A Capacity-Aware Distributed Denial-of-Service Attack in Low-Power and Lossy Networks

Rajorshi Biswas, Jie Wu, and Xiuqi Li
Department of Computer and Information Sciences
Temple University, Philadelphia, PA, USA

***Abstract*—Low-Power and Lossy Network (LLN) is composed of embedded devices with limited power, memory, and processing resources. LLN has a wide variety of applications including industrial monitoring, connected home, health care, urban sensor networks and environmental monitoring. LLN uses Routing Protocol for Low-power and lossy networks (RPL) protocol. The RPL maintains directed acyclic graphs for routing packets. By exploiting some features, a Distributed Denial-of-Service (DDoS) attack can be conducted easily. DDoS attacks are very popular and well studied in the context of the Internet, but not in the context of LLNs. In this paper, we propose a powerful DDoS attack framework in LLNs. We formulate the attack as an optimization problem for selecting an optimal set of attackers and their targeted neighbors constrained by a limited link bandwidth. We propose an optimal solution by transforming the optimization problem into a max-flow problem. We provide simulations to support our model.**

***Index Terms*—Low-Power and Lossy Network, Distributed Denial-of-Service, Security, Attack Strategy, Powerful DDoS attack**

I. Introduction

In a denial-of-service attack (DoS attack) the attacker makes a service or resource temporarily unavailable to its users. DoS attacks are considered a federal crime and the penalty includes years of imprisonment [1]. Usually DDoS attacks are conducted by compromised machines called bots. Bots are malicious programs in users' computers. A coordinator called the master controls the bots. The master commands the bots to send a huge number of requests. Because of the huge number of packets, the bandwidth of the victim becomes exhausted.

DDoS attacks are popular in traditional networks and a lot of defense mechanisms are also available, such as [2, 3]. The scope of DDoS is not limited to traditional networks only. One of the potential playgrounds of DDoS is Low-Power and Lossy Networks (LLNs). LLNs are composed of embedded devices with limited memory, energy, and processing resources. The interconnected links are also heterogeneous. Different links can use different protocols including IEEE 802.15.4, bluetooth, or low power Wi-Fi. The LLNs use Routing Protocol for LLNs (RPL) for packet routing [4]. One or multiple directed acyclic graphs (DAGs) are maintained by the roots of the network. The roots are special devices with higher capability and internet/WAN connectivity. The roots maintain destination oriented DAGs (DODAGs) for routing packets. When a device wants to send a packet to another, the packet is routed to the root. After that, the root uses strict source routing to deliver the packet to the destination. In strict source routing, a packet header contains hop by hop routing information. The intermediate nodes follow the routing information in the header. When a node cannot deliver a packet with routing information to the next hop, it drops the packet and sends an error message to the source. The error message travels through the root like regular packets.

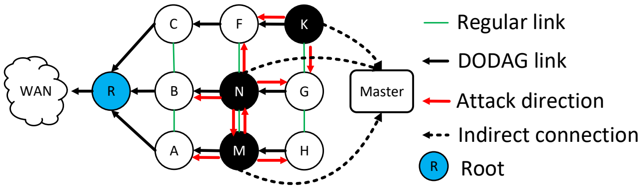
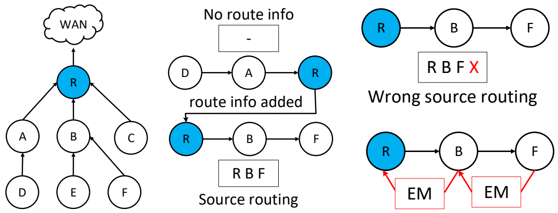


Fig. 1: A DDoS attack in a LLN.

