Peer-to-peer Detection of DoS Attacks on City-Scale IoT Mesh Networks

Michael J. Rausch[1], Varun Badrinath Krishna[1], Peng Gu[1], Rupak Chandra[2], Brett Feddersen[1], Ahmed Fawaz[1], and William H. Sanders[1]
[1]Information Trust Institute, University of Illinois at Urbana-Champaign, 1308 W. Main St, Urbana, IL 61801
[2]Cisco Systems Inc. 560 McCarthy Blvd, Milpitas, CA 95035
E-mail: {mjrausc2,varunbk,penggu2,bfeddrsn,afawaz2,whs}@illinois.edu, ruchandr@cisco.com

***Abstract*—Wireless IoT mesh networks are being widely deployed for use in applications such as operational technology networks in power grids, city-scale surveillance, and monitoring. The benefits of such networks, which may include mission-critical communications, can be undermined by an adversary who launches denial-of-service (DoS) attacks on them. In this paper, we present a peer-to-peer approach to detecting and localizing such adversaries by leveraging the topology of the mesh network. In doing so, we make three main contributions. First, we present insights from a preliminary implementation on a standards-based IoT platform used in real smart meter deployments. Second, we propose an optimal choice of peers that can help detect a jammed node, while minimizing the risk that the peers themselves are jammed. Finally, we present a tool to help generate datasets of city-scale IoT mesh topologies for simulation studies.**

I. Introduction

The Internet of Things (IoT) revolution is transforming many industries, with improved connectivity driving profits [1]. Wireless IoT devices are widely used in mission-critical applications, such as operational technology (OT) networks in power grids, oil & gas infrastructure, water delivery systems, and city-scale surveillance systems. Those devices often form mesh networks and exchange information to enable better monitoring and control. The benefits of such networks can be undermined by an adversary who launches denial-of-service (DoS) attacks on them. In this paper, we limit our focus to jamming attacks and flooding attacks on wireless IoT devices, both of which interfere with the connectivity between those devices. We present our preliminary work on detecting and localizing such adversaries by leveraging the topology of the mesh network. Although our work is motivated by city-scale IoT applications, the methods are applicable to smaller-scale applications (e.g., IoT networks inside buildings).

Our solution was first developed on Cisco's Connected Grid Mesh Network (CG-Mesh) platform [2]. CG-Mesh is a standards-based IoT suite designed primarily for *field area networks*, such as the advanced metering infrastructure (AMI) and distribution automation applications in power grid OT networks. AMI is a city-scale (or even larger) infrastructure that comprises a mesh network of wireless smart meters that measure and report electricity consumption readings to electric utilities that typically own and operate them.

CG-Mesh implements the full IoT communications stack. At the MAC layer, the wireless nodes use the IEEE 802.15.4 standard for low-powered and lossy networks (LNNs). At the network layer, they use the 6LoWPAN standard, which is a version of IPv6 for LLNs. Routing is accomplished using the routing protocol for LLNs (RPL) [3]. As these standards are open, our solution is broadly applicable, and does not include any proprietary components.

Implementations of CG-Mesh involve multiple mesh networks, each connected to an RPL border router. Through RPL, a destination-oriented directed acyclic graph (DODAG) is formed for each mesh network and router. The nodes, which are meters in the case of AMI, send measurements to the RPL border router through the best route, as determined by RPL. We assume that nodes are wireless and susceptible to wireless DoS attacks, which may include beacon frame flooding and signal jamming. We assume that the RPL border routers are not susceptible to those attacks because they use WAN technologies (3G/4G/LTE/WiMAX) to communicate upstream. They communicate with the field network director (FND) at the utility's data center, which manages the entire IoT network for the application.

The proposed approach comprises two steps: detection and localization. During detection, we determine whether nodes are affected by the DoS attack, using other nodes (peers) in the network to observe any drop in link quality. During localization, we create a map of all unreachable nodes and use a priori information about the nodes' physical locations to estimate the location of the attacker.

