A Modified Version Number Attack Against RPL Based Networks

Mehdi ROUISSAT
Department of Electronics
Nour Bachir University Center. El-Bayadh, Algeria
STIC Laboratory, University Abou bekr Belkaid, Tlemcen, Algeria.
mehdi.m.rouissat@gmail.com

Mohammed BELKHEIR
LTTNS Laboratory, University Djilali Liabes, Sidi Bel Abbes, Algeria
LIMA Laboratory, Nour Bachir University Center. El-Bayadh, Algeria
belkheir_m@yahoo.fr

Hicham Sid Ahmed Belkhira
Department of Electronics
Nour Bachir University Center. El-Bayadh, Algeria
belkhira.hichem@gmail.com

***Abstract*—Routing Protocol for Low Power and Lossy Networks (RPL) has been proposed to efficiently manage the different functions of the network layer for wireless sensor networks when providing Internet connectivity. RPL has been designed for constrained devices and networks. Owing to their constrained power, memory, and processing, RPL-based networks are exposed to a wide range of security attacks that mainly include control message tampering. In this paper, we present a new modified Version Number attack, called Decreased Version Number Attack (DVNA), where the idea behind it is that the version number is decreased instead of increased as in the traditional version number attack. The aim of the attack is to initialize continually the trickle timer of the malicious node's neighbors, which leads to unnecessary exchange of control messages, therefore exhausts the resources of the nodes and reduce the network lifetime. The obtained results show an increase of up to 71% in the exchanged control messages and an increase of up to 47.77 % in energy consumption.**

Keywords—RPL, Security, DVNA, resources, version number

I. INTRODUCTION

Due to the constraint of Internet of Things (IoT) devices, routing is one of the most researched fields nowadays. Routing Protocol for Low Power and Lossy Networks (RPL), introduced by Internet Engineering Task Force (IETF) [1], is adequate and widely used with the power, memory, and processing constrained devices, where it became the standard for routing in many IoT networks, while providing effective routing service. Routing security was an integral part of RPL's design with several, but optional, security mechanisms available [1.2].

RPL is an IPv6 proactive distance-vector routing protocol for lossy wireless networks [1]. It arranges and organizes the nodes in the network into an inverted treelike topology known as Destination Oriented Directed Acyclic Graphs (DODAGs) [3]: a network of nodes connected without loops and where the traffic is rooted at a single destination called the root node or sink node [3]. The DODAG construction depends on the used Objective Function (OF), which defines essential configurations such as the used routing metrics, how the rank of the nodes is calculated (the rank represents its distance to the root node based on the routing metrics defined by the OF), and how to select the preferred parent in the DODAG [2].

Much work in both academia and industry has shown that RPL provides a promising routing solution for a wide range of network types [4], Nevertheless, security is particularly challenging in RPL based networks, not only because of the sensors limitation in terms of resources but because of their open, mobile and unattended deployment.

Some security mechanisms have been already considered in basic RPL. However, they can be useful and effective only against external attack, in other words they are unable to protect the network from internal attacks, like abnormal behavior of nodes that exhibit an egoistic conduct and control message tampering attacks. Therefore, RPL can be exposed to a wide variety of security attacks [5]

Some of the existing attacks aim to waste the resources of the motes, by making legitimate nodes performing unnecessary processing in order to exhaust their resource. [6]. Version Number attack is a well known attack against the resources, it leads to a repetitive recreation of the network by provoking global repairs, this is done by illegitimately increasing the version number by the malicious node.

In this paper we propose a modified Version Number Attack, in which the attacker node does not provoke global repairs, as the traditional version number attack acts, it provokes the initialization of the trickle timers of each mote listening to the attacker mote's DIO messages. In the next sections the impact of the proposed attack in terms of control over head and energy is introduced.

The rest of this paper is organized as follows. Section II introduces RPL protocol, section III presents the reference network that presents the base of our comparisons, in Section IV we present the proposed attack and highlight its impact. Finally, Section IV concludes the work and presents several directions for further work.

