2011 Eighth IEEE International Conference on Mobile Ad-Hoc and Sensor Systems

VeRA - Version Number and Rank Authentication in RPL

Amit Dvir, Támas Holczer, Levente Buttyan
Laboratory of Cryptography and System Security (CrySyS)
Budapest University of Technology and Economics, Hungary
http://www.crysys.hu/

Abstract—**Designing a routing protocol for large low-power and lossy networks (LLNs), consisting of thousands of constrained nodes and unreliable links, presents new challenges. The IPv6 Routing Protocol for Low-power and Lossy Networks (RPL), have been developed by the IETF ROLL Working Group as a preferred routing protocol to provide IPv6 routing functionality in LLNs. RPL provides path diversity by building and maintaining directed acyclic graphs (DAG) rooted at one (or more) gateway. However, an adversary that impersonates a gateway or has compromised one of the nodes close to the gateway can divert a large part of network traffic forward itself and/or exhaust the nodes' batteries. Therefore in RPL, special security care must be taken when the Destination Oriented Directed Acyclic Graph (DODAG) root is updating the Version Number by which reconstruction of the routing topology can be initiated. The same care also must be taken to prevent an internal attacker (compromised DODAG node) to publish decreased Rank value, which causes a large part of the DODAG to connect to the DODAG root via the attacker and give it the ability to eavesdrop a large part of the network traffic forward itself. Unfortunately, the currently available security services in RPL will not protect against a compromised internal node that can construct and disseminate fake messages. In this paper, a new security service is described that prevents any misbehaving node from illegitimately increasing the Version Number and compromise illegitimate decreased Rank values.**

I. Introduction

Low power and Lossy Networks (LLNs) are a class of wireless and wired networks, such as personal area networks (PANs), wireless sensor networks (WSNs), and low-power Power Line Communication (PLC) networks. In all of these LLNs, the nodes typically operate with constraints on processing power, memory, and battery. With the increasing demand for LLN services, and recognizing the need for LLNs to be connected to IP networks, the IETF [1] Routing Over Low power and Lossy networks (ROLL) Working Group [2] was formed to design a routing solution for LLNs. Existing routing protocols, such as Open Shortest Path First (OSPF) [3], Intermediate system to intermediate system (IS-IS) [4], Ad hoc On-Demand Distance Vector (AODV) Routing [5], and Optimized Link State Routing Protocol (OLSR) [6], have been extensively evaluated by the ROLL Working Group and have been found to not satisfy, in their current form, all of the specific routing requirements of LLNs [7]–[10]. Therefore, the ROLL Working Group is currently designing the Routing Protocol for Low power and Lossy Networks (RPL) [11] to meet the core requirements specified in [7]–[10].

An adversary can intercept, forge, modify, inject, replay, and create messages (data or control) in order to interfere with the operation of entire network and exhaust nodes' batteries. The security services that RPL currently provides [11], [12] natively are sufficient to protect the network against such an attack if only an external attacker is assumed. However, native RPL security services will not protect against a compromised internal node. Therefore, we consider the case where the attacker is a compromised node, where the source of the compromise can be logical [13] or physical [14].

An internal adversary can attack RPL in many different ways. However, some of these attacks have only a minor influence, while others may have a major influence in the network. A consequence of an attack that will lead to reconstructing the entire DAG and exhaust the nodes' batteries, or eavesdropping a large part of the network traffic can have major or even critical influence. One way for an internal adversary to achieve these goals is to change/modify the Version Number of the DODAG, a sequential counter that is incremented by the DODAG root to form a new Version of a DODAG, or to change/modify the DODAG node Rank value, representation of the location of that node within a DODAG [11]. Therefore, in this paper we present a security scheme to prevent the following attacks: (i) an internal attacker impersonating a DODAG root and illegitimately increasing the Version Number, and (ii) an internal attacker publishing an illegitimately decreased Rank value.

The organization of the paper is the following: first a survey of related works is provided in Section II. In Section III, we briefly summarize the operation of the RPL protocol. The paper's main contribution our proposed security mechanism is presented in Section IV. An overhead analysis is presented in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

The scope of the current RPL security services [11] is the link, authenticity of the messages sent by the DODAG root relies on the trustworthiness of all intermediate nodes and the fact that none of the keys are compromised. Tsao et al. [12] build a security framework for LLN upon previous routing security protocols, and adapts ideas to fulfill the LLN constraints. However, because of to the lack of physical protection, nodes can be compromised (including their keys). Moreover, the security services in RPL [11], [12] will not

978-0-7695-4469-4/11 $26.00 © 2011 IEEE
DOI 10.1109/MASS.2011.76

protect against a compromised internal node that can construct and disseminate fake messages. For further information about analysis different aspects of RPL see [15]–[23].