The error messages in RPL bring a great opportunity for DDoS attacks in LLNs. A malicious device can create many packets with the wrong routing information and forward them to its neighbors. When the packets arrive, they drop them and send error messages to the source. When many error messages come to the root, the capacity of links is exhausted at the root and DoS occurs. Fig. 1 shows an attack scenario in LLN. Nodes M, N, and K are attackers and they are controlled by a coordinator called the master. The attackers are compromised devices and the master can be one of them. The master's goal is to maximize the attack strength. The easiest way to achieve this is to activate all the attackers. In reality, the strongest attack can be launched by activating a subset of the attackers. This is because each link has a limited bandwidth and attackers can attack a limited number of neighbors. Attacking a neighbor means sending packets with the wrong routing information at the highest rate. For example, if the master selects N and K, and they both attack F and G, then the attack intensity will not be the highest. Assume an attacker can attack two neighbors at the same time and the bandwidth of each link is the same. This is because error messages from F for N and K's packets travel through link $F \rightarrow C$ which has a limited bandwidth. Error messages from G travel back through link $G \rightarrow N$ which is used for the attack. Therefore, the master needs to select the attackers and the targeted neighbors wisely.

In this paper, we propose an attack framework called Capacity-Aware DoS Attack in LLN (CADAL). To the best of our knowledge, for the first time we treat the attack as an optimization problem for selecting an optimal set of attackers and their targeted neighbors under the condition of limited link bandwidth. We assume that the capacity of each link is the same and each attacker can attack a limited number of neighbors simultaneously. We solve the problem by transforming it into a max-flow problem. We conduct simulations and the results support our model.



(a) DODAG. (b) Packet routing. (c) Error message.

Fig. 2: RPL routing in LLN.

The remainder of this paper is arranged as follows: Section II reviews some related work. Section III introduces some background information and the network model. Section IV describes the proposed attack framework. In Section V, we present some simulation results that demonstrate the effectiveness of our proposed model. Section VI concludes the paper.

II. Related Work

Though RPL is not new, its increased usage in IoT motivates several research works in this field recently.

In a selective-forwarding attack [5], attacker nodes selectively forward packets to disrupt routing paths. For example, an attacker forwards all RPL control messages but drops the other traffic. This attack can be launched with other attacks, such as sinkhole attacks. One way to protect against selective-forwarding attacks is to maintain multiple disjoint paths between the root and the destination nodes. The root needs to dynamically select the paths. In [6], authors implemented the selective-forwarding, sinkhole, hello flood, wormhole, and clone ID attacks in LLN. All but the selective forwarding attack are originally targeted at wireless sensor networks. They propose a method for placing intrusion detection systems and eliminating malicious nodes in the LLN.

In the storing mode of RPL, the packet is forwarded to the least common ancestor of the source and destination. The least common ancestor node uses the source routing like the root. An attacker node generates a large number of packets and forwards them to its neighbors so that the least common ancestor needs to forward a considerable number of packets. This drains the energy of that node. This kind of DDoS attack is called energy depletion attack [7]. The authors also propose a misbehavior-aware threshold detection scheme to prevent the energy depletion attack. Other types of attacks include sinkhole, application layer, jamming, and cloning of things attacks [8, 9].

The work most related to ours is [10]. They first identify a new DDoS attack in RPL-based LLNs, called a hatchetman attack. The attackers send packets with the wrong routing information to all of their neighbors. The error messages created by the neighbors make DoS at the root of DODAG. The hatchetman attack does not consider the limited bandwidth of links. Because of the limited link bandwidth, many error messages may be dropped before reaching the root. Besides, the same number of error messages can be generated by attacking a smaller number of neighbors.

III. Network Model

In this section, we describe some background information and wireless settings. There are several versions of RPL. We present the version of RPL we are considering for our work.

A. RPL Summary

LLNs are composed of the constrained routers and interconnections. LLN routers/devices operate with limited processing power, memory, and battery power. Their interconnections have high loss rates, low data rates, and instability. The routers/devices do not store any routing table. There is a highly capable gateway called the root which is connected to the Internet. All the packets are routed through the root. The root maintains a DAG called the destination oriented DAG (DODAG) structure to route the packets. Other devices do not store routing information. We divide the operations of RPL into three parts: DODAG maintenance, packet routing, and error handling.

DODAG Maintenance: To build a DODAG, the root first broadcasts a control message to its neighbors. The control message contains the rank of the root. The rank is basically the number of hops from the root. The rank of the root is 0. When a neighbor receives that control message for the first time it increases its rank and makes the sender its parent. Then it broadcasts its rank to its neighbors. When a device decides its parent it reports that information to the root. The root then creates the DODAG by using the parents' information from each device. The DODAG indicates the network state, which changes over time. Fig. 2(a) shows a DODAG for a LLN. In the figure, R is the root and connected to WAN. The arrows represent the parent relationship between two nodes.

