Electronic Preprint for Journal of Information Processing Vol.22 No.2

Regular Paper

Trust-based Adaptive Routing for Smart Grid Systems

Ming Xiang[1] Quan Bai[1] William Liu[1,a)]

Received: June 27, 2013, Accepted: January 8, 2014

Abstract: Smart Grid is the trend of next generation electrical power system which makes the power grid intelligent and energy efficient. It requires high level of network reliability to support the two-way communication among electrical services, electrical units such as smart meters, and applications. The wireless mesh network infrastructure can provide redundant routes for the Smart Grid communication network to ensure the network availability. Also due to its high level of flexibility and scalability features that make it become a promising solution for Smart Grid. However, similar with many other distributed ad-hoc networks, trust is a critical issue for wireless mesh networks. In this paper, we proposed a novel trust-based geographical routing protocol, named as Dynamic Trust Elective Geo Routing (DTEGR), which allows peers in a Smart Grid system to adjust their interaction behaviors based on the trustworthiness of others. The simulation studies have confirmed that DTEGR can achieve better routing performance in different network scenarios, and also to achieve high level of reliable data transmission in Smart Grid communication networks.

Keywords: Smart Grid, wireless mesh network, trust-based geographical routing, vulnerabilities and attacks, energy efficiency

1. Introduction

Smart Grid is the trend of next generation electrical power system. It enables the functionalities of two-way communication among electrical services, electrical units, and applications. The National Institute of Standards and Technology (NIST) defines Smart Grid standard in seven domains, which are market, customer, service provider, bulk generation, distribution, operation, and transmission [1]. A Smart Grid communication network underpins and connects the seven domains together through data control transmission to enable the interactive operation so as to optimize resource allocation among power grid.

Compare to the traditional power grid, Smart Grid is the integration of telecommunication, information and power grid technologies. As the communication network becomes crucial in Smart Grid, a highly reliable and robust connection is required to support interactive operations between electrical services and applications. A wireless mesh network is a communications network made up of wireless nodes organized in a mesh topology which allows flexible interactions and communications. It is a promising infrastructure for Smart Grid which has embedded reliability and robustness in its mesh architecture. Moreover, it is low cost scalability, and flexibility [2]. Similar with many other distributed ad-hoc networks, trust is a critical issue of wireless mesh networks. When the network is under cyber-attacks, the interactions among peers will collapse, and overhead will be required. Therefore, although Smart Grid promises as a "green" technology which can save energy, without handling the trust issues properly, it may consume huge amount of energy by introducing interactive operation and control.

In Ref. [3], we highlight the importance of trust issues in Smart Grid systems, and initially introduced an energy aware geographical routing protocol to tackle these trust issues. This paper advances the Dynamic Trust Elective Geo Routing (DTEGR) protocol to extend our previous work and also provides more technical details of the DTEGR and comprehensive simulation results to validate its advantages.

The rest of paper is organized as follows. In Section 2 we give a brief introduction of typical Smart Grid communication network infrastructure and the major security concerns. Then a review of related work in the trust-based routing area is presented in Section 3. In Section 4, we present DTEGR algorithm in details, and then simulation and evaluation results are presented in Section 5. Finally, the conclusion and future work are drawn in Section 6.

2. Smart Grid Communication Network Infrastructure and Related Trust Issues

2.1 Network Infrastructure

IEEE has defined three foundational layers for Smart Grid, i.e., the energy and power layer, the communication layer, and the IT/computer layer [1]. In this paper, we focus on the communication layer. The hierarchical communication is a typical infrastructure adopted in Smart Grid communication network. The layers can be further classified by geographical size, i.e., Wide Area Network (WAN), Neighborhood Area Network (NAN), and Home Area Network (HAN) [4]. Also it can be classified by domain, i.e., HAN, Business Area Network (BAN), NAN, data center, substation automation integration system in Ref. [5].

[1] School of Computer and Mathematical Sciences, Auckland University of Technology, New Zealand
[a)] william.liu@aut.ac.nz

2.2 Trust Issues in Smart Grid Systems

Communication network is playing a crucial role in Smart Grid, which makes trust a crucial issue in Smart Grid. The research study in Ref. [6] defined that there are three major trust related concerns in the Smart Grid communication network. From most important to less important, they are network availability, data integrity, and information privacy.

Smart Grid communication network requires an uninterruptable connection to ensure the Smart Grid available all the time. The typical attacks are black, grey-hole attack, and flooding attack [6]. Black-hole attack is dropping all the received packets, while grey-hole attack is dropping some of the packets randomly or by purpose. These two cyber-attacks could disable the network and increase the energy consumption. In addition, the flooding attack is adversary sending huge amount of junk message to exhaust network resources and consume large amount of energy.

Additionally, the data integrity is defined as the data transmit in Smart Grid communication is intercepted by adversaries and manipulate without authorization. As monetary information involves in Smart Grid, this could cost huge financial loss, and also adversaries could take over the grid. Moreover, the Smart Grid is also holding end-users' profiles and this could be misused by adversaries for any purposes, namely information privacy concern.

3. Related Works

The study in Ref. [2] has recommended that wireless mesh network infrastructure for the Smart Grid communication network, and the study in Ref. [7] has also given many successful cases of wireless mesh network deployment in different cities in various forms. All of these are showing wireless mesh network infrastructure is a promising solution to Smart Grid. On the other hand, the inherent trust concerns of wireless mesh network infrastructure raise the trust issue on the nodes in the network. In Smart Grid communication network, these nodes are various electric units, e.g. smart meters. Trust is a critical issue for most ad-hoc networks and complex distributed systems [8], [9]. The studies in Refs. [10], [11], [12] defined trust is a belief or expectation on other parties' behavior without malicious intent, and this experience can be shared among the network of people. From this definition, there are two parts of behavior trust involved; those are direct trust and reputation. Direct trust is the direct experiences of target, where the reputation is the third parties experiences of target. In study [13], the combination of direct trust and reputation is in two major ways which are discrete model and the model based on the fuzzy logic. Discrete model is to use confident factors between direct trust and reputation to find out the final trust value, ATSR is a typical example of using this technique referenced in Ref. [3]. While the fuzzy logic approach is using value sets to describe the behavior trust then finally use different algorithms to calculate the final value, i.e., average of the value sets.