II. RPL PROTOCOL DESCRIPTION

RPL is a standardized routing protocol for the IoT. It was designed for power, memory, and processing constrained devices and networks. The design of RPL is a combination of multiple DODAGs networks, each of which considers many nodes connected to a DODAG sink. To create and maintain the DODAG graph and routing operation, RPL use several ICMPv6 control messages [5]:

A. Control messages

RPL Internet Control Message Protocol (ICMP) are used to created and maintained the DODAG, the most used messages are :

- DODAG Information Solicitation (DIS) : it is a broadcasted message used by nodes wishing to join a network to solicit DIO messages,
- DODAG Information Object (DIO) : are initially multicast by the DODAG root to create a new DAG. This message contains information to allow nodes discovering a RPL instance, compute their own ranks based on the objective functions (OF), select a preferred parent, and maintain the graph.. The rate at which these DIO messages are broadcasted is determined by the trickle algorithm [7]. Essentially, the longer a DODAG has been stable, the fewer the number of DIO transmissions.
- Destination Advertisement Object (DAO): is sent up the DODAG to form downwards routes to parents. [8,9]. They are sent by each node to spread and advertise routing tables with prefixes of their children as well as their addresses and prefixes to their parents [4].

B. *Version Number*

The version number is one of the important fields in DIO messages. It is incremented only by the root each time a rebuilding of the DODAG is necessary (also called global repair), to ensure that all nodes in the DODAG are up-to-date with the routing state [10], this value is propagated unchanged down the DODAG graph. An older value indicates that the node has not migrated to the new DODAG graph and cannot be used as a parent node [11].

However, there is no default mechanism to check the integrity of the version number by the nodes. A malicious node may modify this value and share it in its own DIO messages in order to harm the network [10].

Version number attack is an indirect attack against the resources of the network, where an attacker can makes other nodes generate an important overload for the network through illegitimately incrementing the version number associated with a topology, thereby forcing an unnecessary rebuild of the whole DODAG graph, which can negatively impact the energy consumption of the nodes, and therefore the network lifetime.

C. *Trickle timer algorithm*

As the generation and the forwarding of the different control messages incurs additional energy consumption, and in order to achieve optimal messages exchange between nodes, RPL uses trickle timer algorithm to assign the main interval for each node in the network through dynamically adjusting the transmission windows, which allow an exchange of information in a highly robust, energy efficient, simple, and scalable manner [12].

When the network is stable, the trickle timer interval is large. However, upon detection of inconsistencies, the trickle timer is reset, and more DIO messages are sent by the nodes.

The transmission window is controlled using an exponentially incremented interval. After the convergence of the network, the need for DIO messages decreases and thus the trickle timer algorithm is used to control the number of messages and the period of sending messages reaches its maximum.

The 'RPL_DIO_INTERVAL_MIN' value and the 'RPL_DIO_INTERVAL_DOUBLINGS' value define the DIO trickle timer, and are used to calculate respectively, the minimum and the maximum interval between DIO messages, [13], these values are defined by the sink and shared to all nodes through DIO messages.

The minimum interval of the trickle timer between DIO messages is given by:

$$I_{min} = 2^{n1} \tag{1}$$

Where "n1" presents the RPL DIO INTERVAL MIN, the default value in Contiki is 12. If n = 12, means:

$$I_{min} = 2^{12} = 4096 \text{ ms} \approx 4 \text{ s} \tag{2}$$

which is the initial and the minimum interval between two successive DIO sent by a given node.

On the other hand, the maximum interval between two successive DIO for a given node is used in the case where the network is stable, it is based on I_{min} and $I_{doubling}$.

$$I_{max} = I_{min} * 2^{n2} \tag{3}$$

Where n_2 is RPL DIO INTERVAL DOUBLINGS, the default value in Contiki is 8, based on that :

$$I_{max} = 4096 * 2^8 = 1048576 \text{ ms} \approx 17{,}5 \text{ min} \tag{4}$$

The configuration of the trickle timer through I_{min} and I_{max} is critical for the design of the network, in order to wisely use the energy of the nodes, especially in low power and lossy networks. Large used intervals improve energy efficiency but lead to low responsiveness while lower intervals improve responsiveness on at the cost of energy consumption and lifetime of the network. [14]

One of the well known issues that face LLNs is limited energy, the design of RPL protocol takes energy consumption seriously into consideration and proposes methods to minimize and limit its usage, where is addressed by the trickle timer [12], which mainly aims to minimize the unnecessary exchanged of control messages within the network. An attack that provokes the initialization of the trickle timer may affect the quality of the communication and the network lifetime.