Packet Routing: When a device wants to send a message to another, it creates a packet and forwards it to its own parent. The parent device forwards the packet to its own parent. When the packet arrives at the root, the root finds out the path to the destination and piggybacks that information in the header of the packet. Then it forwards the packet to the next hop router according to the path. When a router/device receives a packet with routing information it forwards that packet to the next hop according to that route information. Strict source routing is used in RPL. It means that the routing information is the hop by hop routers/devices information to the destination. The forwarding routers/devices do not remove their own id during forwarding. Fig. 2(b) shows packet routing in a LLN. In the figure, D sends a message to F. D does not know the route, so that it forwards the packet to its parent A. The packet does not contain any routing information. A forwards the packet to its own parent R. The root R knows the routing path to F. So, it adds the routing information to the packet. Then, R forwards the packet to the next hop. Intermediate nodes follow the routing information in the packet header. Finally, the packet arrives at the destination F.

Error Handling: In LLN links have low data rates and instability. Packets suffer from high loss. If a router cannot deliver a data packet to the next hop, then it creates an error message and sends back to the source of the packet. The error message contains a code of ”Error in Source Routing Header”. The message follows the same packet routing rule. So, the error message reaches the root, then it is source routed to the source of the data packet. Fig. 2(c) shows a routing error in RPL. D sends a packet with X being the destination. When the packet arrives at R, it adds routing information $\{R, B, F, X\}$. The packet arrives at F via B. Then, F tries to forward it to X which is unavailable. Therefore, F drops the packet and sends an error message to D. The error message is forwarded to F's parent B. B sends it to its own parent R. Then R uses source routing to send the message to D.

B. Wireless Settings

In LLNs, the capabilities of devices and interconnected links are heterogeneous. For simplicity, we assume that there is only one root. All devices are homogeneous and interconnected wireless links have the same bandwidth. The attackers also have the same capability as other devices. The attackers are compromised devices that are controlled by the master. The master resides either inside or outside the LLN. All the attackers send their neighbors' reporting paths to the master. The reporting path of a node is the path from that node to the root. The reporting path can be inferred by eavesdropping on the packets of the neighbors. For example, node E can eavesdrop on the packets which are forwarded from B to F. Node E remains in the transmission range of B. Therefore, any packet transmitted from B clearly reaches F. The packets contain routing information ($R \rightarrow B \rightarrow F$) in their header. The reporting path of F is simply the reverse of the routing information ($F \rightarrow B \rightarrow R$). Sometimes it is not possible to get all neighbors' reporting paths. For example, node E cannot get the reporting path of node D. This is because E is not A's neighbor, and D does not have any children. Therefore, A's forwarded packets do not reach E and D does not forward any packet with routing information.

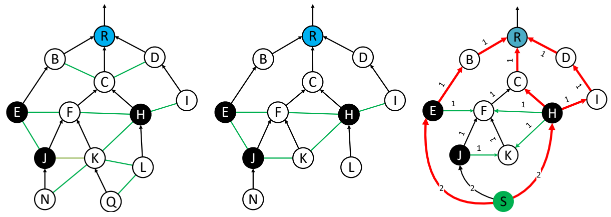
An attacker can attack a limited number of neighbors. If the links with the neighbors are in different channels (different frequencies) then it is possible to attack multiple neighbors at the highest data rate. In real LLNs, the number of simultaneous attacks is different for different attackers. For simplicity, we assume that each attacker can attack the same number of neighbors simultaneously.

IV. Capacity Aware Ddos Attack in LLN (CADAL)

In this section, we propose the CADAL framework. We formulate an optimization for finding best target set of neighbors.

A. The Attack Optimization Problem

In the CADAL framework, the attackers report their neighbors' reporting paths to the master. The master combines the reporting paths and forms a DAG. The DAG is not the same as the real DODAG maintained by the root. This is because the master cannot get the reporting paths of all nodes. Therefore, the master does not have a global view of the DODAG. With the partial DODAG information, the master wants to maximize the strength of the attack against the root. At the same time, the master wants to attack the minimum number of neighbors. Attacking a neighbor is associated with a cost. The cost is energy consumption for computation and transmission.