The ATSR is one of the typical trust-based routing protocols can be applied into wireless mesh network. It distributes trust management protocol which consists with direct trust and indirect trust (reputation) as metric to evaluate the behavior of target nodes. It has 8 metrics to measure the trust level of the target nodes, such as success forwarding ratio, data integrity, etc. These metrics can be considered as the expectation in the trust definition. There are 2 strategies in the ATSR model, which are direct trust and indirect trust. Direct trust is the expectation of target nodes from source node, which mean the trust evaluation is performed by source node. Indirect is the shared experience from source node's neighbors to the target nodes, which means the trust evaluation is performed by target nodes' neighbors. These 2 trust values will be putting together with a weight factor of each to become a final trust value in order to measure target nodes' trust level. After the trust level is determined, ATSR will calculate each of source node's neighbors' distance to the destination. The closer distance to the destination, neighbor will gain higher value in the distance metric. Moreover, this distance value and final trust value will be putting together again with a weight factor of each to become final value of target nodes. The nodes with highest final value will be selected as next hop forwarding node and eventually reach the destination.

In wireless mesh network topology, IP routing has 2 ways to find the path, either records the whole network which is link-state protocol, or using the opportunistic techniques, which is vector protocol. Geographical routing protocol uses the distance vector which is the geographical information to find the direction to the destination, so as to avoid the flooding of nodes state information. It make source node more efficient to locate the path to destination in a large scale network. GPSR [17] is the most typical geographical routing protocol that it consists with 2 strategies. They are greedy forwarding and perimeter forwarding. ATSR is using the greedy forwarding strategies that it selects the node with closest distance to the destination. When in a situation that source node is the one with closest distance to the destination within its radio range, greedy forwarding here will not work. And the 2nd strategy of GPSR perimeter forwarding will take part here. Perimeter forwarding is using the right-hand rule to select the next hop until the next hop with closer distance to the destination is found.

The study in Ref. [14] has proposed a trust-based routing framework, and this framework contains some nodes which are assumed being trusted to monitor its neighbors' behaviors within the radio range. The activities monitor log will be collected by trust builder for evaluation, and then trust builder will send a copy to reputation manager for a record. Therefore, this information can be used as reputation for other in the future. The method proposed in Ref. [15] is similar to ATSR protocol. However, rather than avoiding the malicious nodes, it blacklists the physical area where the malicious behaviors take place. As they think the malicious attacks normally take place in the same region.

4. Dynamic Trust Elective Geo Routing (DTEGR)

The Dynamic Trust Elective Geo Routing (DTEGR) protocol is a trust-based geographical routing protocol, which is inspired from the ATSR algorithm after we have identified the shortcoming of it. The ATSR algorithm is using static weight factors between distance metric and trust metric to evaluate the final score for each neighbor node. If the trust weight factor is too high, the distance metric will barely affect the final score to have the packet forwarded to the right destination. On the other hand, if

the distance weight factor is too high, the algorithm will hardly detect malicious nodes. It needs the trade-off between the detection sensitivity and distance, thus obviously a static weight factor is not the master key to optimize all the scenarios. In such case, we proposed the DTEGR protocol to split the evaluation into 2 steps. First of all, DTEGR will setup the trust value threshold h, if neighbor nodes' trust value are higher than h, these neighbors would be considered trustable and they will be in the trust forwarding list. Then the next step is to select the neighbor with closest distance to the destination as next hop from the trust forwarding list. In such case, we save one calculation operation from the trust metric and distance metric combination, namely the energy consumption for this operation can be optimized compare to ATSR algorithm.

$$t_{x,A} \geq h \Rightarrow n_{x,A} \in l_A \tag{1}$$

In Eq. (1), $n_{x,A}$ is neighbor of node A, $t_{x,A}$ is the trust value of $n_{x,A}$ in node A, h is the trust threshold value, and l_A is the trust forwarding list generated by node A.

In such case, the trade-off between trust and distance is no longer needed, but now the threshold value is an issue. If the threshold is too high, it can cause the trust forwarding list empty. Or if it is too low, it is possible to include the malicious node in the list. To tackle this problem, we propose a dynamic threshold algorithm in DTEGR. First of all, the maximum value of threshold needs to be determined by averaging of the good behavior nodes' trust value. The equation is below:

$$h_{\max} = \left(\frac{\sum_{i}^{n} t_i}{n} \right) - 0.1, \tag{2}$$

where $h_{\max}$ is the default trust threshold value for all the nodes in the network, n is the number of selected good behavior nodes, and t_i is the trust value. In such case, this threshold can make sure all the good behavior neighbors in the trust forwarding list. There is a possibility that some of the good nodes have bad performance by accident, and the trust forwarding list size could be decreasing over time because of this. The DTEGR can make sure there is sufficient choices in the list and also giving the second chance to the nodes have poor performance previously. When threshold is equal to $h_{\max}$ and trust forwarding list size less than 30% of number of neighbors, threshold value will drop by 0.1. This is to give the second chance to those nodes which have poor performance before. If those nodes performance well again, their trust values should be increased back to standard level again. However, after first decrease on threshold value, the trust forwarding list is still less than 30% then nothing will happen until the list is empty. When the list is empty, DTEGR will drop the threshold by 0.1 again until the list is not empty any more.