Extensive work has been performed in the area of securing routing protocols in ad hoc networks. An overview of the various approaches for securing routing in ad hoc networks can be found in [24], [25]. In this paper, we use similar ideas such as hash chain [26]–[28] but in a new context provided by RPL. Note that for standards, it is important to be based on stable mechanisms.

III. The RPL Protocol

In LLNs, the links may be unstable and the participating nodes may have power, computational, and storage constraints. Usually, there is one or a few special nodes called base stations/gateways. A base station/gateway can be responsible for collecting the data measured by the nodes and to control these nodes. Therefore, the destination or the source of most of the flows is the base station/gateway. In RPL the base station/gateway is called DODAG root.

RPL is based on the principles of distance vector routing and on the notion of DAG [11]. There are upward routes directed from DODAG nodes to the DODAG root and there are downward routes directed from the DODAG root to the DODAG nodes. The upward routes and downward routes are established independently. The support for downward routes is optional; thus, a network can operate according to RPL even if only DODAG nodes can send messages to the DODAG root. RPL [11] also supports the communication between two regular DODAG nodes (p2p) by combining upward and downward routes or using reactive discovery [29].

The construction of upward routes is performed through the dissemination of DODAG Information Object (DIO) messages initiated by the DODAG root. The network forms a directed acyclic graph (DAG) such that each node based on the DIO messages received from its neighbors decides which neighbors become its DODAG parents. There can be multiple neighbors becoming DODAG parents, but the DODAG node selects one which is called preferred parent. When a DODAG node is about to forward a message towards the DODAG root, it first tries to send the message to the preferred parent. If the transmission is unsuccessful, then it tries to forward the message to any of the non-preferred DODAG parents, one after the other.

Once a node has connected to the network (i.e., set its DODAG parents), it multicasts an updated DIO message to its neighbors. The nodes that are not yet connected to the DODAG may consider this DODAG node as a potential DODAG parent, while other DODAG nodes, already connected staying at the same level in the graph, consider the DODAG node as sibling, nodes with the same level in the DAG as the node being considered. Sibling connections are used when none of the parents can forward the message to the DODAG root. The DIO messages, in addition to DODAGID (the identifier of a DODAG root), contain information for the nodes to decide which neighbors become DODAG parents. There is also information with which the nodes can decide at what level they are in the graph (i.e., they can decide which nodes are their siblings). For the former, there is a metric container [30] and for the latter, a so-called Rank is used.

The metric container [30] is an optional part in DIO messages and supports the description of a large variety of routing metrics and constraints. The metrics and constraints can refer to a node (e.g., main powered versus battery operated) or to a link (e.g., throughput, latency, reliability). The constraints are used to filter out some nodes or links that do not satisfy some specific requirements. Each routing metric can be additive, or it can report a maximum or a minimum value. The metric container may contain multiple metrics and constraints. Which metrics are included in the DIO messages are described by a so-called Objective Function (OF) [11]. The OF is not fully defined in RPL; thus, RPL can be adapted to a large variety of networks by defining the right OF. The OF describes what routing metrics should be included in the DIO messages; furthermore, an OF also describes how the nodes have to consider the routing information when they choose DODAG parents and preferred parent. The OF also determines how the nodes set their Rank. An example OF can be found in [31].

The Rank is a value that helps the nodes to determine at what level they are in the directed acyclic graph. The lower the Rank is, the closer (in a sense of the hop numbers) the node is to the DODAG root. Each node must set the Rank such that the Rank of all the selected DODAG parents are lower. The siblings of a node are those neighboring nodes whose Rank value is equal to the Rank value of the node. MinHopRankIncrease (MHRI) is the minimum increase in Rank value between a node and any of its DODAG parents. RPL allows the nodes to increase their Rank in order to become distant from the DODAG root. This can be beneficial for the node, because the node has to forward fewer messages, the closer a node is placed to the DODAG root the more messages it has to forward. In that case the DODAG parents and the siblings of the node may change, and also the nodes that set this node as a parent have to change the relation, otherwise a loop may be formed. RPL has some mechanisms to detect loops that may arise because of other reasons than described above. If the network detects a loop that can be difficult to repair, it may reconstruct the whole graph. The reconstruction, global repair, can be initiated by the DODAG root by sending DIO messages with an increased Version Number.

