Analysis and Enhancement of RPL under Packet Drop Attacks

Binbin Chen, Yuan Li, and Daisuke Mashima
Advanced Digital Sciences Center
{binbin.chen, yuan.li, daisuke.m}@adsc.com.sg

***Abstract*—We study the performance of IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) under packet drop attacks. We consider an external jamming attacker who can selectively interfere with traffic around a targeted router. We show that the RPL implementation in Contiki OS allows such an attacker to continuously drop packets forwarded via the targeted router without triggering any rerouting, even when link-layer security mechanisms are in place. To counter such attacks, we design and analyze additional measures that can be built on RPL. In particular, we show that adding measures at the targeted router is more effective than doing so at affected children nodes. We also evaluate the performance of our enhanced RPL using Cooja, the Contiki network simulator. Our results show that our proposed measures have the potential to significantly reduce the fraction of packets being dropped without affecting RPL's performance when there is no attack.**

I. Introduction

Routing over low-power and lossy networks (LLN), e.g., an 802.15.4g/e-based mesh network of smart meters, sensors, or other IoT devices, is an important and challenging problem that has been extensively studied by both academia and industry. IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [1] has emerged as a major proposal to address this topic. RPL is designed to run on routers with limited processing power, memory, or energy supply. It deals with links that can have time-varying quality and high loss rates. Furthermore, RPL targets to support large networks that can scale to thousands of nodes.

While RPL presents carefully engineered solutions to tackle these LLN challenges, the increasing cyber security risk associated with such networks calls for a systematic re-examination of the RPL design from an adversary perspective. In particular, cyber attacks on such networks, e.g., attempts to abuse advanced metering infrastructures, have become a real threat. In this paper, we present our initial work to study RPL under packet drop attacks that can be mounted by sophisticated jammers, which could have detrimental impact on such networks.

Compared to other cyber attacks, packet drop attacks have several unique properties that deserve special treatments. 1) Unlike attacks on message confidentiality, integrity, or authenticity, a packet drop attack cannot be handled in a purely cryptographic way. Specifically, as we will show in Section V-A, an external attacker, without knowing any secret keys used in the network, can still launch packet drop attacks effectively by intelligently jamming packets sent by a targeted router.

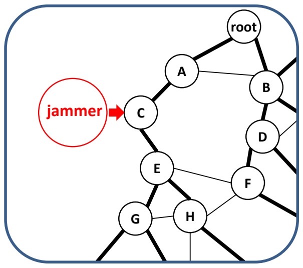

Fig. 1. An illustration of the packet drop attacks studied in this work. The jamming attacker can interfere with the traffic around a targeted router (node C in the figure). As a result, C and its descendant nodes (including E, G, H, and their children)'s traffic can be dropped. An edge between two nodes in the graph indicates that they can communicate directly. Edges selected by the routing protocol for packet forwarding are thickened.

2) The attack can have a global instead of local impact, due to the packet forwarding nature of a mesh network. In other words, by disabling a single router, the attacker may affect a significant fraction of traffic in a large-scale network that is routed through the jammed region. 3) It could be difficult to distinguish a packet drop attack from link instability situation that happens naturally in LLN. Hence, measures to deal with packet drop attacks need to ensure they don't affect RPL's capability to deal with unstable links.

Our work considers an external packet drop attacker, who cannot change the internal logic of legitimate routers in the network, or impersonate them. Such assumptions can be achieved, e.g., in a cryptographic way, by authenticating and encrypting all network traffic, including MAC-layer acknowledgment (ACK) frames. In particular, RFC 8180 [2] specifies that such link-layer security mechanisms must be used with RPL. To simplify our discussion, we consider a single static jammer that interferes with the traffic around a targeted router. We assume an intelligent attacker who can distinguish type of packets (e.g., MAC-layer ACK), based on physical-layer characteristics etc., and thereby mount jamming in a selective manner. Such an attacker can cause consequences similar to blackhole attack [3], as will be discussed later, without compromising nodes in the network.

Fig. 1 illustrates one such setting. For the jammer, while it can easily cause local damage by interfering with packets originated from the targeted router C (and C's immediate neighbors such as A and E), its ultimate goal is to create bigger and global impacts by continuously dropping packets from other regions that are routed through the targeted router. In this work, we focus on the multi-point-to-point, upward traffic from nodes in a network to a root node. Upward traffic dominates in many such systems (e.g., advanced metering infrastructure). Furthermore, we assume UDP and local MAC-layer retransmissions are used for delivering the upward traffic and there is no end-to-end feedback mechanism between the source and destination.

Under this threat model, we study the attacker's strategies for launching packet drop attacks to increase the damage it can cause. We analyze, focusing on ContikiRPL [4], the behavior of RPL under such attacks and present enhancements to reduce the potential damage. Our contributions include:

- We demonstrate that the RPL implementation in Contiki OS [4] allows an external attacker to continuously drop packets at the targeted router without triggering any rerouting for network recovery. We call this an *externally-induced blackhole attack*.
- We propose three countermeasures and analyze their effectiveness. In particular, we show that adding measures at the targeted router can be more effective than doing so at affected children nodes.
- We evaluate the performance of our enhanced RPL under packet drop attacks. Our results show that our best proposed measure can reduce the average packet loss rate under attack from over 70% to less than 7%. At the same time, it incurs similar routing overhead as the original RPL under no-attack scenarios.