For the trust metric evaluation, it consists of direct trust metric and indirect trust metric. Direct trust metric is formulated as following.

$$t_{direct} = \frac{s}{s+f} \tag{3}$$

where s is total number of well performances of target node, and f is the total number of bad performances or namely malicious behaviors. In such case, every time neighbors have malicious behaviors, the algorithm will record down and thus the trust value will decrease. Indirect trust is the trust value obtains from other neighbor who knows the target node. The final trust value can be calculated using the following equation.

$$t_{final} = c_{indirect} \times t_{indirect} + (1 - c_{indirect}) \times t_{direct} \tag{4}$$

The confident factor for indirect trust indicates the confident of source which is $c_{indirect}$ in the Eq. (4). As the equation showed, the confident factor can determine how important the indirect trust will affect the final trust value.

After the trust forwarding list is generated, the distance metric algorithm in DTEGR will select the neighbor with shortest distance to the destination as next hop from the list. The distance equation is as following:

$$d = \sqrt{|x_n - x_d|^2 + |y_n - y_d|^2} \tag{5}$$

In Eq. (5), (x_n, y_n) and (x_d, y_d) is the longitude and latitude of neighbor and destination. DTEGR will select the neighbor from trust forwarding list with shortest distance d as next hop to forward the packets.

5. Simulation

In this section, we have carried out the simulation studies of DTEGR by using a Java based simulation tool named J-Sim [16]. The topology of wireless mesh network is assumed as a 10×10 grid network, namely 100 nodes (i.e., smart meters) as shown in **Fig. 1**.

There are 26 malicious nodes have been deployed in the network, (red dots in Fig. 1). All the malicious nodes will perform grey-hole attacks which they will drop the received packets randomly, except for node 12. Node 12 will drop all the received packets to conduct black-hole attacks. In the simulation, each scenario has 300 sessions will be preceded, and each session interval is 4 seconds. Each session will forward 1 UDP packet with 31 bytes data, and packet's time to live (TTL) is 128 milliseconds.

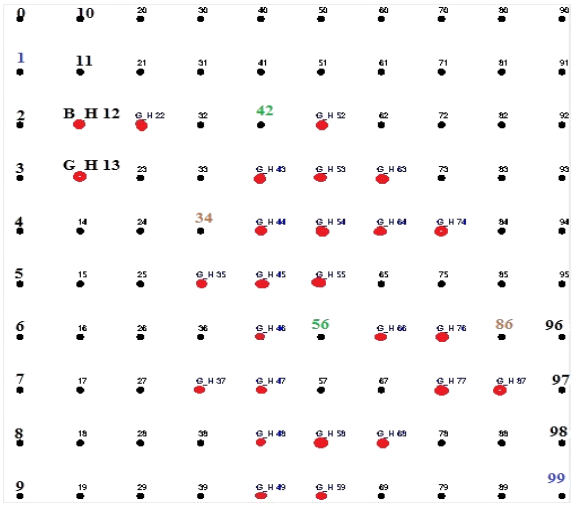

Fig. 1 10×10 wireless mesh network topology.

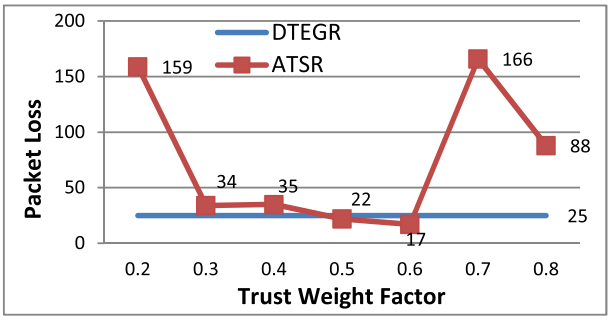

Fig. 2 Packets loss vs. trust weight factor.

Table 1 Mean latency vs. trust weight factor.

ATSR	ATSR Trust Weight Factor							DTEGR
	0.2	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	
Latency (ms)	11	13.3	15.88	15.76	15.86	34.58	16.87	13.42

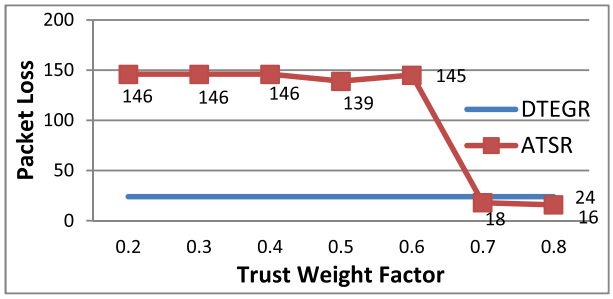

Fig. 3 Packets loss vs. trust weight factor 2.

Table 2 Mean latency vs. trust weight factor 2.

ATSR	ATSR Trust Weight Factor							DTEGR
	0.2	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	
Latency (ms)	2.83	2.83	2.83	2.837	4.37	4.46	6.32	6.3

5.1 The Comparison Between DTEGR and ATSR

There are three scenarios to compare DTEGR with ATSR algorithm, and each scenario has different node-pairs as source and destination nodes. ATSR protocol is conducted in each scenario to compare with DTEGR protocol. ATSR is conducted for 7 times with trust weight factor from 0.2 to 0.8, namely the distance weight factor is from 0.8 to 0.2. The trust weight factors $\{0, 0.1, 0.9, 1\}$ are not included because it is either disable the trust metric or distance metric. The DTEGR is only conducted once as it does not have weight factor issue, and the threshold will be adjusted automatically by the algorithm. The initial threshold value for DTEGR is 0.7 in this simulation as the average trust values of good behaves nodes is 0.8. We have the threshold value a little bit lower than the average good behaves node trust value so as to have sufficient neighbors in the trust forwarding list.