III. Reference Network

We used RPL implementation in the Contiki OS to implement a given network, in a normal case, and under attack, using Cooja simulator. In our simulations, each node is created as a Z1 node.

Figure 1, shows the reference network in the normal cases where all the nodes are running a legitimate code, including node 20, which presents the malicious node in the next section.

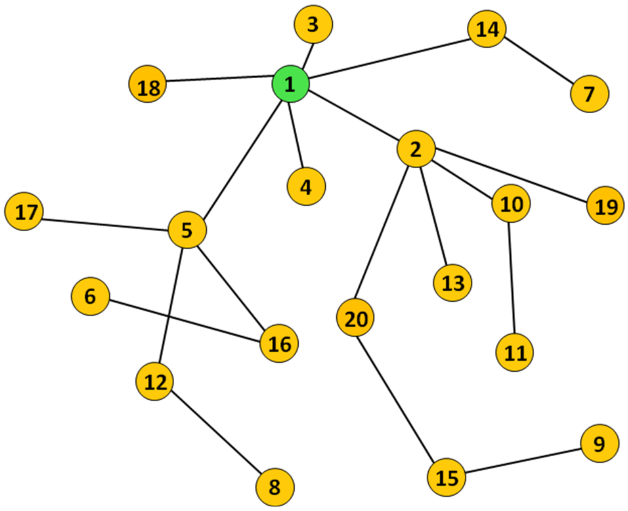

Fig. 1. Reference Network

It should be noticed that in this case the node 20 has two motes attached to it, which are node 15 and node 9.

A. Control overhead in the normal case

Table 1 shows the different control messages exchanged by all the nodes in 10 minutes of simulation, on the one hand, and by node 20 on the other hand. These gathered data is used as a benchmark to compare and show the behavior of the network in the case of an attack.

TABLE I. PROCESSED DIO AND DAO MESSAGES IN THE NORMAL CASE

	Sent / Generated		Transferred	Received	
Mote	**DIO**	**DAO**	**DAO**	**DIO**	**DAO**
All	244	150	138	818	291
20	12	6	15	57	17

B. Energy consumption in the normal case

Figure 2, shows the consumed energy by the different nodes. The nodes show different levels of consumption, based on their positions in the network, in terms of hops toward the root, number and distances to children and number and distances to neighbors. Where the nodes 18 and node 3 show the minimum consumption because they are close to the root and do not assure any forwarding task.

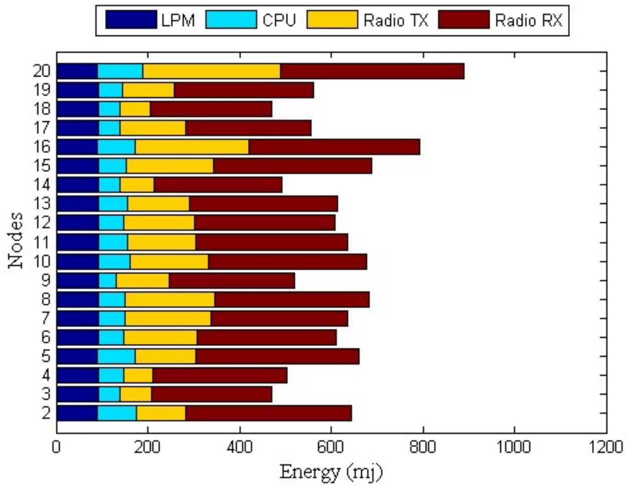

Fig. 2. Energy consumption by the different nodes in the normal case

IV. DECREASED VERSION NUMBER ATTACK

In the case of the proposed modified attack, DVNA "Decreased Version Number Attack", the malicious node decreases the version number of the topology by one in its DIO messages, while being up to date of the current version number of the network.