The remaining of the paper is organized as follows. Section II discusses the related work in LLN routing, RPL security, and packet drop attacks. Section III gives an overview of RPL's design and implementation. Section IV presents our threat model. Section V presents the externally-induced blackhole attacks and our measures to deal with such attacks. Section VI evaluates the performance of our measures. We conclude in Section VII.

II. Related Work

The design of routing protocols for LLN has been extensively studied by both academia and industry (e.g., see surveys in [5], [6], [7]), with many carefully engineered solutions proposed to deal with resource-constrained routers and time-varying links, while ensuring that the proposed protocols incur low overhead and can support large-scale networks. Based on these studies, IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [1] has emerged as an important standard in this area. In Section III, we will present some main mechanisms incorporated in RPL to deal with LLN challenges.

As people begin to adopt RPL for communications in critical infrastructures like smart grids [8], [9], [10], its security under different attacks becomes an important concern. Attacks specifically targeting RPL have been studied in literature [11], [3], [12]. Taxonomy of attacks against RPL is presented in [12]. [11] discusses a wide range of attacks, such as sinkhole attacks, HELLO flooding, and clone ID attacks. These attacks can cause disturbance in routing and communication in the network, and the paper demonstrated that attacks cannot be automatically recovered. [3] discusses other types of attacks, such as blackhole attacks, rank attacks, version attacks, and local repair attacks. Typically mounting these attacks requires physical node compromise, insertion of malicious nodes into the network, and/or tampering and injection of messages into the RPL network, which are in practice considered less feasible than jamming attacks when communications are protected by means of cryptographic manners [1], [2]. Security measures to defend RPL networks have also been studied. Intrusion detection systems for securing RPL against topology attacks (e.g., rank attacks) are studied in [13]. Protection mechanisms against rank attacks and version attacks are presented in [14]. Our countermeasures presented in this paper are orthogonal to such schemes by addressing threats of a different category, and thus can be used in a complementary manner.

There have also been extensive research on packet drop attacks in wireless mesh networks, e.g., [15], [16], [17]. In wireless mesh networks, a large fraction of the prior efforts propose to let nodes monitor the transmissions of their neighbors to detect malicious packet drops, by leveraging the broadcast nature of wireless transmissions. However, the resource constraints of routers and the lossy nature of the links can make such detection difficult for LLN. Past efforts that focus on packet drop attacks in LLN include [18], [19]. Many of these prior efforts assume internal attackers, who can tamper legitimate routers in the network. Hence, they did not analyze the damage an external attacker can cause. For internal attackers, their proposed measures include end-to-end notification, packet forwarding confirmation over two hops, multi-path transmission, and coding. Some of the measures (e.g., notification and multi-path transmission) will significantly increase the communication overhead. Regarding notification, the lossy nature of the links can cause false-alarms, which in turn can cause unnecessary packet retransmission and routing switching when the notifications (instead of data packets) are lost during transmission. Pal et al. [20] propose a real-time packet loss detection mechanism for synchrophasor data that are collected by some special power grid devices, however, their approach assumes transmission over wired network and they use the features of the one-way delay experienced by the packets to achieve so, which is not suitable for LLN.

III. An Overview of RPL Design & Implementation

RPL [1] is specifically designed for LLN using the 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) [21], and it can operate with different LLN link layers, including IEEE 802.15.4 PHY and MAC layers. There are a number of RPL implementations, for example, ContikiRPL [4] and the CISCO IOS RPL implementation [22]. While each RPL implementation follows the logics defined in RFC 6550 [1], it may also include some proprietary design for logics that are not specified by the RFC (e.g., the logic for interacting with neighborhood management scheme). In the rest of this paper, we focus on the open-source ContikiRPL implementation [4], which allows us to conduct deeper, source-code level investigation and makes it easy to validate our findings.

In the center of RPL's design is the concept of Destination-Oriented Directed Acyclic Graphs (DODAGs), which aims to support a tree-like structure for routing between nodes and the roots of the corresponding DODAGs. For upward traffic towards the root of a DODAG (e.g., a data concentrator in advanced metering infrastructure), the traffic is forwarded from a child node (e.g., a smart meter) to its chosen parent node in the DODAG, until the packet reaches the root.

Within a certain DODAG, *rank* is utilized to organize and maintain the network topology. Each RPL node has a rank that represents its relative position with respect to the root node. A node uses rank to select the preferred parent among its neighbors. Specifically, in a DODAG, rank is strictly increasing on each path from the root node to leaf nodes. There can be different ways to calculate the rank, and a common way is to use the ETX (Expected Transmission Count) metric [23], which measures the link quality between neighboring nodes. We will use ETX in our following discussion, but the attack and countermeasures we will discuss are applicable to other metric settings. To avoid frequent rerouting, RPL also employs a hysteresis function to damp the switching of routes [24].

There are three types of control messages used in RPL, namely DODAG Information Object (DIO), DODAG Information Solicitation (DIS), and Destination Advertisement Object (DAO) [1]. DIO carries information required for establishing and maintaining DODAG. Among others, the information conveyed in a DIO packet include: 1) ID and version number to identify DODAG, 2) rank, which is described earlier, and 3) Destination Advertisement Trigger Sequence Number (DTSN) flag, which decides whether a new DAO packet should be triggered. When forming a DODAG, DIO is first sent out (multicast) by the DODAG root with the lowest rank value. Each of the receiving neighbors determines its own rank based on the parent's rank and the ETX metric of its link to the parent. After that, it sends DAO to the parent to indicate its intention to be a child. The recipient of DAO, when requested in the message, sends Destination Advertisement Acknowledgment (DAO-ACK) as a response. DAO in this way allows the root or parent to populate information for downward routing. Then each child sends out DIO to its neighbors with its own rank metric. DIS is a message for requesting DIO from neighbors, which is utilized for neighbor discovery when a node joins a network etc. At the end of DODAG construction, each node has a default upward route to the DODAG root.