In the first scenario, the node-pair is node 1 and node 99 which we are targeting to investigate the path finding behavior of two algorithms under heavy cyber-attacks environment in longer distance. The simulation result is as following in **Fig. 2** and **Table 1**.

According to the simulation results in Fig. 2, the ATSR perform well when the trust weight factor between 0.3 and 0.6. When trust weight factor is set at 0.2 in ATSR, it made trust metric too little to alert the nodes from grey-hole attacks, while when trust weight factor is set at 0.7 and 0.8, it made the distance metric too small to affect the direction selection to destination node as there are many hop selections between node 1 and node 99. In this scenario, ATSR with trust weight factor at 0.6 has the best result as it has the least packets loss. DTEGR had lost 25 packets out of 300. Compared to the ATSR, the trust weight factor at 0.6 is slightly higher. However, if we look at the latency on Table 1, DTEGR has less delay which is 13.42 milliseconds, where ATSR took 15.86 milliseconds when trust weight factor at 0.6. From the mean packet latency performance, it can be found that DTEGR algorithm can select shorter path (less average hop count) to the destination comparing to ATSR. In addition, it has less average hop count to the destination which means use less energy in transmission as each node consume similar energy units to forward a packet. In scenario 1, DTEGR had similar performance on packet loss but consume less energy to complete the task compare to ATSR.

In scenario 2, the source is node 56 and destination is node 42, so as to see whether DTEGR algorithm will be adaptable enough to select the trustable path to destination by sacrificing the distance factor. The result is shown as following in **Fig. 3** and **Table 2**.

According to Fig. 3, ATSR with trust weight factor from 0.2 to 0.6 had more than 40% of packets loss in the scenario 2, which cause network collapse. The reason for high percentage packet loss is because node 56 was surrounded by malicious nodes, thus the trust weight factor was not high enough to avoid these malicious nodes. Those good behaved neighbors were too far away from the sink node compare to the malicious neighbors. The far away from distance means low distance value, and those malicious nodes were close enough to the destination to obtain enough point to ignore trust metric and won the first position in the ATSR evaluation. When trust weight factor at 0.7 and 0.8 in ATSR, it finally made the trust metric high enough to ignore the distance metric so the malicious neighbors can be avoided. In the scenario 2, node 56 and node 42 is not so far away that ATSR was still able to find a short path to the sink node when the distance metric became less important. Compare to scenario 1, when the trust weight factor at 0.7, ATSR was a little bit start getting lost to the sink node. While in scenario 2, ATSR has the best result when trust weight factor was at 0.7 as it takes lower mean latency, in other words less energy consumption. When the trust weight factor is 0.8, the packet loss is lower. DTEGR had lost 24 packets in this scenario which is slightly higher than ATSR, and its mean packet latency is at 6.3 milliseconds which indicate that it selected longer path to the sink node. In scenario 2, there were many malicious nodes appearing between node 56 and node 42, thus the secured paths were either traverse to the left or right so as to get around the malicious nodes and reach the destination. The left path was the shorter path, but the neighbor with closer distance to the destination was on the right hand side, and this was why DTEGR selected the longer path.

In scenario 3, we select node-pair (34, 86) as source and destination nodes respectively, so as to study whether the new DTEGR algorithm able to select a longer but trustworthy path in longer

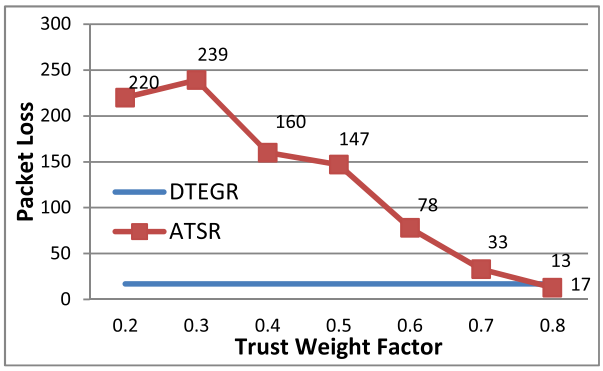

Fig. 4 Packets loss vs. trust weight factor scenario 3.

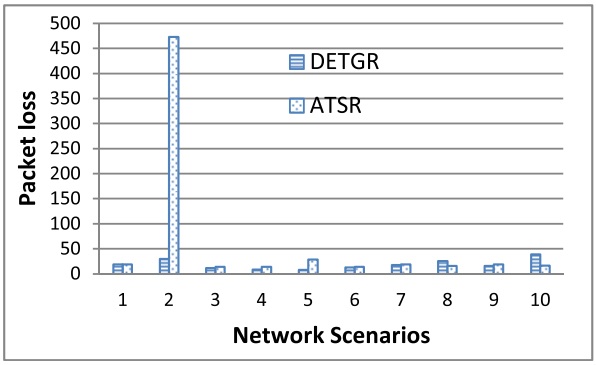

Fig. 5 DTEGR vs. ATSR under 50% malicious nodes attacks.

Table 3 Mean latency vs. trust weight factor.

ATSR	ATSR Trust Weight Factor							DTEGR
	0.2	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	
Latency (ms)	4.58	4.55	4.59	4.61	6.29	10.4	6.43	6.54

Table 4 Mean packet latency for DTEGR vs. ATSR under 50% attacks.

50% attacks	*Network Scenario*									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
DTEGR (ms)	13.36	14.33	13.34	10.97	11.09	13.41	13.35	15.91	13.4	13.36
ATSR (ms)	13.36	12.7	13.38	13.5	10.99	13.5	16.1	13.48	13.97	13.45

source and destination distance compare to 2nd scenario. The result is shown as following in **Fig. 4** and **Table 3**.