(a) Actual topology. (b) Master's view G. (c) Max flow in G'.

Fig. 3: An example.

Algorithm 1 Find an Optimal Set of Target Neighbors

```
Input: The topology $G = (V, E)$, the set of attackers $A$, the number of
    simultaneous attacks launched per attacker $P$.
Output: An optimal mapping $F$ and target nodes $T$.
Procedure: FIND-OPTIMAL-TARGET-SET($G, A, P$)
    Find the neighboring set NEI($a$) for each $a \in A$.
    Prune NEI($a$) for each $a \in A$ using elimination rules.
    $G' \leftarrow$ TRANSFORM($G', A, P$)
    Find $\forall_{e \in E'} Flow[e]$ using Ford-Fulkerson algorithm.
    $(T, F) \leftarrow$ FIND-TARGET($G, A, Flow$)
    return $(T, F)$.
```

The strength of an attack is determined by the rate of error messages delivered to the root of a LLN. Let $G = (V, E)$ be the DAG formed from the received reporting paths and Δ be the degree of the root. Let R be the maximum data rate of a link. If there are enough attackers, then the maximum possible strength of the attack is $R\Delta$. For simplicity we assume that $R = 1$. So, the maximum possible attack strength is Δ.

In the optimal scenario, the number of attacked neighbors is equal to the maximum attack strength. The master needs to find an optimal set of target neighbors and the attackers that will produce the attack of strength Δ.

Let A be the set of attackers and $T \subset$ NEIGHBOR(A) be the set of target neighbors. P is the maximum number of attacks that an attacker can launch simultaneously. $F : T \rightarrow A$ is the mapping from the target to the attacker. For example, $F(t)$ denotes the attacker which attacks the node t. $S(T)$ is the strength achieved by target set T. The problem can be expressed as the following:

$$\begin{aligned} &\text{minimize} \quad |T| \\ &\text{subject to} \quad \forall_{a \in A} |F^{-1}(a)| < P, \quad S(T) = \Delta \end{aligned} \tag{1}$$

B. An Optimal Solution

We transform the optimization problem of finding a minimum set of target neighbors to a maximum flow problem. Specifically, in the CADAL approach, we first eliminate neighbors that don't make much contribution to the goal of achieving the maximum attack strength. Then we create a flow graph by augmenting the reporting path graph with additional nodes and edges. We use the Ford-Fulkerson algorithm to compute the maximum flow. An optimal set of target neighbors can be obtained based on the initial pruned neighbor set and the maximum flow information. The entire approach is detailed in Algorithm 1. Procedures TRANSFORM and FIND-TARGET are not shown in detail to save space.

*a) **Neighbor Pruning**:* We first find the set of neighbors of each attacker and eliminate some of them based on the two rules. **Rule 1:** If an attacker has another attacker as a neighbor, then this neighbor is removed from the target neighbor set. An attacker is not a good target neighbor because it does not have to be forced to create error messages. The master can directly order to do so. **Rule 2:** If an attacker remains on all of the reporting paths of a neighbor, then the neighbor can be removed. The reason is similar to the previous case.

*b) **Flow Graph Creation**:* After pruning, we transform G into a flow graph $G' = (V', E')$. We keep all edges on the reporting paths in G. We add a virtual source S to G'. Then we connect S to each attacker by adding an edge from S to the attacker. For each attacker, we add an edge from the attacker to each neighbor in its pruned neighbor set. Each edge from S gets the capacity P. All other edges get the capacity 1. The capacity P is used to limit the outgoing flows at an attacker. The outgoing flows can travel at most P links.

*c) **Optimal Target Set Computation**:* The flow from S travels through the target neighbors in T. The outgoing flows from an attacker also indicate the target to attacker assignment F. The maximum flow in G' is equal to the maximum strength attack launched by the given set of attackers. The maximum flow problem is solved using the Ford-Fulkerson algorithm. It is a greedy algorithm that computes the maximum flow in a flow network. With the maximum flow information, the set of target neighbors T can be computed. For each attacker, we find the links with flows. The node at the other end of each link becomes the target of the attacker.