DIO messages are used to maintain the DODAG in an up-to-date state. In order to manage the routing overhead, RPL uses the Trickle algorithm [25] to control multicast DIO transmissions. Here is a brief description of Trickle algorithm: When the algorithm starts execution, it sets I, the interval for sending multicast DIO packets to I_{min}. When an interval begins, it resets a consistency counter c to 0 and sets its waiting time t to a random value in the range $[I/2, I)$. Whenever a node hears a transmission that is considered "consistent" (e.g., when the rank of its neighbors remain unchanged), it increments the counter c. At time t, a node transmits DIO if and only if c is less than a pre-defined threshold k, which is used to suppress transmissions of routing updates. Under normal situation, when an interval expires, the timer doubles the interval length until it reaches some constant I_{max}. However, if a node receives a transmission that is "inconsistent" (e.g., the detection of a loop), it resets the interval I to I_{min} and starts a new interval to speed up the response.

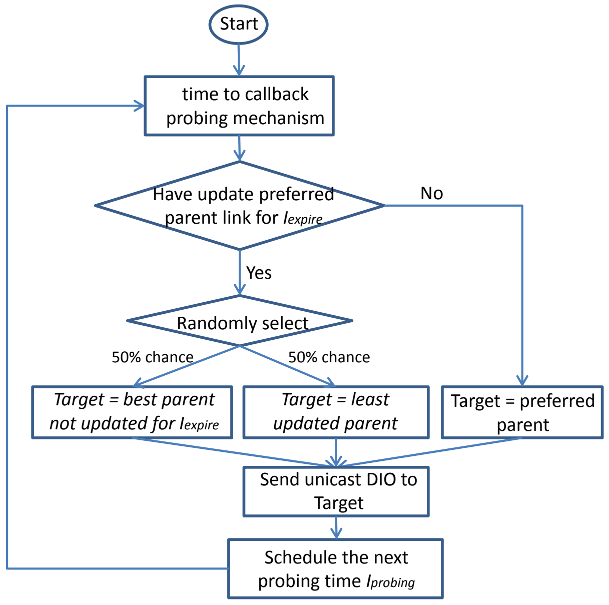
Start
time to callback probing mechanism
Have update preferred parent link for I_{expire}
No
Yes
Randomly select
50% chance
50% chance
Target = best parent not updated for I_{expire}
Target = least updated parent
Target = preferred parent
Send unicast DIO to Target
Schedule the next probing time $I_{probing}$

Fig. 2. Unicast-DIO-based Probing Mechanism in ContikiRPL [4].

TABLE I
RPL PARAMETERS USED IN CONTIKIRPL [4]

	Meaning	Default value
I_{min}	Trickle parameter of the minimum interval to send DIO message	12, for a minimum interval of 2^{12} ms (around 4 seconds)
I_{max}	Trickle parameter of the maximum interval to send DIO message	20, for a maximum interval of 2^{20} ms (around 17 minutes)
k	Trickle parameter of DIO redundancy threshold	10
H	Parent switching hysteresis threshold	128 (in ETX)
$I_{probing}$	Probing interval in Fig. 2	randomly selected in [60s, 180s]
I_{expire}	Probing expiration time in Fig. 2	600s

Nodes in an RPL network need to also maintain the link quality estimation among them. To manage the overhead, ContikiRPL implementation disables the underlying neighborhood management logic at the MAC or IPv6 layer. Instead, it uses unicast DIO messages to maintain the link metrics between neighboring nodes. As shown in Fig. 2, ContikiRPL selects one neighbor randomly at a time to send DIO message periodically. This ensures that the neighborhood update overhead does not increase with the density of the neighborhood. The probing logic of ContikiRPL aims at striking a balance between the update of a node's currently chosen parent, the second best parent, which can serve as the main backup to the preferred parent, and the other neighbors.

Table I summarizes some key parameters used in RPL.

IV. Threat Model

In this work, we focus on the multi-point-to-point, upward traffic that are originated from nodes in a network to a root node. This kind of traffic dominates in many LLN applications, e.g., a smart meter network [7]. We assume UDP and local

MAC-layer retransmissions are used for the upward traffic, and there is no end-to-end feedback mechanism to detect packet loss. While introducing such a feedback mechanism is possible, it could incur high communication overhead, and coordination with RPL's built-in mechanism for responding to network instability could become complex.

We focus on an external attacker, who cannot compromise the internal logic of any legitimate router in the network, or impersonate them to send out fake but legitimately-looking packets. This is achieved using a cryptographic way, e.g,. by authenticating and encrypting all network traffic (including MAC-layer ACK frames) using link-layer security mechanisms as recommended in [1], [2], [26]. We consider a single static attacker that can jam the traffic around a targeted router. We leave the generalization of our results to multiple or mobile attackers for future work.