According to Fig. 3, the most suitable trust weight factor for ATSR was at 0.8 as it costs the least packet loss and less mean packet latency. The packet loss is over 70% when the trust weight factor was at 0.2 and 0.3. This was because the trust metric was too low to affect the algorithm to avoid 2 of malicious nodes in between source node and sink node. As the trust weight factor in ATSR is increasing, the packet loss e was dropping significantly as well. It was the same reason for the scenario 2, because source node and sink node were close enough to each other that the algorithm did not get lost when distance metric become less important. DTEGR had 17 packets lost out of 300 in the scenario 3, and mean packet latency was at 6.54 milliseconds which had similar performance to ATSR, but less packets loss.

From the three scenarios evaluated as above, it can be seen that ATSR is sensitive on the weight factors and need to configure different suitable trust weight factor for different scenarios. In scenario 1 was at 0.6, scenario 2 was at 0.7, and scenario 3 was at 0.8. This was a huge different in the trust weight factor selection. As the trust and distance weight factors in ATSR are static factors, it cannot use the same weight factor to tackle all the scenarios. For example, in scenario 1, 0.6 was the trust weight factor that can achieve best result, but if used 0.6 in scenario 2 or 3, which will cause the network collapse as there are over 40% of packet loss. DTEGR has testified that it has overcome the problem which ATSR was experienced. In all 3 scenarios, DTEGR was able to maintain the packet loss and latency level which ATSR require changes the weight factors manually to obtain the best performance.

5.2 The Stability of DTEGR

To further investigate the stability of DTEGR algorithm's performance to avoid malicious attacks, we setup different malicious attacks scenarios in the network, i.e., 30%, 40%, 50% malicious nodes of the network to compare with ATSR trust weight factor at 0.5. The 20% or less of malicious nodes is not sufficient to block the route between node 1 and node 99 to cause any packet loss in most of the scenarios as we have tried 10 random simulations where 6 has 0% packet loss. While the 60% or more of malicious nodes in the network is too many that it can completely block the route from the node 1 to node 99 in most of the scenarios. We also have tried the 60% scenario ten times, which 7 of the simulations have the malicious nodes completely blocked the path to sink node. We deploy the malicious nodes in random positions for ten times for each level of attacks. For each scenario, there are 900 traffic sessions to be proceeded as we are testing the performance stability for the DTEGR algorithm, and the interval is 1 second. The traffic is travel from node 1 to node 99.

First of all, we setup 50% of the malicious nodes deploy in the network randomly with ten simulations, and the simulation results are shown in below.

From **Fig. 5**, we can see the performance results in packet loss for DTEGR and ATSR algorithm were similar as the 0.5 for trust weight factor is an optimal trust weight factor for these random scenarios, except for scenarios 2, 5, and 10. In the scenario 2, ATSR has huge amount of packet loss was due to the trust weight factor was not high enough to enforce the algorithm detour to a longer route so as to avoid the malicious nodes attacks. This can be also seen in the **Table 4** packet latency, ATSR achieved lower packet latency compare to DTEGR it is because it selected a shorter route to the destination but cannot exclude the malicious nodes from the route. In scenario 5, ATSR algorithm resulted in a higher packet loss in the simulation. This is actually the same reason as in scenario 2, the trust weight factor was low that it sacrificed so many packets to make the algorithm decided to select a longer but secure route to the destination. Finally in the scenario 10, DTEGR algorithm has a higher packet loss compare to ATSR algorithm, this was because DTEGR algorithm is using trust threshold to determine whether the nodes are malicious or legitimate. In such case, the nodes have to bad enough to fall out the safe forwarding list so the algorithm can select another

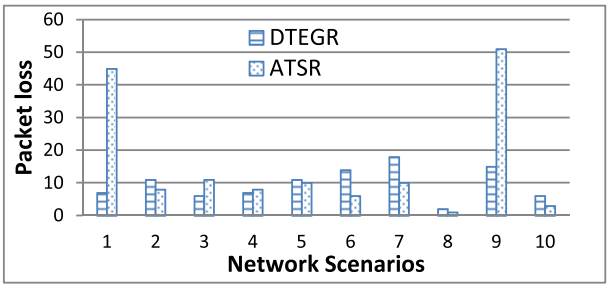

Fig. 6 DTEGR vs. ATSR under 40% malicious nodes attacks.

Table 5 Mean packet latency for DTEGR vs. ATSR under 40% attacks.

40% attacks	*Network Scenario*									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
DTEGR (ms)	12	13.25	11	11.26	13.28	13.44	13.47	11.05	13.31	11.06
ATSR (ms)	13.52	13.59	11.4	10.98	13.52	13.49	13.41	11.03	13.3	11.07

neighbor as next hop which with shorter distance to the destination. In ATSR algorithm case, if there are two nodes have similar distance to the destination, the change on trust metric value will make the ATSR algorithm switch the route very quickly. For example, node A and B has the same distance to the sink node, DTEGR and ATSR both first select node A as next hop to forward the packet. Node A start dropping packets after a while, ATSR will quickly switch the route to node B as node A and B has the same distance to sink node, as long as trust metric decrease, ATSR algorithm will switch. But for the DTEGR case, it has to wait for node A to drop more packets so as to have the trust metric value below the threshold, then DTEGR can exclude node A from the list and switch the route to node B. From this example, ATSR algorithm is actually detect and avoid the malicious quicker than DTEGR algorithm, and thus ATSR has less packet loss.

Next we decrease the attack level from 50% of malicious nodes in the network to 40% with the same network setup as last scenario. The results of this scenario are shown in below.