We make three main contributions. First, we present insights from a preliminary implementation of the detection approach on a standards-based IoT platform used in real smart meter deployments. Second, we propose an optimal choice of peer nodes that can help detect a jammed node, while minimizing the risk that the peers themselves have been jammed. Finally, we present a tool to help generate datasets of city-scale IoT mesh topologies for simulation studies.

This paper is organized as follows. First, in Section II, we discuss limitations of approaches in related work. Next, in Section III, we discuss an active approach to detect jamming. The approach uses peers that observe link quality metrics. We then present, in Section IV, a tool to help generate datasets of city-scale IoT mesh topologies for simulation studies. Then, in Section V, we propose a strategy to optimally identify peers to minimize the risk that the peers themselves are

jammed. We use a real city-scale layout obtained from our tool to evaluate that placement strategy. Finally, we evaluate existing localization approaches on that real physical layout in Section VI. We conclude in Section VII.

II. Related Work

Many solutions have been proposed for detecting jammers, as surveyed in [4]. In our work we assume that the attacker can jam all the channels in the wireless spectrum of operation. Therefore, approaches like [5] that use channel-hopping to provide resilience against jammers would be ineffective. The authors of [6] consider attacks wherein the jammer creates collisions and forces retransmissions, while considering the probability of collisions. In our model, we assume that the attacker has the ability to incessantly transmit beacon frames, so the probability of a collision is 1. Further, the receive nodes are flooded as a consequence of the attack.

The authors of [7] also discuss detection of jamming attacks in the context of power grid devices. Their detection approach is also proactive, like ours, but they do not discuss the risk that the probing node will itself be jammed. The discussion of that risk is a main contribution of our work.

Further, most approaches in prior work were designed for uniformly distributed mesh nodes in a spatial region [8], [9]. As a result, lack of uniformity, which is common in real-world deployments, degrades the localization accuracy. In smart meter deployments, for example, as we show in Section IV, the locations of the meters are not random and follow the careful city planning of streets and neighborhoods.

We use the minimum enclosing circle approach in identifying the best observer nodes that can be chosen for a given observed node in our active observation detection scheme. The approach was used for localization in [10] and [11]. Other approaches for localization were discussed in [12], [13], [14].

To the best of our knowledge, none of the prior work was based on industry communication standards. We use such standards to ensure that our approaches can be adopted by researchers and industry.

III. Detection of Affected Nodes

We explain two ways in which wireless DoS attacks can be detected. Our contribution is in detailing an active approach, but we also describe a passive approach. In both approaches, the FND aggregates information from multiple nodes and determines whether the failure is isolated or is indicative of a jamming attack. That role can also be played by the RPL border router, in which case the intelligence is pushed to the router. That strategy is referred to as *fog computing*.

A. Passive Monitoring

In passive monitoring, the FND waits for nodes to time out before deciding whether they have been affected. If a sufficiently large number of nearby nodes time out simultaneously, it is indicative of a DoS attack. The manner in which that sufficient number is determined would vary depending on the specific application. For example, if there is an outdoor monitoring application wherein IoT nodes are solar-powered and designed not to send messages to the router at night, lack of communication during the night would not indicate a DoS attack.

If the nodes are expected to report telemetry data periodically, then the timeout could be defined by a certain number of periods. If the nodes are not expected to report data, but are programmed to send heartbeats to the head-end router to confirm that they are active and reachable, then that heartbeat period can be used to design a timeout value. In the case of LLNs, packet loss is common, and thus the timeout value would need to account for the expected packet drop rate.

The advantage of that approach is that no additional code needs to be installed on the IoT nodes to perform DoS attack detection. The disadvantage is that if the timeout is set at a high value, then it would take longer for the FND to determine that a DoS attack has happened. If the IoT application is not mission-critical, then the passive approach may strike a good balance between cost and benefit.