The goal behind this version number falsification is to create an inconsistency in the network and instability in the trickle timer of the malicious node's neighbors. The nodes initialize theirs timers upon detecting an old version in the network. The malicious node takes advantage of this feature and share a dynamic falsified value equal to the current topology version number minus 1. This dynamism in the chosen value is used to hide the malicious behavior of the attacker.

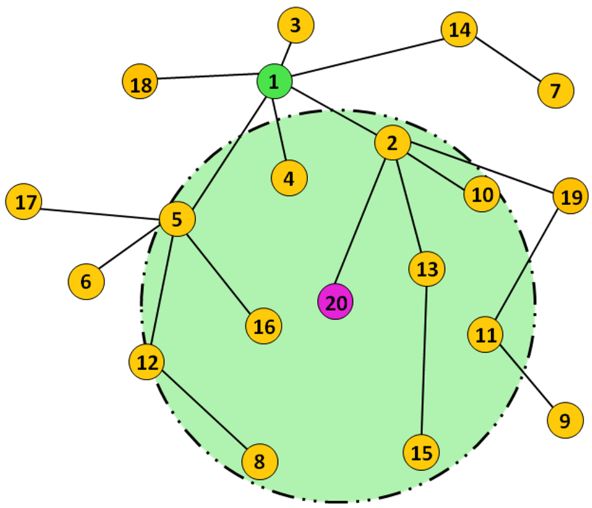

Fig. 3. Network Topology after the attack

Figure 3 illustrates the network topology after the attack. It can be seen that the node 20 does not has any children connected to it, this is due to the fact that nodes choose parents with newest version number, while node 20 is pretending to has an old version. The malicious node can be chosen as best parent and have children connected to it only in the case where it is the only available neighbor.

A. Control overhead in the case of the attack

The exchanged control messages before and after the attack are shown in table 2.

TABLE II. PROCESSED DIO AND DAO MESSAGES IN THE NORMAL CASE AND AFTER THE ATTACK

	Sent / Generated		Transferred	Received	
	DIO	**DAO**	**DAO**	**DIO**	**DAO**
	All the nodes				
normal	244	150	138	818	291
DVNA	377	256	317	1635	579
	Node 20				
normal	12	6	15	57	17
DVNA	12	17	1	153	4

The table 2 shows an increase in the control overhead after the attack, where the number of sent DIO messages jumped from 244 to 377, which presents an increase of 55%, and an average of 37 DIO message sent per minute. On the other hand the number of sent DAO messages jumped from 150 to 256, which presents an increase of 71%.

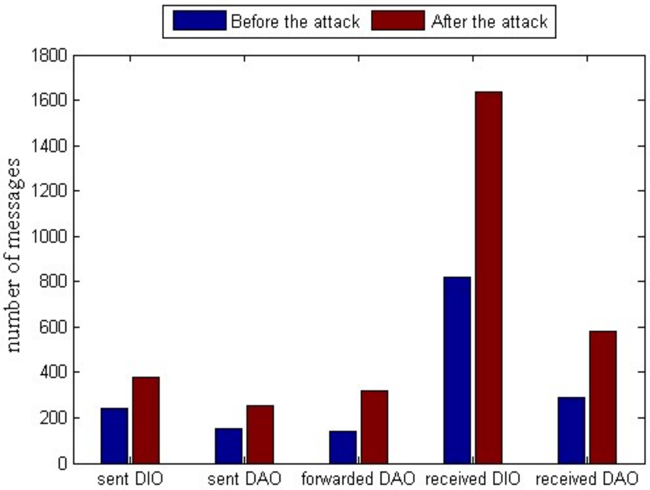

Fig. 4. Number of exchanged messages before and after the attack

The number of forwarded DAO messages shows the most important increase from 138 to 317 messages which presents an increase of 130 %. The results of table 2 are illustrated in figure 4, where it can show a noticeable increase in the received DIO messages which increase consequently the consumed energy, as is shown in the next section.

B. Energy consumption in the case of the attack

The consumed energy by the different nodes in terms of LPM, CPU, TX and RX are shown in figure 5.It can be seen that the conducted attack leads to global increase in the consumed energy.

