and Wassim Itani. Centera: a centralised trust-based efficient routing protocol with authentication for wireless sensor networks. *Sensors*, 15(2):3299–3333, 2015.

[17] Vinod Kumar Verma. Pheromone and path length factor-based trustworthiness estimations in heterogeneous wireless sensor networks. *IEEE Sensors Journal*, 17(1):215–220, 2017.

[18] Jian Wang, Shuai Jiang, and Abraham O Fapojuwo. A protocol layer trust-based intrusion detection scheme for wireless sensor networks. *Sensors*, 17(6):1227, 2017.

[19] Li Xiong and Ling Liu. Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE transactions on Knowledge and Data Engineering*, 16(7):843–857, 2004.

[20] Mingchuan Zhang, Changqiao Xu, Jianfeng Guan, Ruijuan Zheng, Qingtao Wu, and Hongke Zhang. A novel bio-inspired trusted routing protocol for mobile wireless sensor networks. *TIIS*, 8(1):74–90, 2014.

[21] Runfang Zhou and Kai Hwang. Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Transactions on parallel and distributed systems*, 18(4):460–473, 2007.