B. Active Approach

We propose an active approach in which we install code on each node to proactively perform link quality estimation on the neighboring nodes. A drop in link quality can be signaled to the FND, which may then decide whether the drop is a consequence of an attack or of the lossy channel. The advantage of this approach is that it can be used by the network administrator to investigate a potential DoS attack on demand. The disadvantage is that it involves installing code on each IoT node specifically for the purpose of detection.

The proactive detection approach is peer-to-peer (P2P), and the FND does not participate in it. The FND aggregates reports of poor link quality and uses that aggregate data to determine whether a DoS attack is happening.

1) Peer-to-Peer Detection: The P2P detection approach assigns each node two or more *observer nodes* that run code to perform link quality estimation over a period of time. Every node in the network is observed by an observer node, but not every node needs to be an observer node. If node A is an observer of node B, it collects a sequence of link quality estimates taken periodically. From that sequence, it uses statistical learning to identify dramatic changes in link quality that may indicate that the link is affected by an attack. We refer to node B in this context as the *observed node*.

It is required that every observer node be able to communicate with every node that it checks. The link layer contains the most comprehensive list of neighbors. In comparison, the network layer neighbors (determined by the Neighbor Discovery Protocol of 6LowPAN) are restricted to the candidate parents in the DODAG, as determined by RPL.

2) Assignment of Observer Nodes: Any of an observed node's one-hop neighbors may serve as an observer node. In some circumstances, it may be preferred to have every node in direct communication with an observed node serve as an observer node, to provide redundancy. However, it may not be feasible to have every eligible node serve as an observer node.

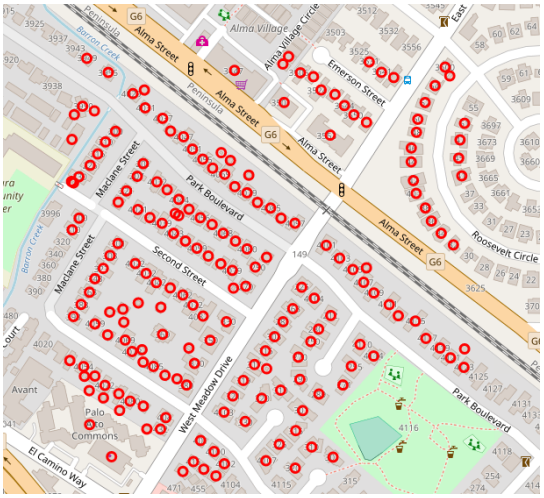

Fig. 1: Sample output from a city-scale layout generator for a neighborhood in Mountain View, California.

For example, energy, bandwidth, and computation constraints may limit the number of observer nodes that can be assigned to each observed node. The challenge then is to select some subset of all eligible nodes to serve as observers in a way that minimizes the chance that an adversary would be able to jam the observed node and all of the observer nodes. We compare two approaches to observer node selection in Section V.

3) Link Quality Estimation: For its simplicity, we recommend the use of the Expected Transmission Count (ETX) to measure the link quality between two IoT nodes. ETX is the expected number of transmissions of a packet necessary for it to be received without error at its destination. Our approach does not depend on the use of this metric, but we found that it is easy to implement. As part of 802.15.4, ETX is measured and updated for every link neighbor, every time a message is acknowledged. In our RPL implementation, ETX is used as the link quality metric to determine the least-cost path to the RPL border router (root of the DODAG). No additional code is necessary to calculate the ETX, but we extended the MAC layer to allow historic ETX values to be stored in each observer node for all the observed nodes. For example, if an observer node were assigned ten observed nodes, and the ETX were stored for the past 100 acknowledgments, then the total memory requirement on each observer node would be 1000 integers. Setting a cap on the number of observed nodes for a given observer node ensures that the memory requirement is capped according to the hardware constraints of the node. As this approach actively monitors link quality, we needed a way to ensure that the ETX value was up to date. That is not automatically ensured in the IoT stack because ETX is asynchronously updated only when there is a need to send a message to a particular neighbor. Also, the ICMPv6 echo (ping) messages could not be leveraged, because the 6LowPAN layer does not have access to all the link neighbors. (As mentioned before, the 6LowPAN layer only keeps track of candidate parents, as determined by RPL.) Therefore, we implemented a link layer echo command using the reserved header for proprietary commands in IEEE 802.15.4e. We intend to propose amendments to the standard through the Wi-SUN Alliance to add a header option for link layer management. Especially in the case of IoT, in which the network layer does not see all neighbors, we believe that there are several benefits to having link layer query functionality. In addition to implementing the link layer echo command, we implemented an empty acknowledgment of the message, and that automatically triggered the recalculation of the ETX value on the sender (the observer node). The new ETX value was then added to the list of historical ETX values for that observed node. The periodicity with which the observer node sends link layer echo commands can be tuned based on the requirements of the application. Note that the echo and its acknowledgment can be restricted to neighbors that are authenticated using 802.1x. That prevents the echo from being used like a beacon for flooding attacks.