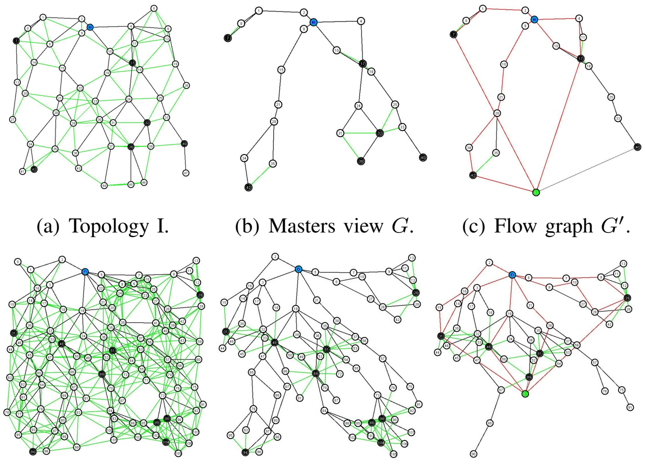
C. Example

Let us consider the topology in Fig. 3(a). The black arrows form DODAG and the green lines are general links. We assume that an attacker can attack at most two neighbors ($P = 2$) in this example. Node R is the DODAG root. Nodes E, J, and H are the attackers. After receiving the reporting paths of the neighbors from the attackers, the master generates the partial DAG G (Fig. 3(b)).

Next, we remove some of the nodes according to the elimination rules. The neighbor sets of E, J, and H are $\{J, F, B\}$, $\{E, F, K, N\}$, and $\{C, I, L, K, F\}$, respectively. According to the elimination rule 1, we remove E from J's neighbor set and J from E's neighbor set. According to the elimination rule 2, we remove L from H's and N from J's neighbor sets. Therefore, pruned neighbor sets of E, J, and H are $\{F, B\}$, $\{F, K\}$, and $\{C, I, K, F\}$, respectively.

Then, we formulate G' from G by first adding a virtual source S. We add three edges from S to the attackers E, J, and H. The capacities of these edges are 2. Then we add some more edges with capacity 1 from the attackers to their pruned neighbors.

Next, we use the Ford-Fulkerson algorithm to find a maximum flow in G', shown in Fig. 3(c) (marked as red). The maximum flow is 3, which is the highest achievable attack strength in this scenario. The flow travels through the attackers E and H. Therefore, the master will activate the attackers E and H. As the flow travels to B from E, attacker E will attack B. Similarly, the flow from H travels through C and I. So, the attacker H will attack C and I. At this point we have found our desired target set $T = \{B, C, I\}$ and the mapping function F ($F(B) = E$, $F(C) = H$, and $F(I) = H$).



(a) Topology I. (b) Masters view G. (c) Flow graph G'.

(d) Topology II (e) Masters view G. (f) Flow graph G'.

Fig. 4: Topologies used in simulation.

Theorem 1. *The complexity of Algorithm 1 is $O(|V|^2 + |V|\Delta)$.*

Proof. In Step 2 of Algorithm 1, finding neighbor sets of all attackers takes $O(N)$. Elimination based on rule 1 takes $O(|V|^2)$. Elimination based on rule 2 needs $O(|V|log(|V|))$ time. Therefore, Step 3 takes $O(|V|^2)$ time. The creation of the flow graph in Step 4 also takes $O(|V|^2)$ time. Step 5 takes $O(|V|\Delta)$ which is the running time for Ford-Fulkerson. Step 6 takes $O(|V|^2)$ time. Therefore, the overall running time of Algorithm 1 is $O(|V|^2 + |V|\Delta)$. □

Theorem 2. *Algorithm 1 provides an optimal target set.*

Proof. The procedure TRANSFORM converts the master's DAG G to flow graph G' in such a way that each attacker's incoming edge has the capacity P and each outgoing edge has the capacity 1. Therefore, no more than P links can carry flows. So, the constraint that an attacker cannot attack more than P neighbors simultaneously holds. Next we need to show that the target set T is the minimum. Let us assume that there is another set T' which is the minimum ($|T| > |T'|$). As each of the nodes in T (or T') has only one incoming link and the capacity of each link is 1, the maximum flow is $|T|$ or $|T'|$. According to the assumption $|T| > |T'|$, the target set T' cannot produce an attack of the maximum strength. Therefore, T' is not an optimal target set. □