We assume the jammer can overhear all the packets transmitted by the targeted router and its immediate neighbors. It can also interfere with all the packets received by the targeted router and its immediate neighbors. In Fig. 1, node C is the targeted router, and node A and E are C's immediate neighbors. In the simplest form of the attack, the jammer could keep transmitting interfering signals, hence, all the three nodes, i.e., A, C, and E, will not be able to receive or transmit any packets. However, doing so may not be the best strategy for the attack. In particular, if A, C, and E cannot receive or send any packets, other nodes in the network (e.g., G and H in Fig. 1) will choose routes that bypass them. Hence, the jammer can only make a local impact. Alternatively, if the jammer can find a way to drop packets while continuing to attract traffic from other region, it could make a global and longitudinal impact.

Indeed, a jammer can act more intelligently than blindly interfering with all the traffic. In particular, we assume that the jammer can distinguish between packets sent by different nodes (e.g., by examining the source MAC address if that field is not encrypted, or otherwise by examining physical layer properties such as the received signal strength or the angle of arrival of a transmission). We also assume the attacker can distinguish between data frames and MAC-layer ACK frames. The latter are generated by a receiver to acknowledge the successful receipt of unicast frames. This can be easily achieved because MAC-layer ACK frames have small and constant packet size and are sent with a short delay after the frames they acknowledge. We further assume that the attacker, upon the detection of the starting of a MAC-layer frame, can react fast enough to jam that detected frame if it decides to do so. In particular, a packet will be corrupted if the checksum portion (typically transmitted near the end of the whole MAC frame) is interfered. This gives realistic time buffer for the jammer. With such capabilities, the jammer will be able to conduct the jamming in a selective manner. For example, the attacker can employ a strategy that only jams outgoing packets from the targeted router, but not its incoming packets, nor the MAC-layer ACK packets.

Fig. 3 presents the three main states under the threat model we consider. As shown, without attack, the network should

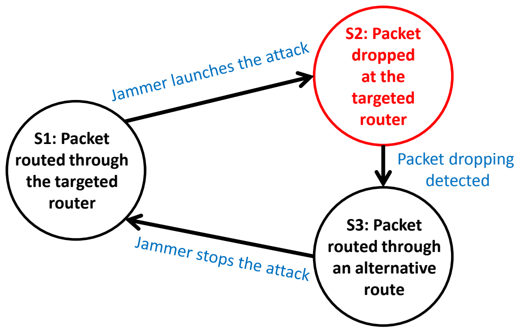
S1: Packet routed through the targeted router
Jammer launches the attack
S2: Packet dropped at the targeted router
Packet dropping detected
S3: Packet routed through an alternative route
Jammer stops the attack

Fig. 3. The three main states of a network under packet drop attacks.

spend most of time running on the S1 state, where packets are routed through the best paths. When the attack happens, the network enters the S2 state, where packets forwarded via the targeted router are dropped. If there are measures to detect the packet drop attack, the children will switch to other routes, hence entering the S3 state. While the network's performance may be degraded in S3 as compared to S1, the attacker can no longer jam more packets than those locally generated by the nodes within its interference range. Hence, to re-enter the S2 state, an attacker may temporarily stop its attack and wait for other nodes to switch back to select the targeted router (i.e., going back to the S1 state). Once that happens, the jammer could launch the attack again. In sum, the attacker' goal is to maximize the fraction of time that the network is in the S2 state, where packets are dropped at the best path.

V. Security Analysis and Enhancement of RPL

A. *Externally-induced Blackhole Attack is Feasible*

Based on the threat model in Section IV, we found out that there is a simple but effective attack strategy that can allow a jammer to continuously drop all packets forwarded by a targeted router, without triggering any rerouting. Specifically, the jammer just needs to do the following:

- Interfere with all outgoing data and routing packets from the targeted router and make sure they cannot be received by the neighbors.
- Do not interfere with any incoming packets to the targeted router, nor the MAC-layer ACK frames corresponding to those incoming packets.

When an attacker carries out the above actions, the targeted router cannot forward any packets to its parent. It also cannot send out routing updates to inform its children of the change of its rank. At the same time, from the children's perspective, everything seems normal: The data packets they send to the parent are received as usual (since both the incoming packets at the targeted router and the MAC-layer ACK frames are not jammed by the attacker). Also, since the attacker jams all the routing update (e.g., DIO, DIS, or DAO packet) from the targeted router, the children nodes do not know that their parent's rank has deteriorated. We call this an *externally-induced blackhole attack* as the attacker does not need to compromise the internal logic of a legitimate router, and it can drop all packets through the targeted router indefinitely.

Intuitively, the children nodes should become suspicious if they do not receive any routing updates from their parent for a prolonged period of time. In fact, we find that ContikiRPL does maintain built-in timer about the valid lifetime of a route [4]. However, the default value of the route lifetime is very long. It is greater than 100 days in the default setting. Even if one reduces the timer, it still cannot detect our attack in the ContikiRPL implementation, since each successful transmission of a unicast packet to its parent resets the timer.

It is worth noting that such a design is not an oversight, but a deliberate choice of ContikiRPL implementation to deal with the challenges in LLN. In particular, RPL's routing update interval is highly optimized by the Trickle algorithm and the neighbor probing mechanism (see Section III) to reduce unnecessary routing and network management traffic. For example, when the network is stable and when the neighborhood size is big, a node may suppress its broadcast DIO messages for unlimited amount of time (when the consistent transmission count is greater than the threshold k). Also, it can take very long for a node to become its parent node's unicast probing target, especially when the network is dense. As a result, even without any attacks, there can be no routing update from the parent node.