4) Statistical Anomaly Detection: We use the historic ETX values to determine whether the latest ETX value is anomalous. We propose two alternative models for this detection. The first model assumes the ETX values follow a Gaussian distribution, so the mean (μ) and the standard deviation (σ) of the historical values are calculated, and a threshold for the new ETX value is set at $\mu + k\sigma$. The second model assumes the ETX values follow a Laplacian distribution, and the corresponding maximum likelihood estimates are calculated as they were for the Gaussian distribution. Specifically, the median (ν) and the mean absolute deviation (γ) are calculated from historical values, and the threshold for the new ETX value is set at $\nu + k\gamma$. k is a configurable parameter, and a value between 3 and 6 is typically chosen.

An alternative approach was suggested by the authors of [15], who also use a derivative of ETX as a feature to train a machine learning classifier (random forests) to detect a jamming attack. Random forests have a large memory and CPU requirement, and in our AMI deployments, the hardware cannot support such an algorithm.

IV. City-Scale IoT Layout Generator

We present a tool that we created for this study that will be released open-source with this paper. The tool uses the Overpass API from OpenStreetMap to create a map of every house in a city. The input to the tool is the name of the city, and the output is a list of latitude/longitude coordinates of all the buildings in the city, many of which are named.

We used the tool to create real layouts of smart meters in cities for testing attacker localization algorithms. A sample of the output of our tool for the city of Mountain View, California is illustrated by the circular markers in Fig. 1. As illustrated, the real data reveal that the distribution of meters in a 2D geographical space is not uniformly random. Thus, many centroid-based localization approaches, which assume uniformly random IoT node locations, may not work well in the case of city-scale applications in which each node is in a different building. While the creation of smart meter maps is the most obvious use case for our tool, we believe that the

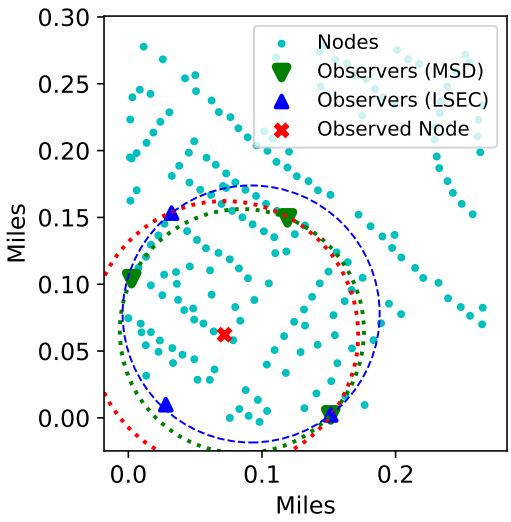

Fig. 2: Illustration of different choices of observer nodes by different methods for the same observed node. The layout is that of smart meters in a neighborhood of Mountain View.

members of the community may have other reasons to generate city-scale IoT node layouts to aid their research.