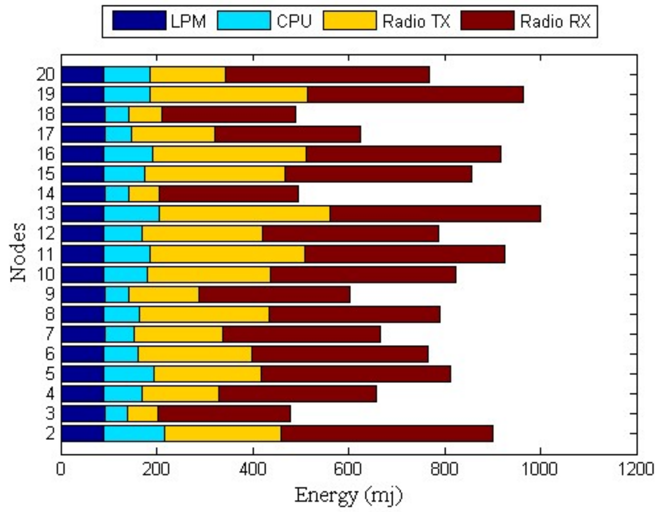

Fig. 5. Energy consumption by the different nodes in the case of the attack

Figure 6 details the differences in energy consumption in the two cases. The figure shows that the node 20 is the only node that shows a decrease in the consumed energy, this is due to the fact that its trickle timer had net been initialized since it is not receiving an old version number, moreover it is not assuring any forwarding task as it has no nodes connected to it, unlike the normal case.

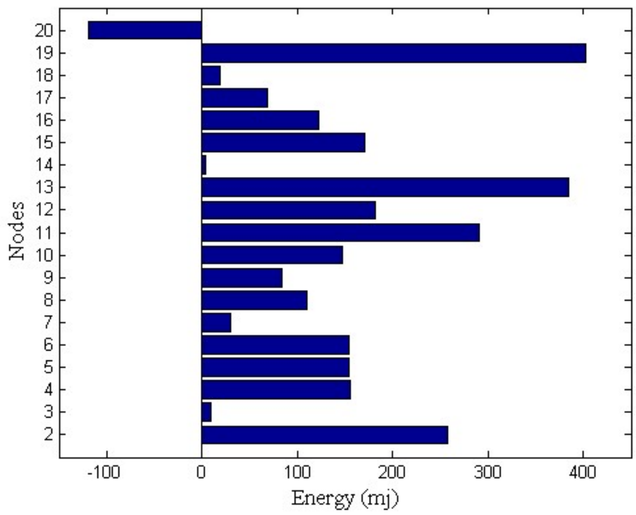

Fig. 6. Difference in energy consumption before and after the attack

Figure 6 also illustrates that the remaining nodes show different behaviors, where the node 19 is the most affected node where it shows an increase by 400 mj, which presents an increase by 72 %. This increase for node 19 is mainly because the transmitting mode, where the consumed energy jumped from 114 mj to 328 mj which present an increase by 188 %. This logic can be applied to all the remaining nodes.

In order to clarify this latter point, figure 7 shows the total consumed energy by all the nodes in terms of LPM, CPU, TX and RX before and after the attack.

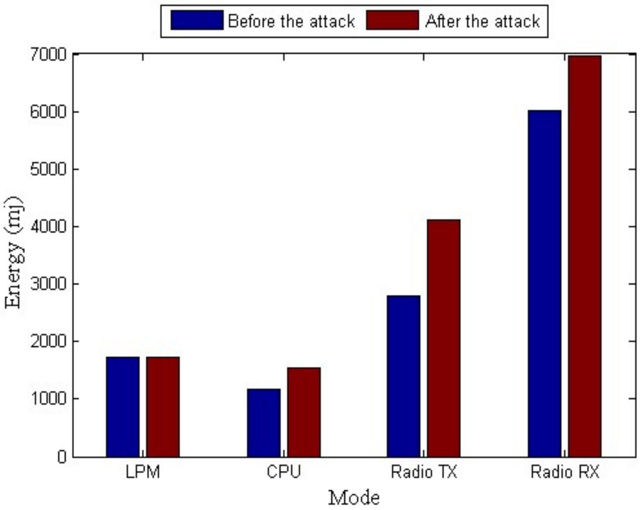

Fig. 7. Total consumed energy by all the nodes before and after the attack

Figure 7 aims to show the mode that demonstrates the most noticeable increase in energy consumption. The change in the total consumed energy by all the nodes in terms of LPM, CPU, TX and RX are respectively -0.63 %, 30.93 %, 47.77 % and 15.68 %. These obtained results confirm the impact of the attack in the energy consumed in the transmitting mode, where an increase of 47.77 % is shown, which allows us to describe the attack as successful.

