Comparison of data forwarding mechanisms for AMI networks

Conference Paper · January 2012

DOI: 10.1109/ISGT.2012.6175683

CITATIONS
21

READS
55

3 authors, including:

Sandra Céspedes
University of Chile
44 PUBLICATIONS **295** CITATIONS
SEE PROFILE

Alvaro Cardenas
University of Texas at Dallas
107 PUBLICATIONS **2,725** CITATIONS
SEE PROFILE

Comparison of Data Forwarding Mechanisms for AMI Networks

Sandra Céspedes, *Student Member, IEEE,* Alvaro A. Cárdenas, *Member, IEEE,* Tadashige Iwao

***Abstract*—Advance Metering Infrastructure (AMI) networks are often deployed under challenging and unreliable conditions. One of the issues for the transmission of data packet in these unreliable networks is the routing of packets, because routing paths may behave differently from the time when the route is discovered to the time when a data packet is forwarded. In addition, control packets may get lost and give routers an inconsistent view of the network.**

While previous research has focused on designing the control-plane of routing protocols to deal with the AMI network conditions, there is comparatively a smaller amount of research on the advantages of new data forwarding mechanisms designed for unreliable networks.

This paper introduces a set of data forwarding mechanisms inspired by distributed depth-first search algorithms, and designed for the challenging conditions of large-scale unreliable networks envisioned by smart-grid deployments. These forwarding mechanisms use data packets to detect loops, update routing tables, and perform rerouting of data packets through alternate paths, recovering thus, packets that would have been normally dropped due to failures at the link layer. We perform simulations based on a real field AMI deployment to evaluate the performance of the proposed mechanisms. We also provide the evaluation results for the data forwarding mechanisms that have been implemented in a real AMI network.

***Index Terms*—AMI networks, data forwarding, depth first search, routing, smart grid.**

I. Introduction

Wireless mesh networks are one of the technology solutions for smart grid deployments such as Advanced Metering Infrastructure (AMI) and Transmission/Distribution Automation. The deployment of these mesh networks is placing new constraints and requirements on the technology required to maintain them. Not only are these networks large (requiring thousands of nodes in mesh topologies) but they are also deployed in highly unreliable environments, making the routing protocol that maintains these large networks a critical technology for the advancement of the smart grid.

In last year's SmartGridComm several routing protocols were analyzed to determine their suitability for these networks [1], [2], [3], [4], [5]. Each of these protocols proposes promising contributions to advance the deployment of large wireless networks. However, an often forgotten part of these protocols is the improvements that can be enabled by new forwarding mechanisms.

Routing protocols are generally composed of two independent phases, the control plane and the data forwarding plane. The control plane discovers and maintains routes, and the data forwarding plane performs a table lookup operation on information (generally) gathered by the control plane to forward the packet toward the destination. Most of the routing protocols focus on the control plane, and the data forwarding plane is left as an afterthought or as a choice for implementers. Moreover, in unreliable networks, the control overhead for detecting routing errors and for fixing paths happens often, so it is important to avoid expensive control plane mechanisms that might overreact in the presence of instability.

As the networks expected to be deployed in the Smart Grid become more complex and unreliable, routing algorithms are starting to include new options in their data forwarding plane. For example, the newly ratified ITEF standard RPL [2], [1] has a data forwarding plane with two options: 1) it uses data packets to detect inconsistencies (e.g., loops), and 2) it has a forwarding error flag that can potentially be used to reroute packets that cannot advance toward the destination. These forwarding mechanisms can improve the reliability of the network and show a lot of promise; however, even the latest RPL draft mentions that the standard document is a routing protocol, not a forwarding protocol, and thus the forwarding options are non-normative and not fully specified.

In this paper we consider more sophisticated forwarding options that can increase the reliability of networks. We propose 5 different forwarding mechanisms, compare their advantages and disadvantages, and identify which forwarding mechanisms can lead to more reliable networks. We perform simulations based in real AMI deployment data and show results from a field test. In general we find that new data forwarding mechanisms can improve the performance of large wireless mesh networks.

In previous work [4], we introduced DADR, a routing protocol for unreliable networks which included an initial proposal for a data forwarding mechanism. While this previous work focuses on the description and evaluation of a routing protocol following a specific data forwarding mechanism, this paper gives a detailed evaluation of different options and algorithms applicable exclusively to the data forwarding plane (and thus independent of specific routing protocols). We also include a new implementation of the data forwarding mechanisms in Omnest, a robust and general network simulation tool (as opposed to the previous work which had used a

S. Céspedes is with the Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada, and with the Department of Communications and Information Technology, Icesi University, Cali, Colombia email: slcesped@bbcr.uwaterloo.ca

A. Cárdenas is with the Trusted Systems Innovation Group, Fujitsu Laboratories of America, Sunnyvale, CA, USA email:alvaro.cardenas-mora@us.fujitsu.com

T. Iwao is with the Smart Network Division, Fujitsu Limited, Fukuoka, Japan email:iwao@jp.fujitsu.com

978-1-4577-2159-5/12/$31.00 ©2011 IEEE

simple simulation engine developed internally). An additional contribution of this paper compared to [4] is the inclusion of real-world data from an AMI deployment in New Mexico.