V. Optimal Observer Placement

Observers should be chosen to minimize the risk that an attack will jam all of the observer nodes along with the node that they are observing. If the observed nodes are jammed, it may still be possible to detect jamming so long as the observers in turn have observers that are not jammed. We shall first describe one intuitive, but naive, approach for assigning observer nodes, for a comparison baseline. Then, we will explain our approach, which is optimal and minimizes the risk that the observers are jammed. Its performance is greater than or equal to that of the naive approach.

In our threat model, the attacker possesses an omnidirectional jammer that has a fixed range. He or she can disrupt communication to and from any node within that range, and is free to place the jammer anywhere in the IoT network. Given a set of observer nodes, A, and another set of observer nodes, B, we say that A is a *superior* selection of observers relative to B iff an adversary that conforms to our threat model incurs a greater cost in jamming all the nodes in A than in B, given that operating a jammer with a larger radius costs an adversary more than operating a jammer with a smaller radius. Specifically, if a jammer with a fixed jamming radius were able to jam all the observers in B, but unable to jam all the observers in A with the same jammer by moving to a different location (or by staying in the same place), then A is a superior choice. We assume that an adversary will always be detected if he or she fails to jam both the observed node and all observers.

A. Maximizing Sum of Distances Approach

The first approach, the *Maximizing Sum of Distances* (MSD) approach, is illustrated in Fig. 2, wherein three observer nodes have been assigned to the observed node. The assignment is done in a manner that maximizes the sum of the distances between the three observer nodes. First, the Euclidean distance between every pair of nodes is calculated in $O(N^2)$ time, where N is the number of all nodes. Then, for each observed node, every combination of four distances (between the three candidate observer nodes and one observed node) is evaluated to maximize the sum of the distances. If the nodes have at most $M < N$ neighbors within range, and there are k observer nodes, the total cost in terms of complexity is $O(\binom{M}{k})$. Although the complexity is high, the observer selection algorithm can be run on the FND, which has sufficient computational resources. The algorithm only needs to be run once, unless more nodes join or leave. Also, M is usually small. Although the approach works well in most scenarios, it does not maximize the attacker's cost.

B. Largest Smallest Enclosing Circle Approach

The second approach, the *Largest Smallest Enclosing Circle* (LSEC) approach, is illustrated in Fig. 2, where three observer nodes have been assigned to the observed node. The assignment is done such that it maximizes the smallest circle that encloses all three observer nodes, in addition to the observed node. The smallest enclosing circle given a set of points can be obtained in linear time [16]. Again, let N be the total number of nodes; let M be the number of neighboring nodes that are within wireless range of the observed node; and let k be the number of observers required. For each observed node, every combination of k link neighbors is evaluated by calculating the smallest circle that encloses those neighbors and the observed node itself. The combination of nodes that makes the largest circle is selected to form the observer group. The total complexity is $O(\binom{M}{k})$. As discussed previously, the relatively high algorithmic complexity should not be problematic if the algorithm is run on the FND.

C. Comparing the Two Approaches

We evaluated the two approaches on a real physical layout of 197 nodes from a neighborhood in Mountain View, California obtained from our city-scale layout generation tool. The choices of observers made by the two approaches, LSEC and MSD, are illustrated in Fig. 2. The red dotted circle in the figure indicates the wireless range of the observed node. We used both approaches to select observer nodes for each node in the neighborhood. Each approach chose three observers. In 79 of the 197 cases ($\sim$40%), both approaches selected the same observers, and thus provided equivalent defensive coverage.

In each of the 118 other cases, LSEC selected better observer nodes than did MSD. In other words, for every one of those 118 nodes, there exists a jammer with a fixed radius that could disable the observed node and all the observer nodes selected by MSD, but would be unable to disable the observed node and all of the observer nodes selected by LSEC.