V. SIMULATION

A. Experimental settings

We conduct the experiments with a custom built Java simulator. The main reason for doing so is efficiency. We do not need to analyze transmission time, real bandwidth, or

TABLE I: Topology Parameters

Number of	Topology I	Topology II
Nodes	47	107
Attackers	6	9
Edges	131	558
Node degree	1-8	2-17
Degree of root	4	6

packet drops issues. We only need to count the attack strength for different networks. The network topologies we use contain $47 - 107$ devices. Using NS3 or other similar simulators for this kind of simulation would take a long time. So, we build our own Java simulator to get the results quickly.

We generate two random topologies for some experiments. They are drawn in Fig. 4. The topologies are unit disk graphs. We first consider a rectangular region of 500×500 which is divided into multiple 50×50 blocks. A specific number of nodes are placed at random locations in each block. Links are added between two nodes if their distance is less than 70 units. The root is selected randomly at an edge. The locations of the attackers are also selected randomly. Detailed information about the topologies is listed in Table I.

We observe the attack strengths variation of CADAL by changing different parameters, including the number of neighbors an attacker can attack simultaneously (P), % of neighbors known to an attacker, and neighborhood radius (or transmission range). Three attacker ratios, 5%, 10%, and 15%, are used. We compare the performances of CADAL in both clustered and uniform attacker distributions. We evaluate CADAL against the random P-neighbor attack. All the numbers of the plots are averages and standard deviations (SD) of $1,000$ runs.

B. Simulation Result

At first we change the number of neighbors an attacker can attack simultaneously with three attacker ratios 5%, 10%, and 15%. There are 47 (or 107) nodes in Topology I (or Topology II). 2, 4, and 7 (or 5, 10, and 16) nodes in Topology I (or Topology II) are randomly selected as attackers. We vary P from 1 to 10 and observe the attack strengths. The result is shown in Fig. 5. The highest attack strength is with 15% attackers, the lowest is 5%, and 10% is in between for all P values. The higher the value of P, the more traffic can get to the root. When P is greater than 3 in Topology I and more than 4 in Topology II, the attack strength remains unchanged. Therefore, highly capable attackers are not always needed to launch powerful attacks.

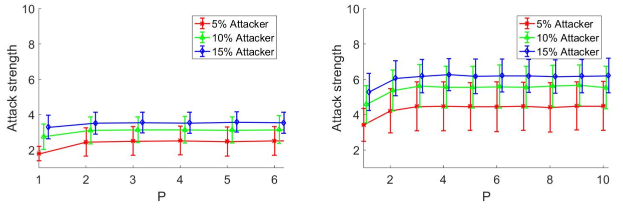
Fig. 6 shows how the attack strength changes with the location of the root. When the root is at one of the four corners of the region, the average degree of the root is the lowest. The number of disjoint paths from other nodes also decreases. Therefore, the attack strength is the lowest when the root is at any corner. When the root is at any edge of the region, the average degree of the root is slightly higher than that of the corner. The number of disjoint paths from other nodes also increases a bit. As a result, we can produce slightly stronger attack. The jump in the attack strength is observed when the root resides at the center of the region. This is because the number of disjoint paths from other nodes is the largest with the root at the center. For this experiment, we set the attacker ratio to 10%. For $P = 1$ the average attack strengths (and SD) are 1.49 (and 0.50), 1.59 (and 0.49), and 1.83 (and 0.36) when the root is at corner, edge, and center in Topology I. For $P = 6$ the average attack strengths (and SD) are 1.95 (and 1.03), 2.26 (and 0.85), and 3.80 (and 1.50) when the root is at corner, edge, and center in Topology I. For $P = 1$ the average attack strengths (and SD) are 2.24 (and 0.89), 2.94 (and 0.91), and 4.14 (and 0.76) when the root is at corner, edge, and center in Topology II. For $P = 10$ the average attack strengths (and SD) are 3.3 (and 2.11), 4.88 (and 1.63), and 10.21 (and 2.64) when the root is at corner, edge, and center in Topology II. So, the attack strength becomes 1-2 times in Topology I and 2-4 times in Topology II. The performance contrast is more dramatic in dense topology than Topology I.