Version number is an element of each DIO message and related to the network. The Version Number is monotonically incremented by the root each time the DODAG root decides to form a new Version of the DODAG in order to revalidate the integrity and allow global repairs to occur. The Version Number is propagated unchanged Down the DODAG as node join the new DODAG. The Version Number value is globally significant in a DODAG and indicates the Version of the DODAG that a node is operating in. An older (lesser) value indicates that the node has not migrated to the new DODAG and cannot be used as a parent once the receiving node has

TABLE I
DEFINITIONS

Notation	Definition
i	index of the Version Number hash chain
j	index of the Rank hash chain
r	Random number
x_i	Random number
$n+1$	Version Number hash chain length
$l+1$	Rank hash chain length
$h()$	A cryptographic one-way hash function
V_0	Root of the Version Number hash chain
R_{mrh}	Max Rank hash value associated with the ith Version Number hash chain element
V_i	The ith Version Number hash chain value
$R_{i,j}$	The jth Rank hash value associated with the ith Version Number hash chain element
R_{sender}	Rank element value
$Rank_{root}$	Rank value of the DODAG root
$Rank_{sender}$	Rank value of the parent
M	Message
$Init\_VN$	Initial value of the Version Number
VN	Current Version Number value
IP	Integrity protection
MAC	Message authentication code
$MHRI$	MinHopRankIncrease [11]
OF	Objective Function

migrated to the newer DODAG [11].

When a node is initialized [11] (not yet connected), the node either decides to stay silent, waiting to receive DIO messages from a DODAG (as described above) and not send any multicast DIO messages until it has joined a DODAG, or decides to send one or more DODAG Information Solicitation (DIS) messages as an initial probe for nearby DODAGs.

IV. VeRA - Version Number and Rank Authentication

A. Overview of Our Contribution

RPL allows the Version Number to be increased regularly or occasionally. Moreover, reconstruction of the routing topology can be initiated by sending a DIO message with an increased Version Number. Therefore, preventing any misbehaving DODAG node from impersonating a DODAG root and increasing the Version Number is crucial. Note that how often the Version Number is increased is out of scope the RPL draft [11] and this paper (application dependent).

By publishing a high Rank value, an attacker can move deeper in the DODAG in order to increase the size of the parent set or improve some other metric. However, the most effective Rank attack is by decreasing the Rank. By publishing a low Rank value, a large part of the DODAG will connect to the DODAG root via the attacker, and give it the ability to eavesdrop and manipulate a large part of the network traffic.

The Version Number and Rank Authentication (VeRA) security scheme presented in this paper prevents: a) misbehaving (compromised) nodes from impersonating a DODAG root and sending a DIO message with an illegitimate increased Version Number; b) misbehaving (compromised) nodes from publishing an illegitimate decreased Rank.

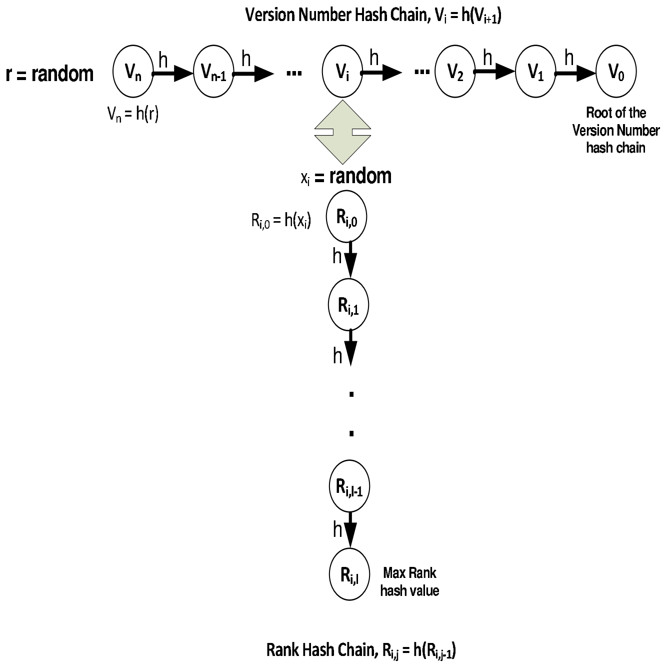

Fig. 1. The Hash Chains.