An example of the phenomenon is illustrated in Fig. 2. The red "x" denotes the node to be observed; the upside-down green triangles represent the observer nodes selected

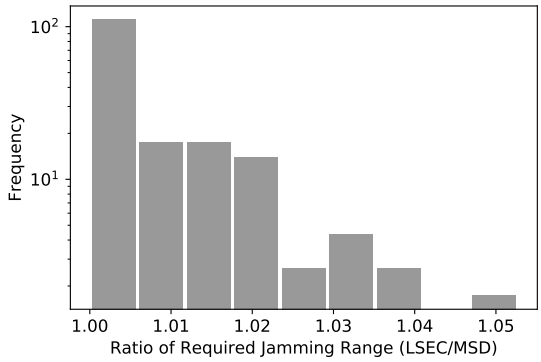

Fig. 3: A histogram showing the relative benefit of using LSEC vs. MSD, given by taking the ratio of the smallest jamming radius required to block all observer nodes selected by LSEC, and the smallest jamming radius required to block all the observer nodes selected by MSD.

by MSD; and the blue triangles represent the observer nodes selected by LSEC. Note that the observer node farthest to the south-east (bottom right) was selected to be an observer node by both MSD and LSEC. The smaller dotted green circle connecting the MSD observer nodes represents the coverage of the jammer with the smallest radius that could still disable the observed node and all of the observer nodes. The larger dashed blue circle connecting the LSEC observer nodes represents the coverage of the jammer with the smallest radius that could still disable the observed node and all of the LSEC observer nodes. It is clear from the illustration that a jammer that could disable all of the observer nodes chosen by MSD may not necessarily be able to disable all of the observer nodes selected by LSEC. We plot the relative strength of LSEC vs. MSD in Fig. 3.

In addition, the MSD approach suffers from an interesting weakness compared to the LSEC approach. Adding more observer nodes will sometimes result in poorer coverage with the MSD approach, which is not a property of the LSEC approach. This is illustrated in Fig. 4. Two observer nodes chosen by MSD don't provide as much DoS protection as one observer node chosen by LSEC. If the MSD approach had chosen one observer, it would have picked the same observer as LSEC, but since it picked two observers, it chose observers that provided poorer DoS protection. The LSEC approach will never choose an observer that decreases its ability to defend against a DoS attack.

D. Proof of Optimality of LSEC

We have shown that the LSEC approach is superior to the MSD approach in certain cases, in both theory and practice. What follows is a proof of the optimality of the LSEC approach. Consider an observed node, l. Let M denote its set of m neighbors. We wish to identify a set of k neighbors, denoted by K, where $K \subseteq M$. Assume that all the nodes lie in a 2D plane.

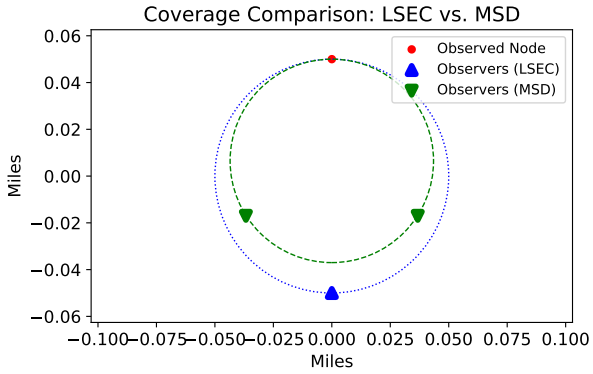

Fig. 4: A comparison showing that in certain cases, one observer node selected by LSEC provides better DoS protection than two observer nodes selected by MSD.

Definition V.1. The *DoS-resistance radius* of the set $\{K, l\}$ is the radius of the smallest circle that would contain every node in the set (the minimum enclosing circle). An adversary would need an omnidirectional jammer with a radius at least as large as the DoS-resistance radius to successfully jam the observed node l and all observers in K.