The remainder of this paper is organized as follows. Section II introduces the different data forwarding mechanisms. The characteristics of the AMI network employed in the evaluation of the data forwarding mechanisms are described in section III. Sections IV and V discuss the simulation and field test results respectively. Finally, concluding remarks are provided in section VI.

II. Data Forwarding Mechanisms

In this section, we describe five different data forwarding mechanisms for unreliable networks. It is assumed that a control plane exists and is in charge of finding the paths to other nodes in the network. In general, the following data structures are required in each node for the operation of these mechanisms:

1) Routing Table: for storing destinations in the network. This table allows up to K possible next hops to reach a destination.

2) Loop detection table: for storing records of the packets forwarded by the node and the address of the node from which it is received. It is assumed that packets have a unique identifier that serves to determine if they have been previously registered in this table.

As long as the control plane fills up the routing table, the mechanisms described in this section work independent of the underlying routing protocol (and more specifically, its control plane). Examples of protocols that could be used in the control plane are RPL [6] and the Babel routing protocol [7].

Additionally, since it is assumed the network presents unreliable channel conditions and node failures, it is expected the topology to be continually changing with possible loops appearing in the routing table. Hence, it is not required by these mechanisms that the control plane maintains an updated routing table at all times. Instead, we allow the control plane to be light and we shift part of the responsibility of fixing paths to some of the data forwarding mechanisms.

The algorithm followed for the rerouting of data packets in the following mechanisms is inspired in a distributed depth-first search algorithm over the network graph [8][9], as we explain as follows.

A. Simple Forwarding

This forwarding mechanism performs a lookup in the routing table and selects the best candidate to reach the destination of the data packet. It is simple in the sense that it does not keep record of the forwarded packets, hence it does not try to do rerouting when a loop exists and the packet returns to the node, nor it reacts to failures at the link layer. For the case in which the destination of a packet is not found in the routing table, the node proceeds to drop the packet.

It is a light-weight forwarding mechanism that does not employ extra space from the node's memory and that relies completely on the quality of paths and candidates gathered by the control plane. Therefore, if paths contain loops or the link layer is highly unreliable, this mechanism does not guarantee full delivery of packets.

TABLE I
Routing table entry for destination D

T		**U**		**X**	
Destination	D	Destination	D	Destination	D
Next Hop 1	U	Next Hop 1	W	Next Hop 1	Y
Cost (d)	20	Cost (d)	10	Cost (d)	10
Next Hop 2	X	Next Hop 2	X	Next Hop 2	Z
Cost (d)	40	Cost (d)	15	Cost (d)	15
Next Hop 3	S	Next Hop 3	T	Next Hop 3	U
Cost (d)	40	Cost (d)	30	Cost (d)	15
		Next Hop 3	V	Next Hop 3	T
		Cost (d)	50	Cost (d)	70

B. Loop Detection

In this mechanism, the node stores in the loop detection table an identifier for previously seen packets (e.g,. a sequence number created by the source concatenated with the source address) and the address of the node from which they are initially received (i.e, the previous hop). By means of this record, every time a packet is returned, the node identifies it and reroutes it through the next candidate selected from the routing table, so that it resembles a depth-first search over the network. There are a maximum of K tries if paths through all the candidates (i.e, the children) have a loop and the packet is returned. At last, the packet is sent back to the previous hop (i.e, the father) when the K trials have failed to deliver the packet.

Topology in Fig. 1 illustrates the loop detection logic. Assume all links are bidirectional and each node stores up to $K = 4$ candidates per destination. Fig. 1 shows the initial state of the network after all the signaling messages have been exchanged in the control plane, and Table I shows the entry for destination D in routing tables for nodes T, U, and X.

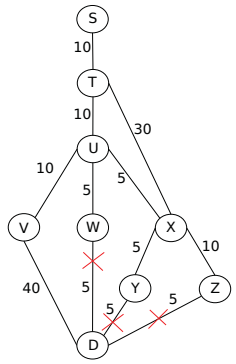


Fig. 1. Network topology

In a later stage, several nodes have lost their link to node D. Assume that node S wants to send a data packet to node D but the intermediate nodes have not updated their routing tables to reflect the new changes. The loop detection logic then operates in the following way:

- Node S registers the data packet in the loop detection table and puts itself as the previous hop. It forwards the packet to T.
- Node T makes a lookup of the packet in its loop detection table. Since it is the first time it receives it, a new record

is created. It sets node S as the previous hop, and selects nodes U and X as the candidates. Packet is forwarded to U. Note that T does not include S as a possible candidate, since it is registered as the previous hop.
- Node U makes a lookup and registers the packet in the loop detection table for the first time. It sets node T as the previous hop and selects nodes W, X, and V as the candidates. Packet is forwarded to node W.
- Node W does not have a valid path to D, so it returns the packet to node U. Once U finds the packet registered in the loop detection table, it is forwarded to the next candidate, node X.
- Node X makes a lookup and registers the packet in the loop detection table for the first time. It sets node U as the previous hop and selects nodes Y, Z, and T as the candidates. Packet is forwarded to node Y.
- X detects a loop when Y returns the packet, so it tries forwarding the packet to Z. Node Z also returns the packet, so the packet is forwarded to the last candidate T.
- T detects a loop through U; therefore it sends the packet to the second option: node X.
- Node X once more detects a loop, so it sends the packet back to the previous hop: node U.
- Node U detects a loop through node X. It forwards the packet to V, which delivers it to the final destination.

This method improves reliability thanks to the rerouting process. Furthermore, when a loop is detected, the node updates its routing table by poisoning the entry for the specific candidate that causes the loop. In this way, not only the node avoids to use the same failing path in the future, but also helps other nodes in the network to progressively remove it, since the update is later disseminated to neighboring nodes by the control plane.

Thanks to the feedback received from the loop detection mechanism, there is no need for the control plane to overreact when links or nodes in the network are unstable. In this way, the control plane can rely on the information received from loop detection once the data is actually being sent, therefore, avoiding the waste of resources for fixing paths that are not commonly used in the network.

The downside of this mechanism comes with the necessity of reserving memory space for the storage of the loop detection table. Moreover, depending on the data rate, this table can fill up quickly, and new packets would have to be forwarded without being registered, increasing the chances of loops to go undetected. One way to manage this situation is by setting a timer for entries to be deleted from the table as soon as possible. If a good estimate exists for the average end-to-end delay, this could be used as a baseline for the timeout of registered packets.

C. Loop on demand

This mechanism works is an extension to *simple forwarding* as described in section II-A. It works the same as simple forwarding except when a route towards the destination is not found in the routing table. In that case, the node proceeds to activate a flag for loop detection in the data packet, and returns the packet to the previous node. Since the loop detection flag has been activated, when the previous node receives the packet, it initiates the loop detection logic by registering the packet and selecting candidates to reroute it in case of loops. Therefore, the mechanism potentially increases the chances of delivery for packets that would have been dropped otherwise.

For nodes supporting this mechanism, they are required to have space reserved for a loop detection table, so that packets can be registered in the potential case that loop detection is initiated at the source or at any intermediate node in the path to destination.

D. Reliable Delivery

This forwarding method employs the loop detection logic described in section II-B. In addition to rerouting, it also reacts to packets losses notified by the MAC layer [10]. When a packet loss occurs, the node follows the same logic as if a loop is detected, which means, it selects the next-hop candidate in the list and reroutes the packet towards it. However, in addition to rerouting, the node activates a duplicate flag in the packet being re-sent, which remains active from that point onwards. This flag is necessary since the original packet may have been successfully received by the next hop, but the acknowledgment at the link-layer may have been lost. Therefore, it is important for intermediate nodes to identify when a received packet is a duplicate, so that they consider the packet coming from a retransmission instead of a loop in order to prevent poisoning routes that might not have loops.

Every time a node receives a packet with the duplicate flag activated (including the node that sets the flag), and the packet already exists in the loop detection table, it refrains from poisoning the last attempted candidate, since this may correspond to a false loop detection. In this way, the node accounts for temporary network congestions that can cause losses of packets or acknowledgments at the MAC layer, while it keeps trying to deliver the packet. There is one case when nodes poison the routing table based on a data packet with a duplicate flag on: the routing table is poisoned when a *return flag* is set. The return flag indicates when a packet has been returned to the previous node because it failed to forward the packet after trying (and failing) with all the possible next hops. This flag is updated hop-by-hop–i.e., after the parent receives a packet with the return flag on, and it forwards it to any of its children, then the flag is turned off.

The reliability of this mechanism is expected to be high, since it recovers losses for failures not only at the routing layer (e.g., loops), but also at the MAC layer (e.g., link failures). Even for cases in which the MAC layer does not operate using acknowledgments, it could react for packets discarded due to the inability to access the wireless channel (e.g., a 802.15.4 MAC layer that gives up after several tries to transmit a frame).

The tradeoff is the generation of duplicate packets that would increase the load in the network. Particularly, in unreliable networks with unidirectional links, a large number of packets are retransmitted as duplicates due to loss of acknowledgments, even if the original packets are already moving toward the destination.

E. Depth-first search (DFS)

The purpose of this forwarding method is essentially the same of reliable delivery. However, in the loop detection logic described in section II-B, a packet that returns to a node might return from a node different from the last attempted candidate. Despite of that, the last attempted candidate is still poisoned once the loop is detected and the packet is rerouted through the next candidate. There may be cases in which this would cause the depth-first search through the poisoned candidate to be stopped before time, leaving some nodes unvisited and possibly an alternative route without being used.