V. CONCLUSION

In this paper, we presented a modified version number attack, called DVNA, which aims to exhaust the network resources. The basic idea of the attack is to force the victim nodes to initialize their trickle timers, which lead to an unnecessary generated, forwarded and listen to traffic.

According to the obtained results, the attack can be described as successful, where it reaches the goal behind it. The network under attack shows an important increase in the control overhead, by about 71 %, on the other hand the total consumed energy shows an increase of 74.77%. This destructive impact limits the nodes lifetime and consequently the network's.

In the proposed attack, the malicious node follows the default increase in its trickle timer, that starts with a minimum and stops at a maximum once the network is stable. Using small fixed interval may be more harmful and damaging to the network, this will be the subject of our future work, beside the focus on how to detect and mitigate such attack.

REFERENCES.

[1] T. Winter *et al.*, "RPL: IPv6 Routing Protocol for Low Power and Lossy Networks," Internet Requests for Comments, RFC Editor, RFC 6550, March 2012, https://ietf.org/doc/*rfc6550*/

[2] A Raoof, A Matrawy, CH Lung, "Secure Routing in IoT: Evaluation of RPL's Secure Mode under Attacks", 2019 IEEE Global Communications Conference (GLOBECOM), 1-6.

[3] J. Granjal, E. Monteiro, and J. Silva "Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues," *IEEE Comm. Surveys and Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[4] S. Yasser Hashemi , F. Shams Aliee, "Dynamic and comprehensive trust model for IoT and its integration into RPL", The Journal of SupercomputingVolume 75Issue 7July 2019 pp 3555–3584

[5] F. Ahmed, Y. Ko, "Mitigation of black hole attacks in Routing Protocol for Low Power and Lossy Networks", Security and Communication Networks, vol. 9, pp. 5143–5154, 2016

[6] S. Mohammad Hossein Mirshahjafari & B. Shahgholi Ghahfarokhi "Sinkhole+CloneID: A hybrid attack on RPL performance and detection method", Information Security Journal: A Global Perspective, 28:4-5, (2019) 107-119,

[7] .Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, J., Alexander, R.: RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. IETF RFC 6550 (March 2012)

[8] P.A Levis, N. Patel, D. Culler, S. Shenker, "Trickle: A Self Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks". In: 1st Symposium on Networked Systems Design and Implementation (NSDI), San Francisco, CA (March 2004)

[9] M. A Boudouaia., A. Ali-Pacha, A. Abouaissa, & P. Lorenz." *Security Against Rank Attack in RPL Protoco"l.2020. IEEE Network, 34(4), 133–139.*

[10] A. Mayzaud, A. Sehgal, R. Badonnel, I. Chrisment, J. Schönwälder, "A Study of RPL DODAG Version Attacks". *8th IFIP International Conference on Autonomous Infrastructure, Management and Security (AIMS)*, Jun 2014, Brno, Czech Republic. pp.92-104

[11] A. Mayzaud, R. Badonnel, I. Chrisment, " A Taxonomy of Attacks in RPL-based Internet of Things", International Journal of Network Security, ACEEE a Division of Engineers Network, 2016, 18 (3), pp.459 - 473.

[12] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, "The Trickle Algorithm," *Internet Engineering Task Force, RFC 6206*, 2011

[13] H. Lamaaz., N. Benamar, "RPL Enhancement Based FL-Trickle: A Novel Flexible Trickle Algorithm for Low Power and Lossy Networks". *Wireless Pers Commun* 110**,** 1403–1428 (2020).

[14] H. Kharrufa, H. Al-Kashoash, A. Kemp*. "RPL-Based Routing Protocols in IoT Applications: A Review." IEEE Sensors Journal 19 (2019): 5952-5967.*