From **Fig. 6**, it can be seen that the packet loss results obviously were decreased in general under 40% of attack level rather than 50% of attack level. In the scenarios 1 and 9, ATSR had high packet loss number compare to DTEGR algorithm's performance in packet loss. Again, this is because the trust weight factor value not high enough, so it selects the shorter path but insecure route which causes large amount of packet loss. For the scenario 6 and 7, DTEGR algorithm had higher packet loss number compare to ATSR algorithm, this is the same reason in scenario 10 when under 50% malicious nodes attacks. DTEGR algorithm normally takes longer time to switch the neighbor as next hop compare to ATSR algorithm when this neighbor has similar distance as current next hop neighbor to the destination. But in overall performance for DTEGR and ATSR algorithm, DTEGR algorithm has more stable and better performance rather than like ATSR algorithm has high packet loss in scenarios 1 and 9.

Finally, we reduce the attack level again from 40% malicious nodes attacks to 30%, and the results are shown in below.

From **Fig. 7**, the scenario 7 had higher packet loss number

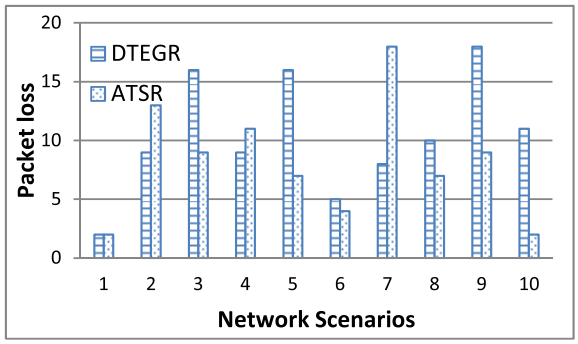

Fig. 7 DTEGR vs. ATSR under 30% malicious nodes attacks.

Table 6 Mean packet latency for DTEGR vs. ATSR under 30% attacks.

30% attacks	*Network Scenario*									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
DTEGR (ms)	12	13.25	11	11.26	13.28	13.44	13.47	11.05	13.31	11.06
ATSR (ms)	13.52	13.59	11.4	10.98	13.52	13.49	13.41	11.03	13.3	11.07

Table 7 DTEGR vs. ATSR on packet loss.

DTEGR/ATSR	*Attack level*		
	30%	**40%**	**50%**
DTEGR standard deviation	5.06	4.95	13.26
ATSR standard deviation	5.01	17.58	143.99
DTEGR average	10.4	9.7	19
ATSR average	8.2	15.3	63.4

with the same reason again as previous scenarios, which was the trust weight factor was not high enough. Also, for the DTEGR algorithms, in scenarios 3, 5, 9, and 10, packet loss number were high compare to ATSR's performance, it was also the same reasons as previous scenarios. But from these different scenarios' results under different levels of malicious nodes attacks, ATSR algorithm with trust weight factor at 0.5 was performing better and better as the attack level decrease.

As can be seen in the **Table 7**, when the attack level decreased, the standard deviation for packet loss on ATSR is getting better and better compared to DTEGR. This also has been reflected on the average packet loss such as when attack level was at 30%, ATSR actually perform slightly better than DTEGR algorithm. In such case, it can tell that ATSR trust weight factor at 0.5 can have better performance when attack level is low. Namely, trust weight factor at 0.5 is optimal value for the ATSR algorithm while the network is under low level of malicious attacks. As when attack level is low, it has higher probability of same distance alternative route to the destination, in such case ATSR algorithm do not require a high priority on trust metric to select a further route so as to avoid the malicious nodes to final reach the destination. When the attack level is high, it will require higher trust value of weight factor; otherwise it has worse performance in scenario 2 upon 50% attack level. While DTEGR able to detect every malicious node in the route and avoid them without adjusting any parameter. DTEGR is performing better when under heavy attack as the trust threshold clearly defines the malicious nodes and legitimate nodes, then avoid the malicious nodes. But ATSR algo-

rithm do not have a clear definition on which one is malicious and which one is legitimate, it just select the neighbor with highest final score which come from distance and trust metric together. In such case, ATSR algorithm is more sensitive to the change of trust metric value so it can detect and avoid the malicious nodes in an earlier stage to save the packet loss. There are two factors to affect the results of packet loss number for both algorithms. First is mention before which is the speed of detect and avoid the malicious nodes, obviously, the faster it happen, the less packet loss will happen. The second factor is the attack level, when the attack level is low, while the algorithms are finding the alternative route, they have less chance to encounter the malicious nodes and cause the packet loss. On the other way round, when the attack level is high, the algorithms are more likely to encounter more malicious nodes to cause packet loss while they are finding the alternative route. In As ATSR will switch the route quicker once encounter malicious attack compare to DTEGR, and under light attack level in the network, the switch of the route is less likely to encounter the malicious nodes again. In such case, ATSR algorithm is more likely to achieve less packet loss at the end compare to DTEGR. On the other hand, while the network is under heavy attack, ATSR still able to switch the route quickly once encounter the malicious attack. But this time ATSR is more likely to encounter another malicious node after the switch, then ATSR quick switch back to the previous malicious node that it creates more packet loss. In such case, ATSR algorithm requires a higher value in trust weight factor, so the trust weight factor has higher priority to detour to a further but a secured route. In DTEGR algorithm, it might take a while to have trust metric collect enough bad feedbacks to let the malicious nodes to lose their position in the safe forwarding list, so the algorithm can avoid them. This can cause higher packet loss compare the ATSR algorithm. But once these malicious nodes lose their position in the safe forwarding list, the algorithm will not select them again until the safe forwarding list trend to empty, and this can prevent the packet loss reoccurred on the same malicious nodes. This situation is more likely happen for ATSR in the network which is under heavy attack. In such case, ATSR normally have better performance in low attack level network scenarios while the trust weight factor is at 0.5, and DTEGR is likely to have a better performance under heavy attacks. The network under heavy attack requires ATSR algorithm assign a higher trust weight factor to have better performance in packet loss. In conclusion, DTEGR resolve the trust weight factor selection problem in ATSR algorithm, and it can maintain the good performance level upon different network scenarios without any parameter adjustment, where ATSR algorithm requires extra process to find optimal factors for each scenario to achieve such performance.