To depict this situation we use the topology in Fig. 2. Assume all links are bidirectional and each node stores up to $K = 3$ candidates per destination. The network has converged, and in a later stage, the link between V and D fails. Assume that node S wants to send a data packet to node D but the intermediate nodes have not updated their routing tables to reflect the new changes. Node S forwards to node X, which has $\{V$:2,T:3,U:3$\}$ as the candidates and costs to reach node D. X forwards the packet to node V, which has $\{D$:1,X:3,W:5$\}$ for reaching D. Then, after node V fails to directly send the packet to D, it reroutes the packet through W (node X is registered as the previous hop). At that point, the entry for destination D in W's routing table is $\{X$:3,V:3,D:4$\}$. Therefore, W forwards the packet to X, and X proceeds to reroute the packet through U while it poisons the path through V. As a result, node W does not continue the trials with the remaining option D.

The above-mentioned anomaly appears due to the limited size for the list of candidates. Thus, node X does not register W as a possible candidate (i.e., there are three other neighbors with better routes), although node W does include X in its list. This unbalanced situation is represented in Fig. 2 by the arrow in the link between X and W–Note that we assume the routing protocol only uses links that are bidirectional at the link layer, and thus there exists a bidirectional link between X and W; however, due to memory constraints in the routing table, X cannot store W as a candidate next hop towards the destination, therefore the arrow in the figure is only a *logical one way link during the depth first search* but does not represent a unidirectional link at the physical layer.

In order to do an exhaustive depth-first search, in this forwarding mechanism a node always checks if the packet for which the loop has been detected comes directly from the last attempted node. If that is not the case, the packet is sent back to the node that just sent it, so that the search can continue. If the packet does come from the last attempted node, that node is poisoned and the rerouting process continues normally.

III. Network Characteristics

We adapted a real field deployment of an AMI network to be used in the performance evaluation of the proposed data forwarding mechanisms (Fig. 10). We have extracted 6 different network sizes from there, with the largest network having a maximum distance of 2Km to the collector, and a total of 448 meters, 17 routers, and 1 collector. From all the nodes in the network, only meters and collectors generate

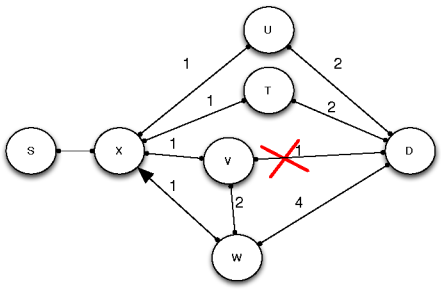


Fig. 2. Example of the loop detection logic with a truncated depth-first search

and process data packets. On the other hand, routers serve to increase network connectivity for meters located far away from the collector.

The network is simulated with the OMNEST simulation tool, and we use MiXiM as the framework to simulate the MAC and PHY layers of the wireless sensor networks. Nodes are equipped with a Texas Instruments CC 2420 802.15.4 network interface card that uses a non-beacon CSMA protocol as specified in IEEE 802.15.4-2006. Meters transmit at a power of 50mW, whereas routers and collector transmit at 200mW. In order to resemble the network characteristics of the field deployment, we implemented a Log-Distance path loss model that adds attenuation based on the following formula: $PLd0 - 10 \times pathLossExponent \times log_{10}(distance/d0)$, where $PLd0 = -55dBm$, $pathLossExponent = 2.4$, and $d0 = 1m$. We also set the thermal noise to $-107dBm$ and the sensitivity to $-110dBm$.

In order to test the channel conditions of the simulated network, we performed a channel test in which every node sends a total of 100 broadcast packets at a rate of 1 per second. During one node's transmission, all the other nodes remain silent in order to minimize the interference over the channel. Fig. 3 illustrates the results obtained for this test. We use the Packet Success Rate (PSR) to show the ratio of broadcast messages successfully received by neighbors of every node in the network.

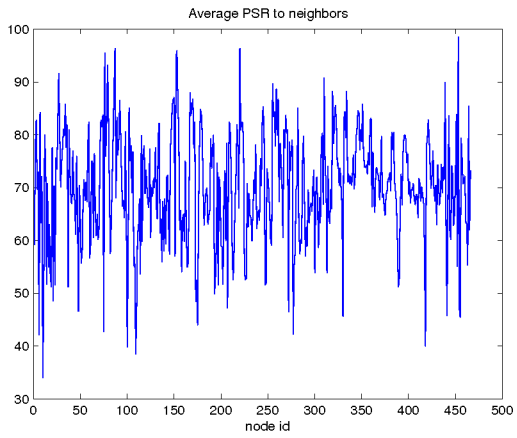


Fig. 3. Average PSR from each node to its neighbors. The average network PSR is 67%.