Fig. 7 illustrates the influence of different distributions of attackers. To generate the clustered distribution, we calculate the desired number of attackers from the attacker ratio. Then, a node is picked up randomly and marked as an attacker with a probability of 10%. The probability becomes 90% when there is an attacker in that node's neighborhood. For this experiment, we set the attacker ratio to 10%. For $P = 1$, the average attack strengths (and SD) are 1.81 (and 0.38), and 1.52 (and 0.49) for uniform and clustered attacker distributions in Topology I. For $P = 6$, the average attack strengths (and SD) are 2.48 (and 0.81) and 2.25 (and 0.80) for uniform and clustered attacker distributions. For $P = 1$, the average attack strengths (and SD) are 3.45 (and 0.88), and 2.41 (and 0.87) for uniform and clustered attacker distributions in Topology II. For $P = 10$, the average attack strengths (and SD) are 4.5 (and 1.39) and 3.70 (and 1.47) for uniform and clustered attacker distributions. The attack becomes 1.07-1.19 times stronger in Topology I and 0.82-1.43 times stronger in Topology II.

Fig. 8 compares performances between the CADAL and random P-neighbor attack. In the random P-neighbor attack, an attacker randomly selects its P neighbors and attacks them. If the number of neighbors is less than P, then the attacker attacks all of its neighbors. We set the attacker ratio to 10%. The attack strengths of CADAL and random P-neighbor attack are the same when the value of P is very high. This is because the attacker attacks almost all the neighbors that are attacked by the CADAL attack. When P is low, the random P-neighbor attack does not select the most appropritate neighbors that maximize the attack strength. As a result, it cannot produce the strongest attack at lower P.

Fig. 9 shows the performance change with different neighborhood knowledge of an attacker in CADAL. We randomly select the desired percentage of neighbors for each attacker. The master receives reporting paths of a percentage of the neighbors. We vary the % of known neighbors from 10% to 100%. The value of P is 2 for these experiments. The attack strength linearly increases as the percentage of known neighbors increases. This trend is the same for all three attacker ratios and in both topologies.

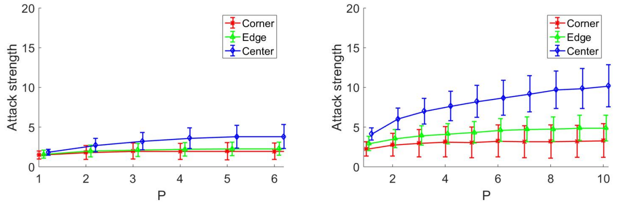
Fig. 10 shows the impacts of different neighborhood radius in CADAL. When the neighborhood radius increases, the



(a) Topology I.

(b) Topology II.

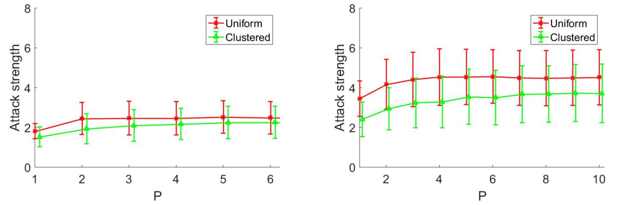
Fig. 5: Attack strength for different P .



(a) Topology I.

(b) Topology II.

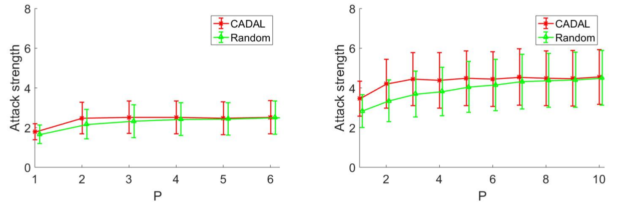
Fig. 6: Attack strength for different root location.



(a) Topology I.

(b) Topology II.

Fig. 7: Clustered and uniform attacker distribution.



(a) Topology I.