5.3 Energy Consumption

In this scenario, we introduced the energy-aware functionality to DTEGR algorithm. The DTEGR algorithm has 2 steps, the first step is to use trust metric to generate a safe forwarding list for next hop selection, and second step will find out the neighbor in the list with shortest distance to the destination. DTEGR algorithm resolved the weight factor selection problem between trust and distance metric in ATSR. But here we have to again put the weight factor between energy and distance metric so as to emerge the energy-aware functionality into the algorithm. Namely, the energy-aware DTEGR algorithm is at the second step using weight factor technique to combine it with distance metric. We set the traffic sessions as 900, and focus on node 55's battery life, so as to see whether energy metric can extend node 55's battery life time. We were using the energy weight factor value sample set $\{0, 0.2, 0.4, 0.6\}$ respectively in this scenario, and the results are shown in below.

Table 8 DTEGR with energy metric.

Energy weight factor	*0.0*	*0.2*	*0.4*	*0.6*
Life time on node 55 (sec)	671.59	682.21	692.41	646.48
Mean Packet Latency (ms)	11.51	11	12.63	23.48
Packet loss	17	5	0	192

As can be seen in the **Table 8**, as the increment on energy weight factor, the life time of node 55 was increasing accordingly until reach 0.6. This is because after 0.6, the energy weight factor was too high to have the distance metric look for the direction to destination for the algorithm. In such case, the algorithm was more likely to find the next hop by the energy metric rather than the distance and this make the algorithm select a very long route to the destination so as to consume more energy (i.e., from overall network point of view) to achieve same task. This can be seen in the packet latency metric which was very high. Moreover, the high packet loss number and packet latency when energy weight factor was at 0.6, this is due to the nodes in the center of the network have the batteries exhausted before the 900 traffic sessions were completed one by one. The high priority in energy metric that it makes the DTEGR algorithm select a long distance route to avoid those low batteries nodes, and at the end it even can't find the way to the destination that cause the high packet lost. In such case, there was no valid route to the destination that causes the packet loss. While the energy weight factor was at 0.4, the battery life on node 55 was extended by about 20 seconds, but at the same time, the packet latency was increased due to the energy load balance. The energy metric in DTEGR algorithm try to select alternative routes to destination rather than a single route, so as to achieve energy load balance. From the packet loss point of view, the packet loss number was decreasing accordingly due to the energy metric affect the algorithm to avoid the low battery nodes before they die. As can be seen that, while energy metric was turned off, it cause 17 packet loss due to flatten batteries. While the energy metric was at 0.4, there was no packet loss at all. The results are indicating the energy metric can help algorithm predict empty batteries node so as to avoid them before they exhaust to save the packet loss.

In the last scenario for the DTEGR algorithm, we deploy 26 malicious nodes into the network to perform grey-hole attacks. The attacks will be performed at the beginning of the simulation all at the same time. The network setup is same as figure 4.2 in the TIGER simulation section. There are 900 traffic sessions will be proceeded, and each traffic session is 1 second interval. We use the energy weight factor sample value sets $\{0, 0.1, 0.2, 0.3, 0.4\}$.

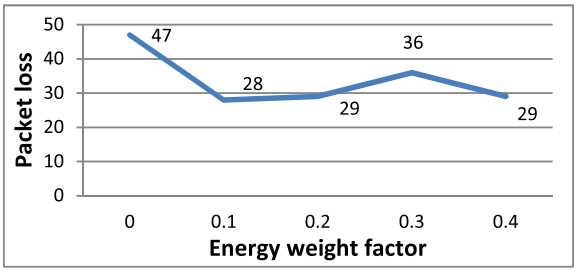

Fig. 8 Energy metric cost in DTEGR packet loss.

Table 9 Mean packet latency for DTEGR.

Means packet latency (ms)	*Energy weight factor*				
	0.0	**0.1**	**0.2**	**0.3**	**0.4**
DTEGR	13.5	15.5	13.32	14.45	17.01

The result are shown in the below.

From **Fig. 8**, we can see that, once the energy metric was turn on, the packet loss number were similar between different energy weight factors. When the energy metric was turn off, the high packet loss result was caused by the flat battery nodes, as almost half of packet loss were due to this reason. The J-Sim Tools has shown every process of the traffic session in the simulation. It is easy to find out what cause the packet loss. In such case, when the energy metric was turn off, excluded the packet loss number which caused by the exhausted battery, the packet lost number were similar to when energy metric was turn on. These results have confirmed the energy metric did not affect the performance of trust metric in DTEGR algorithm. As DTEGR is a two steps algorithm, it first has the trust metric to filer the malicious nodes out of the safe forwarding list, then use the energy and distance to select the neighbor as next hop to forward the packet. In such case, the energy metric cannot affect the trust metric. The packet latency results in **Table 9** have also confirmed the sacrifice on the distance to achieve energy load balance. After the energy metric was turn on, the packet latency results were increased except while energy weight factor was at 0.2. The reason for low packet latency result when energy weight factor was at 0.2 is because of the energy load balance feature in the algorithm. In the network, node 21 had the default choice for the next hop is node 33 in DTEGR algorithm while energy metric is turn off as node 33 is closer to the destination. But when the energy is turn on in the DTEGR algorithm, as the energy weight factor increase, the algorithm will more prefer node 42 as next hop. This is because node 42 is closer to the edge of the network compare to node 33. In such case, the energy consumption per second is less than node 33. In this network scenario, route through node 42 actually is a closer secured route to the destination rather than through node 33. This is the reason why the packet latency is lower while energy weight factor at 0.2.