One may think that the targeted router should suppress the MAC-layer unicast ACK frames to its children when receiving data packets from them, since the targeted router cannot forward these packets out. However, doing so involves cross-layer coordination, which is difficult to implement. Note that the routing status is only available at the network layer, while the transmission of ACK frames is automatically handled at the data link layer.

B. Enhancements for RPL

We now discuss possible ways to deal with such an externally-induced blackhole attack, while following the original design principles of RPL (including the need to control neighborhood management and routing update overhead and the preservation of layer independence). Our proposed measures will focus on either the targeted router or its affected children nodes, but since there is no prior knowledge on which router is targeted, the proposed measures (for either targeted router or children nodes) should be deployed on all routers.

Measure 1: *Active solicitation of routing update by children.* Specifically, if no routing update is received from its parent for a given duration, a child node will send a unicast DIS packet to the parent to solicit for an update. The unicast DIS packet will trigger the reset of the Trickle timer at the parent, which will send out its DIO within the shortest interval ($I_{min} =$ 4s). With this expectation, if there is still no update from the parent, the child node will select another neighbor as its parent. Note that we use unicast DIS here to minimize the overhead. If multicast DIS is used instead, all the neighboring nodes that receive it will reset the Trickle timer and hence incurs significant communication overhead. The key parameter in this measure is the length of the duration that a child node should wait before sending out the DIS. Making it too long delays the detection and response of an attack. Making it too short, however, can cause additional routing overhead, especially, by resetting the Trickle timer at the parent node.

As presented in Section III, the update from the parent can be either a multicast DIO packet or a unicast DIO packet. The interval for transmitting multicast DIO packets is controlled by the Trickle algorithm, while the transmission of unicast DIO packets is controlled by the probing mechanism, which is illustrated in Fig. 2, in ContikiRPL. For the multicast DIO, the longest interval I_{max} can be 17 minutes. Further, there may not be any update in an interval if there are more than k consistent packets received. In the ContikiRPL implementation, the unicast DIO is sent out at a higher frequency (with an average interval of 2 minutes). However, the unicast DIO could be sent to different targets. Because of the broadcasting nature of the wireless network, a child node could overhear these unicast DIO packets from the parent node, even if the child node is not the designated destination. Although it is technically possible, doing so requires cross-layer coordination, which is not aligned with our design goal. In particular, overhearing unicast packets for other destinations may increase the energy consumption and processing overhead at the resource-constrained routers.

Because of these constraints, we consider that the timer duration should not be set too short. It is reasonable to set it around the interval of $I_{max} = 17$ minutes or more, to avoid the triggering of excessive DIO packets. Otherwise, the expected saving from the Trickle algorithm will not be achieved. This is especially the case if a router has several children nodes. If the timer is too short, the chance that one of the children sends out a DIS to the router to reset its Trickle timer will be high. On the other hand, when a router has a large number of children, a reaction delay of 17 minutes is long enough for a large number of packets to be dropped.

Measure 2: *Anomaly detection at the targeted router and self-disconnection of network interface.* Since Measure 1 described earlier may incur long response delay, we propose to add a countermeasure at the targeted router instead. To counter the externally-induced blackhole attack, a router under attack needs to recognize that it keeps receiving data packets from children nodes while it cannot forward their packets out or send routing updates to those children nodes. Once such a situation is detected, the router can conclude that it is likely being abused for a packet drop attack, and it can (temporarily) withdraw itself from the network. To avoid any complicated cross-layer coordination, a simple way for a node to withdraw itself is to shutdown its network interface. This will prevent the MAC-layer ACK from being sent back to the children. Hence, a child node will realize that its link to the preferred parent is no longer available, which in turn triggers the child node to switch to alternative paths.

Under this approach, the delay till recovery mainly consists of two parts: 1) the time for the targeted router to identify the packet drop attack situation and 2) the time for the children to respond to the loss of its link to the parent.

Delay for 1) does not take much longer than a few packet-

TABLE II

A SUMMARY OF PROPOSED MEASURES

	Actor	Design	Key parameter(s)
M1	Child nodes	Solicit update from parent upon timeout	Duration of timeout
M2	Targeted router	Disconnect itself upon anomaly detection	ETX threshold for link to parent; # of retries
M3	Targeted router	Delay recovery from self-disconnection	Self-disconnection duration

forward failures. Namely, the failure of packet forwarding will increase the targeted router's ETX metric to its parent. The router may switch over multiple parents, but as long as the attack proceeds, the ETX metrics to all candidate parents will increase. As a result, the targeted router will increase its own rank, until the rank becomes larger than that of its children nodes. Once this happens, a data packet from its child node will trigger the detection of a loop (recall that in RPL, the path to root should have strictly decreasing ranks).

Intuitively, the detection of a loop should trigger some immediate exchange of routing messages to update the routes. However, to control routing message overhead, RPL deliberately chooses to not to consider a single inconsistency along a path as a critical error (see Section 11.2.2.2 of RFC 6550 [1]). In ContikiRPL implementation, the reset of the Trickle timer to the minimum value ($I_{min} = 4$s) will only be triggered when a packet experiences the second inconsistency while being forwarded [4]. In an externally-induced blackhole attack, the targeted router could be the first and only inconsistency in a path, so it should reset the Trickle timer even if there is only a single inconsistency in the path. While making this change, to reduce false positive, we require the targeted router to also check its ETX value to its chosen parent. The reset of Trickle timer will be triggered only if a loop is detected and the link ETX value to parent is above certain threshold.