In addition, Fig. 4 shows the connectivity of the network

and number of unidirectional links calculated from this test. Since the average PSR obtained is only 67%, we consider this an unreliable network suitable to evaluate the performance of the proposed mechanisms. The following section introduces the results of such evaluation.

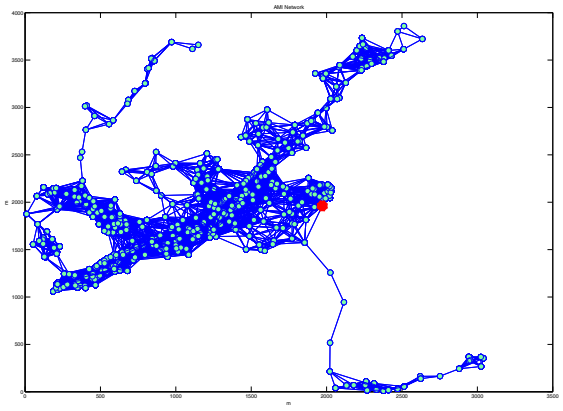


Fig. 4. The simulated network is based on coordinates obtained from an ongoing deployment. It consists of 466 nodes and one collector (in red). There are 26223 links with PSR greater than 0 in the network. Of these, 1815 links are unidirectional and the remaining 12204 are bidirectional. The largest minimum number of hops between two nodes in the graph is 14, and the largest minimum number of hops to the collector is 8. In our simulations, the routing protocol selects routes based on the ETX metric, not hop count; therefore the number of hops in practice could be higher.

IV. Simulation Results

This section discusses the performance of the data forwarding mechanisms over an unreliable network such as the one described in section III. The functionality of each mechanism is implemented as an upper layer that works on top of MiXiM's 802.15.4 MAC and PHY modules for wireless sensor networks. We include a distance-vector control data plane that fills up the routing table with up to K=3 candidates per destination. Since the network is unreliable, the control data plane assumes that all links are unidirectional unless proven otherwise (i.e, each node checks if it appears in the control messages reported by its neighbors). The metric used to calculate the best paths is based on the expected transmission count metric (ETX) [11].

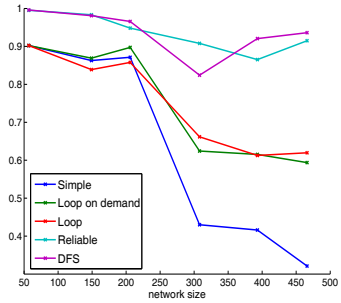
We trigger the sending of data packets at a rate of 1pkt/10min to emulate typical traffic conditions in a AMI network. The total data packet size is 103bytes ($payload + header$). We ran tests for two different scenarios to capture the performance of the data forwarding mechanisms for different parameters of configuration.

First, test 1 simulates a typical AMI network in which meters send packets to the collector at a constant bit rate. The purpose of this test is to evaluate how well the forwarding mechanisms overcome flaw routes and link layer failures, in order to guarantee delivery of packets. Second, we test the performance of the *reliable delivery* mechanism for different K values, under the same data traffic conditions described for test 1. The purpose of test 2 is to evaluate the impact of the number of candidates in the performance of the forwarding mechanism. Results from test 2 could help on the parametrization of the control plane in terms of the next-hop list size to be included in the routing table.

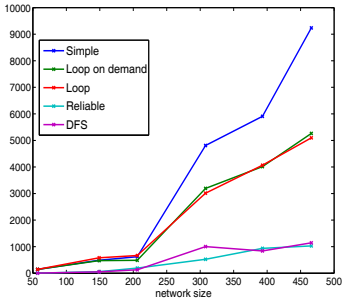
A. Test 1: Traffic to the collector

In this scenario, we compare the forwarding mechanisms in terms of the following metrics:

- Reliability: Measurement of the packet delivery ratio, calculated as $PDR = \frac{PktsReceived}{PktsSent}$
- Average delay: Measurement of the average end-to-end delay of packets received at the destination
- Hop count: Average number of hops traversed by packets from each source to destination
- Black holes: Number of times the forwarding mechanism discards a packet after the MAC layer fails to deliver it to the next hop



(a) Reliability

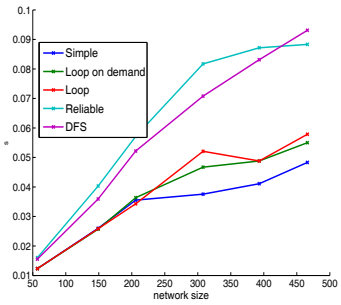


(b) Black holes

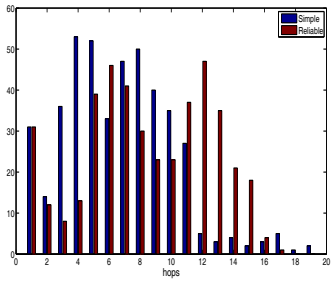
Fig. 5. Reliability and black holes evaluation of data forwarding mechanisms in unreliable networks