6. Conclusion

In this paper, we proposed the DTEGR protocol which can adapt and ensure network availability in various scenarios. It is a trust-based geographical routing protocol that can achieve energy efficient by avoiding packets loss and searching for shorter path from various cyber-attacks. The extensive simulation studies confirm that DTEGR algorithm able to resolve the weight factor selection problem between trust metric and distance metric in ATSR algorithm by introducing the two stages strategy. The first stage it uses trust metric with trust threshold to generate a safe forwarding list, then the second stage is to use the distance metric to select the next hop from that safe forwarding list. We also setup more scenarios with different malicious attack levels to verify the stability of DTEGR algorithm. We found that DTEGR is able to maintain the performance level through different network scenarios, and better performance under heavy malicious attack compare to ATSR algorithm. ATSR algorithm requires adjusting different trust weight factor so as to perform well in different attack level scenarios, where DTEGR does not need to have any adjustment on any parameter. Moreover, the energy-aware functionality in DTEGR requires a weight factor to combine it with distance metric. Through the extensive simulation studies on the energy-aware functionality in DTEGR, it confirmed the energy metric able to help DTEGR achieve the energy load balance, so as to extend the batteries life for the nodes in the network. But at the same time, this energy-aware functionality sacrificed the distance metric performance to achieve the energy load balance. In such case, the selection of optimal weight factor between energy and distance metric is another problem waiting to be resolved.

In the future, more complex traffic flow scenarios and various network topologies will be considered to evaluate the performance of DTEGR.

Acknowledgments The authors would like to thank the anonymous reviewers for their comments that help improve the manuscript.

References

[1] Smart Grid Conceptual Model, available from ⟨http://smartgrid.ieee.org/ieee-smart-grid/smart-grid-conceptual-model⟩.
[2] Geelen, D., Kempen, G.V., Hoogstraten, F.V. and Liotta, A.: A wireless mesh communication protocol for smart-metering, *Computing, Networking and Communications* (*ICNC*) *International Conference*, pp.343–349 (2012).
[3] Xiang, M., Bai, Q. and Liu, W.: Self-Adjustable Trust-Based Energy Efficient Routing for Smart Grid Systems, *WI-IAT 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, Vol.3, pp.378–382 (2012).
[4] Zhang, Y., Sun, W., Wang, L., Wang, H., Green, R.C. and Lam, M.: A Multi-Level Communication Architecture of Smart Grid Based on Congestion Aware Wireless Mesh Network, *North American Power Symposium* (*NAPS*), pp.1–6 (2011).
[5] Yu, R., Zhang, Y., Gjessing, S., Yuen, C., Xie, S. and Guizani, M.: Congitive radio based hierachical communications infrastructure for smart grid, *IEEE Network*, Vol.25, No.5, pp.6–14 (Sep.-Oct. 2011).
[6] Yan, Y., Qian, Y. and Tipper, D.: A Survey on Cyber Security for Smart Grid Communications, *IEEE Communications Surveys & Tutorials*, Vol.14, No.4, pp.998–1010 (2012).
[7] Vasilakis, G., Perantinos, G., Askoxylakis, I., Mechin, N., Spitadakis, V. and Traganitis, A.: Business opportunities and considerations on wireless mesh networks, *IEEE WoWMoM* (2009).
[8] Ramchurn, S.D., Huynh, D. and Jennings, N.R.: Trust in Multiagent Systems, *The Knowledge Engineering Review*, Vol.19, pp.1–25 (2004).
[9] Li, H. and Singhal, M.: Trust Management in Distributed Systems, *Computer*, Vol.40, No.2, pp.45–53 (2007).
[10] Josang, A.: The Right Type of Trust for Distributed Systems, *Proc. ACM New Security Paradigms Workshop*, pp.119–131 (1996).
[11] Mayer, R.C., Davis, J.H. and Schoorman, F.D.: An Integrative Model of Organizational Trust, *Proc. 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Vol.20, No.3, pp.709–734 (1995).
[12] Denning, D.: A new Paradigm for Trust Systems, *Proc. ACM New*

Security Paradigms Workshop, pp.36–41 (1993).
[13] Brinklov, M. and Sharp, R.: Incremental Trust in Grid Computing, *Cluster Computing and the Grid*, pp.135–144 (May 2007).
[14] Marias, G.F., Tsetsos, V., Sekkas, O. and Georgiadis, P.: Performance evaluation of a self- evolving trust building framework, *Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks*, pp.132–141 (Sep. 2005).
[15] Tanachaiwiwat, S., Dave, P., Bhindwale, R. and Helmy, A.: Location-centric isolation of misbehavior and trust routing in energy-constrained sensor networks, *Performance, Computing, and Communications, IEEE International Conference*, pp.463–469 (2004).
[16] J-Sim, available from ⟨https://sites.google.com/site/jsimofficial/⟩.
[17] Karp, B. and Kung, H.T.: GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, *MobiCom '00 Proc. 6th Annual International Conference on Mobile Computing and Networking*, pp.243–254 (2000).

Ming Xiang received his Master of Computing and Information Science degree from Auckland University of Technology in 2013. His research interests are trust-based routing, Smart Grid and wireless mesh network.

Quan Bai received his Ph.D. in Computer Science at the University of Wollongong in 2007. He is presently a senior lecturer at School of Computer and Mathematical Sciences, Auckland University of Technology. He specialises in multi-agent coordination, trust-based computing and service composition.

William Liu received his Ph.D. and Master (with distinction) in Electrical and Computer Engineering at the University of Canterbury, Christchurch, New Zealand. He is currently a lecturer at School of Computer and Mathematical Sciences, Auckland University of Technology. His research interests include network survivability, sustainability and trustworthy computing.