Definition V.2. A *k-optimal DoS-resistant observation set*, denoted by K^*, is a choice of K that maximizes the attacker's cost in jamming all the nodes in $\{K, l\}$. Specifically, there exists no other set containing l and k observer nodes with a greater DoS-resistance radius than $\{K^*, l\}$.

Theorem 1. *The Largest Smallest Enclosing Circle approach always chooses a k-optimal DoS-resistant observation set.*

Proof. The LSEC approach first calculates the radius of the smallest enclosing circle (the DoS-resistance radius) for every possible set $\{K, l\}$, and then selects a set $\{K^*, l\}$ whose DoS-resistance radius is largest. Such a set is, by definition, a k-optimal DoS-resistant observation set. Hence, the theorem is proven by construction of the LSEC approach. □

E. Communicating: Assignments and Abnormalities

The calculation that determines which nodes should be observer nodes for each observed node is done at the FND, and the assignment is communicated to the observer nodes through a management protocol that runs on top of 6LowPAN. In our implementation, the Constrained Application Protocol (CoAP) was used over UDP, and that is typical for IoT applications. The observer node receives a list of observed nodes from the FND and acknowledges to confirm whether or not the observed nodes are in its list of link neighbors. If they are not, the FND finds a different observer node for that observed node.

If all observer nodes of an observed node were to be affected by a jamming attack, then FND would need to discover the affected nodes by leveraging the transitive relationship of the observer nodes. For example, if nodes A and B are observing node C, and the three nodes are affected by an attack, then the FND would find out through the observer nodes for nodes A and B. If every node in the network were affected, then

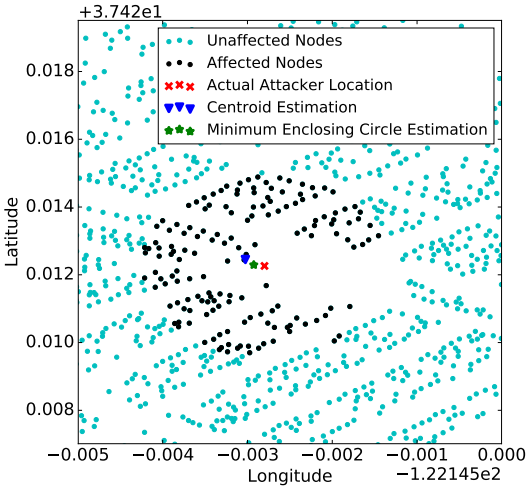

Fig. 5: Jammer localization illustrated on a real physical layout of smart meters in Mountain View, California.

the transitivity would not help. To address that issue, the RPL border router could play the role of an observer node for all of its immediate children in the DODAG. As we assumed the RPL border router uses WAN technologies and is immune to DoS attacks in the 802.15.4 wireless spectrum, it would be able to inform the FND of the situation.

Once the observer node has determined that its link quality with an observed node is abnormally poor, it will try to report the anomaly to the FND through the DODAG. If the observed node was the DODAG parent of the observer node, then RPL would automatically find a different parent for the observer node that would allow end-to-end connectivity with the FND.

VI. Localization

In this section we illustrate two approaches from related work for attacker localization on our smart meter location dataset sample from Mountain View. The first approach calculates the centroid of the locations of the affected nodes, and estimates that to be the attacker's location. The second approach calculates the minimum enclosing circle of all affected nodes, and estimates the attacker's location to be at the center of the circle. The approaches are illustrated in Fig. 5.

We evaluated the accuracy of both localization approaches by placing the attacker at 100 different randomly chosen points on the Mountain View map and calculating the error achieved by each approach in each scenario. The errors were normalized by the jammer radius and expressed as a percentage. On average, the minimum enclosing circle approach produced an error 3.75 percentage points below the error in the centroid-based approach.

VII. Conclusion

In this paper we present a P2P approach to detecting DoS attacks due to jamming in wireless IoT mesh networks. We presented a standards-based approach for detection and evaluated localization approaches by using smart meter layouts obtained from a custom tool that reads data from OpenStreetMap. Our detection approach relies on observers that determine whether a node is jammed based on their observations of link quality metrics. We proposed an optimal strategy to choose observers in a manner that minimizes the risk that the observers will themselves be jammed.