The reliability obtained by each mechanism is depicted in Fig. 5(a). One can observe that a simple forwarding mechanism does not achieve a PDR greater than 50% for large size networks. The main packet losses in this scenario come from the unreliable conditions in the network channel. For example, in a 300-node network, 98% or the total packets dropped are discarded due to MAC-layer failures. The remaining 2% corresponds to packets dropped due to flaws in the paths calculated by the control plane. In general, reliable delivery improves the total reliability between 8% $\sim$ 48% for the different network sizes when compared to simple forwarding, and 9% $\sim$ 25% when compared to loop detection. The difference between the loop detection's PDR and reliable delivery's PDR indicates the packets that were saved from being dropped due to failures at

the MAC layer. An absolute number of packets discarded due to link layer failures is shown in Fig. 5(b).



(a) Average delay



(b) Hop count

Fig. 6. Average delay and hop count evaluation of data forwarding mechanisms in unreliable networks

Fig. 6(a) demonstrates that the more sophisticated the forwarding mechanism, the greater the delay for delivering a packet at the destination. A steep increase is observed specially for the two strategies performing retransmissions due to link-layer failures (Reliable delivery and DFS), which measures the search done by the forwarding mechanism to successfully find a route towards the destination of the "saved" packets–whereas the delay of other forwarding mechanisms does not account packets who could not find their destination. In the same way, an increase is observed in the average number of hops due to the rerouting process. For example, in the 466-node topology, an increase from 6.7 hops in average for simple forwarding to 8.5 for reliable delivery is observed. Fig. 6(b) illustrates the hops histograms for the 466-node topology, and compares the number of hops employed by the simple and the reliable forwarding mechanisms.

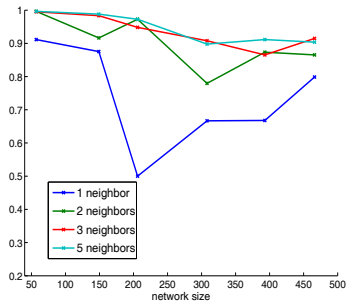
It is observed that, in general, reliable delivery and DFS have a comparable performance, with a slight improvement in PDR (Fig. 5(a)) and a slightly less average delay in the DFS mechanism (Fig. 6(a)).

A tradeoff expected from the good performance of reliable delivery and DFS is the memory employed to store entries in the loop detection table. Several strategies could be used to keep the loop detection table at a reasonable size: 1) by implementing a proactive deletion of entries, so that a timeout is set per entry. This timeout should be adjusted according to the network's data rate and the reliability conditions of the AMI network; 2) by implementing a reactive deletion of entries, so that a new packet registration triggers the deletion of the oldest entry in the loop detection table when the table is full; and 3) by implementing a reactive deletion for the entry that has been registered for a time longer than a specific threshold, and for which loops have not been yet detected. The threshold value should be adjusted according to well-known metrics such as the average end-to-end delay for data delivery in the AMI network.

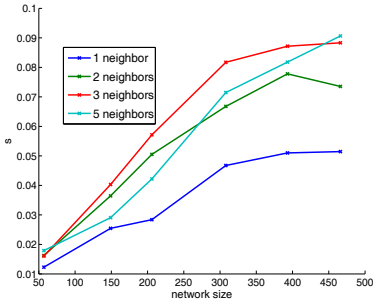
On the other hand, the maximum size of the table should vary according to memory resources. For example, some of our field deployments use a 3000-entry table in 802.11b devices, whereas only a 384-entry table is implemented in a memory-constrained 802.15.4 device.

B. Test 2: Impact of variable K

In this scenario, we run tests using reliable delivery as the forwarding mechanism. We compare the performance with $K = \{1, 2, 3, 5\}$, in which case the maximum number of candidates per destinations varies accordingly in the routing table. Besides comparing the reliability and average delay for each scenario, we also evaluate the total packets losses due to routing and MAC failures, and the number of times a node has to reroute a packet due to link layer failures.



(a) Reliability

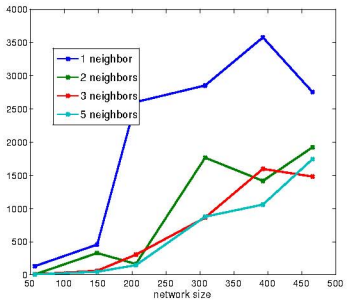


(b) Average delay

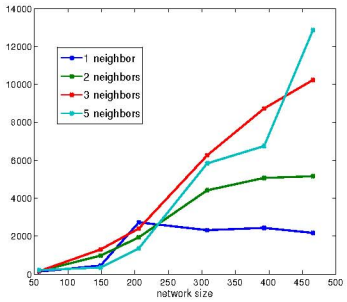
Fig. 7. Reliability and average delay evaluation of reliable delivery data forwarding for different values of K

Fig. 7(a) confirms that a greater number of options for forwarding a data packet impacts positively the chances for that data packet to be delivered. In a single candidate scenario ($K = 1$), the reliability decreases considerably especially for larger size networks, where the total PDR obtained is only 67%.