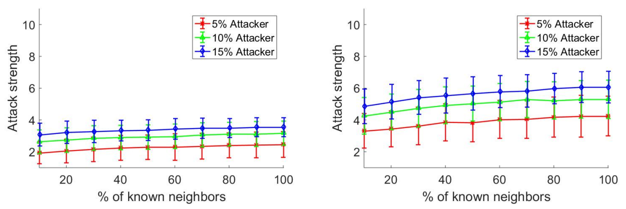
(b) Topology II

Fig. 8: CADAL vs. random P-neighbor attack.

degree of each node increases, and the number of the disjoint paths to the root increases. As a result, the larger neighborhood radius produces stronger attacks. P is 2 for these experiments. The neighborhood radius ranges from 50 to 150. The topology becomes disconnected (or over crowded) when the neighborhood radius is less than 50 or greater than 150. The attack strength increases almost linearly as the neighborhood radius gets larger in both topologies.

In summary, a dense LLN with higher node degree is more exposed to CADAL attacks. With a small number of attackers having P ranging from 1 to 3, a powerful attack can be launched in LLNs if the root is located at the center.

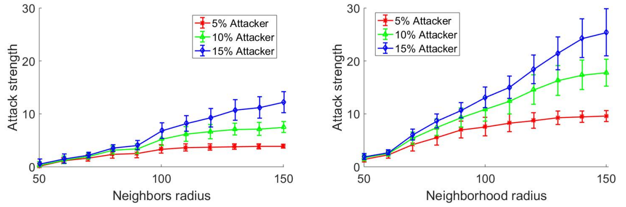
VI. Conclusion



(a) Topology I.

(b) Topology II.

Fig. 9: Attack strength for different neighborhood knowledge.



(a) Topology I.

(b) Topology II

Fig. 10: Attack strength for different neighborhood radius.

The DDoS attack is one of the most powerful and least costly attacks. The DDoS attacks in low power lossy networks (LLNs) are relatively new. Because of the limited battery, storage, and computation of devices, the routing protocol has to be simple. The simplicity of the routing protocol opens up opportunities for powerful DDoS attacks in LLNs. In this work, we present a powerful and capacity-aware DDoS attack framework called CADAL by targeting the root of an LLN. We show that by attacking a few neighbors of the attackers the strongest attack can be launched. CADAL outperforms random P-neighbors attack when attackers have limited attacking power. In the future, we will explore defense mechanisms and DDoS attack strategies for different protocol settings of LLNs.

Acknowledgments

This research was supported in part by NSF grants CNS 1824440, CNS 1828363, CNS 1757533, CNS 1618398, CNS 1651947, and CNS 1564128.

References

[1] US Code: Title 18,1030, "Fraud and related activity in connection with computers, government printing office," www.gpo.gov, 2014.

[2] R. Biswas and J. Wu, "Filter Assignment Policy Against Distributed Denial-of-Service Attack," in *IEEE 24th International Conference on Parallel and Distributed Systems*, Dec 2018.

[3] R. Biswas, J. Wu, W. Chang, and P. Ostovari, "Optimal Filter Assignment Policy Against Transit-link Distributed Denial-of-Service Attack," in *IEEE Global Communications Conference*, Dec 2019.

[4] 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," Tech. Rep., Mar 2012.

[5] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, May 2003.

[6] 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, Aug 2013.

[7] C. Pu, "Energy Depletion Attack Against Routing Protocol in the Internet of Things," in *2019 16th IEEE Annual Consumer Communications & Networking Conference*, Jan 2019.

[8] P. Kasinathan, C. Pastrone, M. A. Spirito, and M. Vinkovits, "Denial-of-Service detection in 6LoWPAN based Internet of Things," in *IEEE 9th international conference on wireless and mobile computing, networking and communications*, Oct 2013.

[9] A. Rghiout, A. Khannous, and M. Bouhorma, "Denial-of-service attacks on 6lowpan-RPL networks: Issues and practical solutions," *Journal of Advanced Computer Science & Technology*, vol. 3, no. 2, 2014.

[10] C. Pu and T. Song, "Hatchetman attack: A denial of service attack against routing in low power and lossy networks," in *5th IEEE International Conference on Cyber Security and Cloud Computing*, Jun 2018.