Acknowledgment

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000780.[1] The authors would like to thank Jenny Applequist for her editorial assistance.

References

[1] C. Thompson, "Here's how iot is transforming 6 different industries," *Business Insider*, October 2016, http://www.businessinsider.com/iot-transforms-industries-2016-10/.

[2] *Release Notes for CG-Mesh Release 5.6.20*, Cisco Systems Inc., https://www.cisco.com/c/en/us/td/docs/routers/connectedgrid/modules/release_notes/cgmesh_rn_5_6.html Accessed November 1, 2017.

[3] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. P. Vasseur, and R. Alexander, *RPL: IPv6 routing protocol for low-power and lossy networks*, RFC6550, March 2012, https://tools.ietf.org/html/rfc6550.

[4] X. Wei, Q. Wang, T. Wang, and J. Fan, "Jammer localization in multi-hop wireless network: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 19, no. 2, pp. 765–799, 2017.

[5] V. Navda, A. Bohra, S. Ganguly, and D. Rubenstein, "Using channel hopping to increase 802.11 resilience to jamming attacks," in *Proc. IEEE INFOCOM 2007: 26th IEEE Int. Conf. Computer Communications*, May, pp. 2526–2530.

[6] M. Li, I. Koutsopoulos, and R. Poovendran, "Optimal jamming attacks and network defense policies in wireless sensor networks," in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, May 2007, pp. 1307–1315.

[7] Z. Lu, W. Wang, and C. Wang, "Modeling, evaluation and detection of jamming attacks in time-critical wireless applications," *IEEE Trans. Mobile Computing*, vol. 13, no. 8, pp. 1746–1759, Aug 2014.

[8] H. Liu, W. Xu, Y. Chen, and Z. Liu, "Localizing jammers in wireless networks," in *Proc. 2009 IEEE Int. Conf. Pervasive Computing and Communications*, pp. 1–6.

[9] Z. Liu, H. Liu, W. Xu, and Y. Chen, "Wireless jamming localization by exploiting nodes' hearing ranges," in *Proc. 6th IEEE Int. Conf. on Distributed Computing in Sensor Systems*. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 348–361.

[10] S. Wang and C. Chu, "Geometry-covering jammer localization based on distance comprehension in wireless sensor networks," *CoRR*, 2015. [Online]. Available: https://arxiv.org/abs/1512.06468

[11] T. Cheng, P. Li, and S. Zhu, "An algorithm for jammer localization in wireless sensor networks," in *Proc. 2012 IEEE 26th Int. Conf. Advanced Information Networking and Applications*, pp. 724–731.

[12] Y. Sun and X. Wang, "Jammer localization in wireless sensor networks," in *2009 5th Int. Conf. Wireless Communications, Networking and Mobile Computing*, pp. 1–4.

[13] Z. Liu, H. Liu, W. Xu, and Y. Chen, "An error-minimizing framework for localizing jammers in wireless networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 25, no. 2, pp. 508–517, Feb. 2014.

[14] T. Cheng, P. Li, and S. Zhu, "Multi-jammer localization in wireless sensor networks," in *Proc. 2011 7th Int. Conf. Computational Intelligence and Security*, pp. 736–740.

[15] O. Pual, I. Akta, C. J. Schnelke, G. Abidin, K. Wehrle, and J. Gross, "Machine learning-based jamming detection for IEEE 802.11: Design and experimental evaluation," in *Proc. IEEE Int. Symp. World of Wireless, Mobile and Multimedia Networks 2014*, pp. 1–10.

[16] E. Welzl, "Smallest enclosing disks (balls and ellipsoids)," in *New Results and New Trends in Computer Science*, H. Maurer, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 359–370.

[1]The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.