The DTSN flag in a multicast DIO packet will trigger the transmission of DAO packet [1] by the receiving neighbors within 6 seconds. Hence, within at most 10s, the child should resolve the loop and report that to the targeted router with an updated DAO packet under a normal situation. If this does not happen, it could be because of a packet drop attack. However, it could also be due to natural loss of the multicast DIO or the resulted DAO packets. Since self-disconnection is a major decision, the node may want to wait for a few repeated failures to get DAO packet before entering that state.

Delay for 2) depends on the rank difference between the current path and an alternative path, as well as the hysteresis threshold for triggering the route switching, but again, after the targeted router shuts down its interface, it will likely only take the failure of a few transmissions to trigger the routing switching. Overall, this mechanism can potentially work faster than Measure 1 to let the children move from S2 to S3, especially when there are frequent traffic being forwarded in the region under attack.

Measure 3: *Delayed recovery from self-disconnected mode.* While Measure 1 and Measure 2 can trigger the children nodes to switch to alternative routes (i.e., transit from state S2 to S3 in Fig. 3), an intelligent attacker can react to that, by temporarily stopping the attacks, so as to re-attract the traffic back to the targeted router (i.e., the network re-enters the state S1 in Fig. 3). Once the network enters the S1 state, the attacker can re-launch the attack to let the network go back to the state S2. Assume, for example, the network stays in State S2 for 5 minutes on average, and it stays in state S3 and S1 for another 5 minutes before it goes back to S2. In this case, the attackers can still effectively drop 50% of the packets that may route through it over a long run.

To reduce the fraction of time the network may stay in S2, RPL needs to distinguish between the following two types of route switching, i.e., the switching due to the deterioration of the current parent and the switching due to the appearance of a better alternative. Note that, the first case is related to the recovery from the attack (i.e., the state change from S2 to S3), while the second case could lead to a potential new attack (i.e., from S3 back to S1). We want to ensure the switching in the first case is as fast as the current RPL implementation, while delaying the second type of switching when the network is under attack.

While such a differential treatment can probably be done at the children node, a simpler solution based on Measure 2 is to let the targeted router, which has disabled its network interface, stay in the self-disconnected mode for a longer period of time. Note that, this strategy will affect the targeted router's own packet transmission. Also, though it is outside the scope of this paper, increasing the duration of self-disconnected mode may actually help a mobile attacker to increase the damage it can cause (i.e., attacking different routers one-by-one to make them all off-line). However, in our current threat model with static attackers, a longer shutting down time serves as an effective measure to slow down the network from reentering the state S2. The key parameter for this countermeasure is the duration for a node to stay in the self-disconnected mode.

Table II summarizes the three proposed measures.

VI. EVALUATION

We evaluate the ContikiRPL implementation (version 3.x) [4] under our attacks using Contiki's companion Cooja simulator. We also implemented the three measures proposed in Section V-B. We evaluate their performance in terms of two key metrics: 1) packet loss rate, i.e., the fraction of data packets that do not reach the root, and 2) routing overhead, i.e., the average number of routing packets (including DIS, DIO, and DAO packets) transmitted in the whole network every minute. We evaluate both attack and no-attack scenarios.

As shown in Fig. 4, we consider a network consisting of 50 Tmote Sky [27] nodes that are randomly distributed in a $280m \times 150m$ rectangle area, where the root node is placed at the upper right corner. We use the Unit Disk Graph Medium radio model in Cooja, with a transmission range of 50m and an interference range of 70m. The transmission success rate is set to 90%. Additionally, packet losses can happen due to packet collisions. We use the default setting in ConkitiRPL, where the queue at MAC layer can hold up to 8 packets and a unicast packet can be transmitted for maximum 5 times if

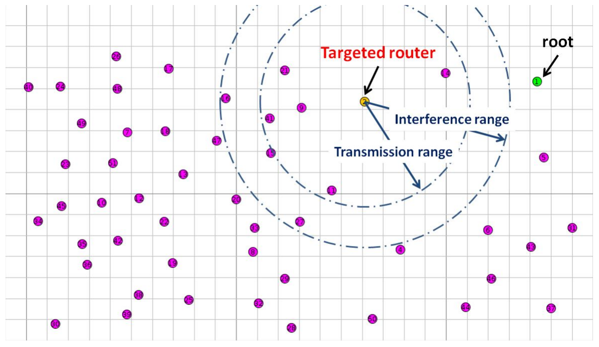

Fig. 4. Experiment setting.

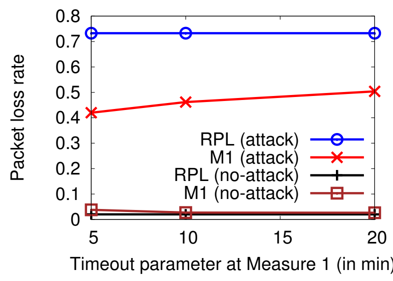

Fig. 5. Packet loss rate under different timeout settings of Measure 1, as compared to RPL.