Accordingly, the average end-to-end delay tends to decrease for greater values of K, as demonstrated in Fig. 7(b). However, it is interesting to note that for $K > 1$ the delay experienced by data packets is comparable, which means that, although more



(a) Link-layer and Routing Failures



(b) Reroutes after link failures

Fig. 8. Routing failures and rerouting attempts evaluation of reliable delivery data forwarding for different values of K

data packets are delivered with the increase of K, those that are delivered do not necessarily reach the destination faster. One should also note that the reason for $K = 1$ to show the lowest delay is due to the fact that the successfully delivered packets have traveled through a high quality path in which a single trial was enough to deliver the packet. Since the routing metric optimizes the ETX, it is expected for those packets to have a very low delay (they are delivered at the first trial). However, for the same value of K, the packets dropped by the forwarding mechanism is considerably high. For example, in a 200-node network, the low delay in $K = 1$ comes at a cost of having near 5 times more packets discarded by the forwarding mechanism for link-layer and routing failures, compared to the number of packets discarded in $K > 1$. This behavior is illustrated in Fig. 8(a).

The number of packets that are retransmitted due to link failures are shown in Fig. 8(b). Although the generation of duplicate packets tend to increase regardless of the value for K employed in the routing table, the trend followed by $K = 1$ shows how the unreliability of the wireless channel causes this experiment to considerably increase the load in the network when the network size also increases, without experiencing a comparable increase in the delivery of packets (Fig. 7(a)). In other words, every time there is a link-layer failure, if $K = 1$ the forwarding mechanism creates a duplicate packet that is always returned to the previous hop, so only few packets move forward to reach the destination.

V. Field Experience

Our field measurements have confirmed that there is no clear definition of working and non-working wireless links in large AMI networks. In general, wireless links are very unpredictable and may experience interference, noise, and random signal attenuations like temporal shadowing and fading. Fig. 9 shows a typical RSSI measurement of our field trials, which is correlated with the highly unreliable links AMI networks have to experience.

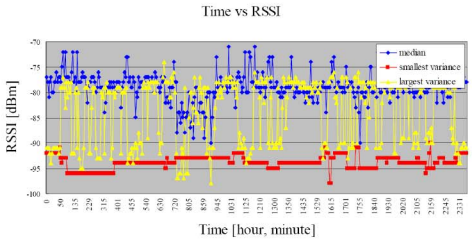


Fig. 9. A typical RSSI measurement in the field. In general the packet error rate at the link-layer is a sigmoid function of the RSSI. High levels of RSSI variability indicate highly unreliable links.

Link failures can lead to loss of control messages, which in consequence generate routing loops. In addition, link failures cause the reliability of a routing protocol to drop if the protocol cannot find an alternative way to forward packets. To address these problems, we have implemented two of the forwarding mechanisms described in section II in the field, in order to improve the reliability of routing protocols for large wireless mesh networks.

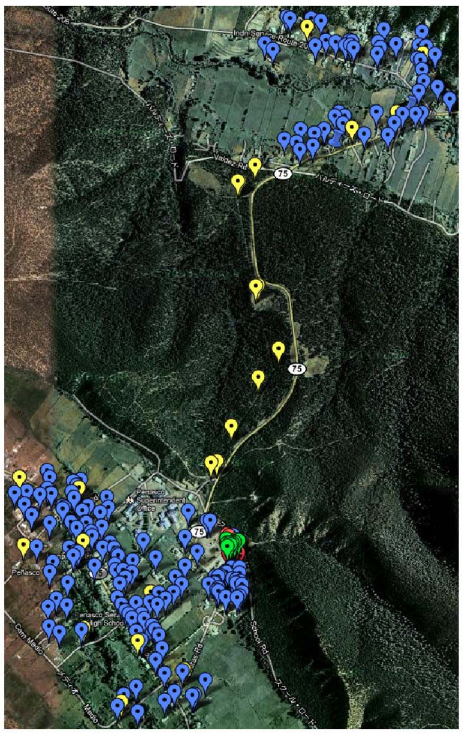


Fig. 10. Field trial in an area of 1.5 x 2 miles. This field trial has two gateways (for fault-tolerance) at the substation (in red), 20 routers and 176 meters.