B. Protocol Description

Upon initialization, a DODAG root generates a random number r and calculates a hash chain [32], named as Version Number hash chain, of size $n+1$: $V_n, V_{n-1}..., V_1, V_0$ where $V_n = h(r), V_i = h(V_{i+1})$, with the random number. For each element V_i of the Version Number hash chain the DODAG root generates a new random number x_i and calculates a new hash chain, named as Rank hash chain, of size $l+1$[1]: $R_{i,0}, R_{i,1}..., R_{i,l-1}, R_{i,l}$ where $R_{i,0} = h(x_i), R_{i,j} = h(R_{i,j-1})$, with the new random number. Figure 1 illustrates the hash chains, while Table I summarizes the notations and definitions.

Then the DODAG root reveals the root of the Version Number hash chain V_0, computes $MAC_{V_1}(R_{mrh})$ a message authentication code (MAC) value over the Max Rank hash (mrh) value $R_{mrh} = R_{1,l}$ of the next element V_1 with the next Version Number hash chain element (V_1) as the key. Using a digital signature $\{V_0, MAC_{V_1}(R_{mrh}\}_{sign}$ the DODAG root binds these values to a particular DODAG and DODAGID. Finally, the DODAG root multicasts a DIO message with V_0, $MAC_{V_1}(R_{mrh})$, the initial value $Init_{VN}$ of the Version Number, and the signature ($M\#1$, in Figure 2); the DODAG root can resend the signed DIO message until the first Version Number update occurs.

Upon receiving the signed DIO message, each intermediate

[1] In theory, l should be 65535 to have a different value for each possible Rank value (Rank is 16 bits long [11]). In practice, 256 is large enough as for most of the operations DAGRank [11] is used (the DAG Rank of a node is the upper 8 bits of the Rank), which is 8 bits long. If DAGRank is used when defining the hash chain, then all occurrences of Rank must be substituted by DAGRank in the sequel.

node verifies the authentication data and in case of success saves the signature, the root V_0 of the Version Number hash chain, the MAC value, and the initial value $Init_{VN}$ of the Version Number. When the trickle timer [33] expires, it multicasts to all neighbors the DIO message ($M\#2$) according to [11].

Upon Version Number update, the DODAG root sends a DIO message ($M\#3$) with the following parameters: next Version Number value VN as described in [11], next Version Number hash chain element V_i, $MAC_{V_{i+1}}(R_{mrh})$ a message authentication code (MAC) value over the Max Rank hash value $R_{mrh} = R_{i+1,l}$ with the next Version Number hash chain element V_{i+1} as the key, and a Rank element value $R_{sender} = h^{Rank_{root}}(x_i)$, where $Rank_{root}$ is the new Rank value[2] of the DODAG root.

Upon receiving a DIO message with a new Version Number or new Rank value, an intermediate node can easily verify the message because, if the Version Number is increased by the DODAG root, $h^i(V_i)$ must be equal to V_0. Upon success, the intermediate node saves the current Version Number value (only if the Version Number increased). Moreover, using the revealed key V_i, the MAC value from the previous update $MAC_{V_i}(R_{mrh} = R_{i,l})$, and the Rank element R_{sender}, the intermediate node can verify whether the Rank value of its parent is monotonically increasing as follows:

- It generates $R_{check} = h^{l-Rank_{sender}}(R_{sender})$, where $Rank_{sender}$ is the Rank value of the parent.
- It checks if $MAC_i = MAC_{V_i}(R_{mrh})$, from the previous update, is equal to $MAC_{check} = MAC_{V_i}(R_{check})$.
- If so, intermediate node v can conclude that the Rank is monotonically increasing and:
  - It calculates its Rank value regarding its Objective Function, $Rank_v$.
  - It calculates $\delta = Rank_v - Rank_{sender}$, the difference between the node Rank value and its parent Rank value. Node v has the parent Rank from the DIO message.
  - It calculates R_{sender}= $h^{\delta}(R_{sender})$.
  - When the trickle timer [33] expires, it multicasts to all neighbors the DIO message ($M\#4$) according to [11] with the new R_{sender} value.

If an attacker wants to increase the Version Number (or decrease the Rank), then it has to compute a pre-image of the last revealed hash chain element of the Version Number chain (or compute a pre-image of the last R_{sender}). However, computing the next element V_{i+1} or previous R_{i-1} knowing V_i or R_i is hard when x, r is not known and $h()$ is a cryptographic one-way hash function.

In the case when a new node wants to join the DODAG, any DODAG node receiving a unicast DIS message ($M\#5$) from the new node must reply with a DIO message ($M\#6$) containing the current Version Number value (VN), the initial Version Number value ($Init\_VN$), the root of the Version Number hash chain (V_0), the current Version Number hash chain element (V_i), saved signature (IP), and the last MAC value. Note that all these data is stored by every node in the network according to our scheme.