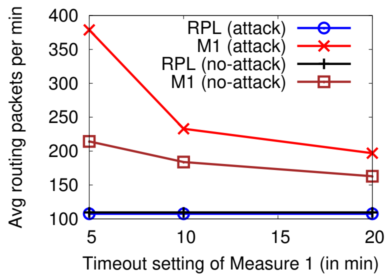

Fig. 6. Routing overhead under different timeout settings of Measure 1

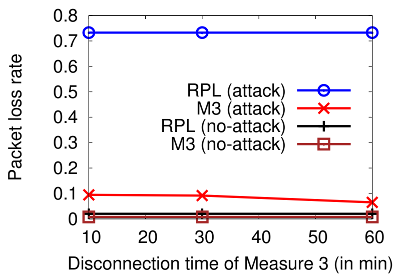

Fig. 7. Packet loss rate under different self-disconnection duration settings of Measure 3

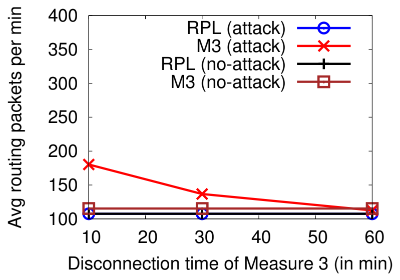

Fig. 8. Routing overhead under different self-disconnection duration settings of Measure 3

no ACK is received. We simulate the upward traffic by letting each node in the network generate one packet every 60 seconds (to stagger the traffic, each source adds a random offset in the range of 0 to 60 seconds before transmitting the packet).

We choose a node that is 2 hops away from the root as the targeted router. In the no-attack scenario, this node helps forward packets from around 35 nodes. In the attack scenario, the attacker adopts the strategy described in Section V. It will jam the network until it detects the network has switched to state S3 (i.e., no more routes through the targeted router). At that moment, the attacker stops jamming and waits for the network to go back to state S1 (we define S1 as at least half of its children reselect it as the parent). Our experiments simulate each setting for 3 to 8 hours, so the network transits from the state S1, S2, and S3 for at least 5 times under attack. We report the average values of the measured metrics.

Recall that the key parameter of Measure 1 is the length of the timeout interval before a child node solicits a routing update from its parent. We evaluate three different timeout parameters, namely 5 mins, 10 mins, and 20 mins. Fig. 5 and Fig. 6 show the packet loss rate and the routing overhead respectively under different timeout settings. We also plot the original RPL's performance for comparison. The original RPL incurs 2% loss rate and 110 routing packets per minute when there is no attack. When we launch the attack, the packet loss rate increases to 73.3%. The number of routing packets reduces slightly to 107 packets per minute when there is an attack. This is because the dropping of the DIO packets from the targeted router results in less DAO packets from its neighbors. As can be seen, Measure 1 can reduce the packet loss rate to 42% when using 5 minute timeout parameter. However, this comes at a price. It incurs 378.8 routing packets per minute on average under attack. Even when there is no attack, it still incurs 214.3 routing packets per minute on average. Increasing the timeout parameter can reduce the routing overhead, but the packet loss rate increases due to slower response. For example, using 20 minutes as the timeout interval reduces the routing overhead to 197 packets per minute on average under attack and 163 packets per minute when there is no attack. However, the packet loss rate increases to more than 50%.

The key parameters for Measure 2 are the ETX threshold for a router's link to its parent and the number of times that a router tries to send loop-detection notification to its children. We set the ETX threshold to 1280 and the number of retries for getting routing updates from the children to 5. Our experiment shows that with this setting, our scheme is able to always shutdown the targeted router when the attack is launched. Meanwhile, it does not cause any false positive where a non-targeted router disconnects itself. In Measure 2's default setting, we let a router disconnect itself for a duration of 10 minutes. This value is chosen to ensure that in all the state transition cycles, the targeted router is shutdown sufficiently long to let all its children switch to alternative routes. As such, Measure 2 can be viewed as a special case of Measure 3 with disconnection time of 10 minutes, hence we report its performance in Fig. 7 and Fig. 8 together with other self-disconnected duration settings of Measure 3. As can be seen in the figures, Measure 2 reduces the packet loss rate to 9.45%, while incurring a routing overhead of 180.3 packets per minute. It outperforms Measure 1 in both metrics. This is because: 1) Measure 2 can trigger the recovery action faster. On average, Measure 2 only needs 3.5 minutes to trigger the rerouting. 2) Under Measure 2, after the targeted router disconnects itself, its children nodes increase their ETX metrics for their links to the targeted route. Hence, it makes the network stay in state S3 longer before those children nodes switch back to the targeted router. In comparison, in Measure 1, the children nodes just change the rank value of the targeted router, but maintain the good ETX metrics to it. Hence, when the attacker stops the jamming and the targeted router regains its good rank, the children nodes switch back to the targeted router quickly. The figures also show that, in no-attack scenario, Measure 2's packet loss rate and routing overhead are both similar to the original RPL.

Lastly, we investigate Measure 3's capability to further reduce the packet loss rate by increasing the self-disconnection duration to 30 minutes and 60 minutes before the targeted router rejoins the network. As can be seen in Fig. 7 and Fig. 8, the longer the self-disconnection time used, the lower the packet loss rate. For example, with disconnection time of 60 minutes, the average packet loss rate is reduced to 6.5%. This is because the network spends an increasing fraction of time in the state S3. For the same reason, the average routing overhead also decreases. However, the self-disconnection affects the targeted router's own traffic. In practice, there may be requirement in the minimal amount of information to be reported by each node, so we should take it into consideration to find optimal duration according to that constraint.

In summary, our evaluation results show that the proposed measures at the targeted router (i.e., Measure 2 and 3) can better reduce the number of packets dropped under an attack scenario. Moreover, they have the potential to achieve so without incurring much overhead as compared to the default RPL implementation under no attack scenario.