Most of our experience pertains to deploying AMI networks of hundreds to thousands of nodes per collector in urban and rural environments. The majority of the large AMI deployments are located in Japan, but the data of these networks is

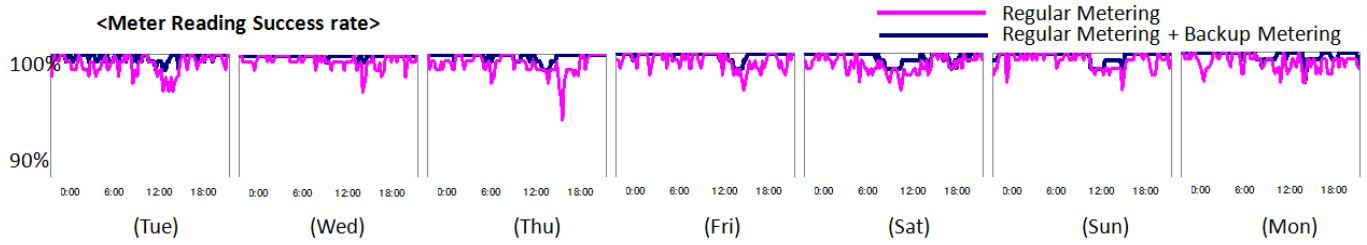


Fig. 11. Reliability for smart meter readings over a week-long period. During this period, 176 meters reported every 15 minutes their readings. The collector can poll a meter reader it has not received a reading from, if the first attempt fails.

property of the Japanese utilities and not easily shareable. On the other hand, one of our deployments outside of Japan can be seen in Fig.10. In this field test, we implemented loop on-demand and reliable delivery as the options that the routing protocol can select when forwarding packets. The forwarding option will depend on the priority of the packet that needs to be delivered or on the size of the loop detection table. For example, a regular meter reading sent to the collector is forwarded with loop on-demand forwarding, whereas a packet sent from the collector to a meter, or the response of the meter to a request from the collector, are forwarded with the reliable delivery option. Fig.11 shows the reliability results for the field deployment with a PDR in the range of 94% to 100%.

VI. Conclusions

We have introduced five different data forwarding mechanisms for the delivery of packets in wireless mesh networks employed for smart grid deployments under unreliable conditions. These mechanisms serve as a data forwarding plane that could be employed with any generic control plane/routing protocol in charge of filling a routing table. The logic employed to forward packets is based on a depth-first search over the network-graph that may use multiple next-hops in order to reach a destination. They also recover packets that would be normally discarded due to losses an the MAC layer. Our simulations results showed a high performance for those mechanisms that perform rerouting and retransmission under link failures. It has been also shown how the unreliable conditions in the AMI network affect its overall performance, and how the proposed data forwarding mechanisms overcome the instability of links to achieve a high delivery of packets under these unreliable conditions. Finally, we have shown results from a successful AMI field deployment that implements two of the proposed mechanisms: loop-on-demand and reliable delivery.

References

[1] J. Tripathi, J. de Oliveira, and J. Vasseur, "Applicability study of rpl with local repair in smart grid substation networks," in *2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, oct. 2010, pp. 262 –267.

[2] C. Chauvenet, B. Tourancheau, D. Genon-Catalot, P.-E. Goudet, and M. Pouillot, "A communication stack over plc for multi physical layer ipv6 networking," in *2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, oct. 2010, pp. 250 –255.

[3] B. Lichtensteiger, B. Bjelajac, C. Müller, and C. Wietfeld, "RF Mesh Systems for Smart Metering: System Architecture and Performance," in *2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, oct. 2010, pp. 379 –384.

[4] T. Iwao, K. Yamada, M. Yura, Y. Nakaya, A. Cárdenas, S. Lee, and R. Masuoka, "Dynamic data forwarding in wireless mesh networks," in *2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, oct. 2010, pp. 385 –390.

[5] S. Dawson-Haggerty, A. Tavakoli, and D. Culler, "Hydro: A hybrid routing protocol for low-power and lossy networks," in *2010 First IEEE International Conference on Smart Grid Communications (SmartGridComm)*, oct. 2010, pp. 268 –273.

[6] P. Thubert *et al.*, "RPL: IPv6 Routing Protocol for Low power and Lossy Networks," IETF Secretariat, Internet-Draft draft-ietf-roll-rpl-19.txt, March 2011.

[7] J. Chroboczek, "The Babel Routing Protocol," IETF Secretariat, RFC 6126, April 2011.

[8] I. Cidon, "Yet another Distributed Depth-First-Search Algorithm," *Information Processing Letters*, vol. 26, no. 6, pp. 301 – 305, 1988.

[9] B. Awerbuch, "A new Distributed Depth-First-Search Algorithm," *Information Processing Letters*, vol. 20, no. 3, pp. 147 – 150, 1985.

[10] A. Cárdenas, S. Céspedes, and T. Iwao, "Depth-First Forwarding in Unreliable Networks," IETF Secretariat, Internet-Draft draft-cardenas-dff-00.txt, July 2011.

[11] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A High-Throughput Path Metric for Multi-hop Wireless Routing," in *Proceedings of the 9th annual international conference on Mobile computing and networking*, ser. MobiCom '03, 2003, pp. 134–146.