VII. Conclusion

In this work, we show that an existing RPL implementation allows an external jammer to continuously drop packets without triggering rerouting. We propose and analyze several measures and show the impact of our enhancement using Cooja simulator. In future work, we will consider variants of packet drop attack models (e.g., a mobile attacker, multiple collaborating attackers, attackers who do not drop all outgoing packets) and evaluate the impact of proposed measures using more thorough evaluations.

Acknowledgment

This research is partly supported by the National Research Foundation, Prime Minister's Office, Singapore under the Energy Programme and administrated by the Energy Market Authority (EP Award No. NRF2014EWT-EIRP002-040) and in part by the research grant for the Human-Centered Cyber-physical Systems Programme at the Advanced Digital Sciences Center from Singapores Agency for Science, Technology and Research (A$*$STAR).

References

[1] IETF, "Rfc 6550: Rpl: Ipv6 routing protocol for low-power and lossy networks," 2012.

[2] ——, "Rfc 8180: Minimal ipv6 over the tsch mode of ieee 802.15.4e (6tisch) configuration," 2017.

[3] P. Pongle and G. Chavan, "A survey: Attacks on rpl and 6lowpan in iot," in *Pervasive Computing (ICPC), 2015 International Conference on*. IEEE, 2015, pp. 1–6.

[4] Contiki team, "ContikiRPL: the new Default Contiki IPv6/6lowpan Routing Protocol," [Online]. Available: https://github.com/contiki-os/contiki/releases.

[5] D. Gadde and M. Chaudhari, "Survey on routing protocol for low-power and lossy networks," in *IEEE ICCIC Conference*, 2015.

[6] E. Aljarrah, M. B. Yassein, and S. Aljawarneh, "Routing protocol of low-power and lossy network: Survey and open issues," in *Engineering & MIS (ICEMIS), International Conference on*. IEEE, 2016, pp. 1–6.

[7] O. Gaddour and A. Koubaa, "Rpl in a nutshell: A survey," vol. 56, no. 14, pp. 3163–3178, 2012.

[8] D. Wang, Z. Tao, J. Zhang, and A. A. Abouzeid, "Rpl based routing for advanced metering infrastructure in smart grid," in *IEEE ICC Communications Workshops*, 2010.

[9] D. Popa, J. Jetcheva, N. Dejean, R. Salazar, J. Hui, and K. Monden, "Applicability statement for the routing protocol for low power and lossy networks (RPL) in AMI networks," *Internet draft*, 2011.

[10] R. Berthier, J. G. Jetcheva, D. Mashima, J. H. Huh, D. Grochocki, R. B. Bobba, A. A. Cárdenas, and W. H. Sanders, "Reconciling security protection and monitoring requirements in advanced metering infrastructures," in *SmartGridComm*, 2013.

[11] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the rpl-based internet of things," *International Journal of Distributed Sensor Networks*, vol. 9, no. 8, p. 794326, 2013.

[12] A. Mayzaud, R. Badonnel, and I. Chrisment, "A taxonomy of attacks in rpl-based internet of things," *International Journal of Network Security*, vol. 18, no. 3, pp. 459–473, 2016.

[13] A. Le, J. Loo, Y. Luo, and A. Lasebae, "Specification-based ids for securing rpl from topology attacks," in *Wireless Days (WD), 2011 IFIP*. IEEE, 2011, pp. 1–3.

[14] A. Dvir, L. Buttyan *et al.*, "Vera-version number and rank authentication in rpl," in *IEEE MASS Conference*, 2011.

[15] D. Djenouri, L. Khelladi, and N. Badache, "A survey of security issues in mobile ad hoc networks," *IEEE communications surveys*, vol. 7, no. 4, pp. 2–28, 2005.

[16] B. Kannhavong, H. Nakayama, Y. Nemoto, N. Kato, and A. Jamalipour, "A survey of routing attacks in mobile ad hoc networks," *IEEE Wireless communications*, vol. 14, no. 5, 2007.

[17] J. Cai, P. Yi, J. Chen, Z. Wang, and N. Liu, "An adaptive approach to detecting black and gray hole attacks in ad hoc network," in *IEEE AINA Conference*, 2010.

[18] K. Heurtefeux, O. Erdene-Ochir, N. Mohsin, and H. Menouar, "Enhancing rpl resilience against routing layer insider attacks," in *IEEE AINA*, 2015.

[19] A. Kumar, R. Matam, and S. Shukla, "Impact of packet dropping attacks on rpl," in *Parallel, Distributed and Grid Computing (PDGC), 2016 Fourth International Conference on*. IEEE, 2016, pp. 694–698.

[20] S. Pal, B. Sikdar, and J. Chow, "Real-time detection of packet drop attacks on synchrophasor data," in *IEEE SmartGridComm*, 2014.

[21] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet*. John Wiley & Sons, 2011, vol. 43.

[22] "Routing Protocol for LLN (RPL) Configuration Guide," [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/rpl/configuration/15-mt/rpl-15-mt-book.html, (Date last accessed on Sep. 10, 2017).

[23] IETF, "Rfc 6551: Routing metrics used for path calculation in low-power and lossy networks," 2012.

[24] ——, "Rfc 6719: The minimum rank with hysteresis objective function," 2012.

[25] ——, "Rfc 6206: The trickle algorithm," 2011.

[26] T. Tsao, R. Alexander, M. Dohler, V. Daza, A. Lozano, and M. Richardson, "A security threat analysis for the routing protocol for low-power and lossy networks (RPLs)," Tech. Rep., 2015.

[27] "Tmote Sky," [Online]. Available: http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf, (Date last accessed on Sep. 10, 2017).