TABLE II
ESTIMATED TIME OF THE BUILDING BLOCKS ON A MICAz

Algorithm	Computing/ Generating Time [msec]	Verification [msec]
SHA-1 [35]	1.4	–
HMAC-SHA-1 [35]	5.6	5.6
RSA-1024 [36]	12040	470
TinyECC-secp160r1 [37]	1900	2400

V. EVALUATION

From the LLN viewpoint, the overhead of cryptography algorithms and their efficiency is crucial. Therefore, in this section we will evaluate the overhead of our new security scheme in a wireless sensor network assuming MICAz motes [34]. As the first step we will define the building blocks of the scheme, then based on [35]–[37] estimate the time for each building block and finally, calculate for each node the time overhead of the security scheme. Note that for simplification we split the overhead computation of VeRA to two parts, the Version Number Authentication (VNA) and Rank Authentication (RA).

The building blocks of our security scheme presented above are the following:

- Hash function, e.g., Secure Hash Algorithm (SHA) [38].
- MAC function, e.g., Keyed-Hashing for Message Authentication (HMAC) [39].
- Digital Signature, e.g., RSA [40] or Elliptic Curve DSA [41].

We can easily conclude that one HMAC operation [39] is equivalent, in the worst case, to 4 hash operations, 2 main hash operations where each of them have to hash 2 blocks. Table II presents the estimated time for each building block running on MICAz.

For Version Number Authentication (VNA) part, the DODAG root has to generate a hash chain with chain length $n+1$ and to sign the first DIO message, while each intermediate node has to verify the signature once and, also verify the hash element per Version Number updated. This requires one signing operation ECC_G and $(n+1)\cdot SHA$ operations in the DODAG root while in the DODAG node, one verification operation ECC_V and $\frac{(2\cdot(n+1)-I)\cdot I}{2}\cdot SHA$ hash operations for all the elements is required, where I is the total number of Version Number updates.

For the Rank Authentication (RA) part, the DODAG root computes a MAC value and generates a new hash chain per Version Number update while, each intermediate node needs to verify the message and check the hash value per Version Number update. This requires one MAC operation (MAC) and $(l+1)\cdot SHA$ operations per Version Number update in

[2] Note that, in any case we are using the Rank value as the indicator to the number of times to hash a value we divided the Rank value by MinHopRankIncrease in order to decrease the size of Rank hash chain, $\frac{Rank}{MinHopRankIncrease}$

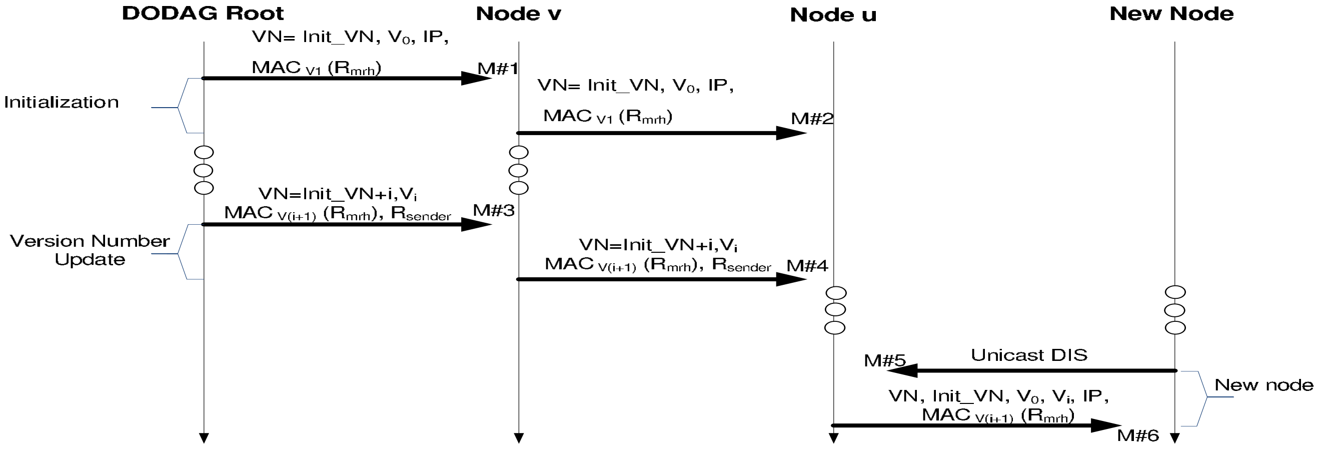

Fig. 2. Sequence Diagram of the Security Scheme.

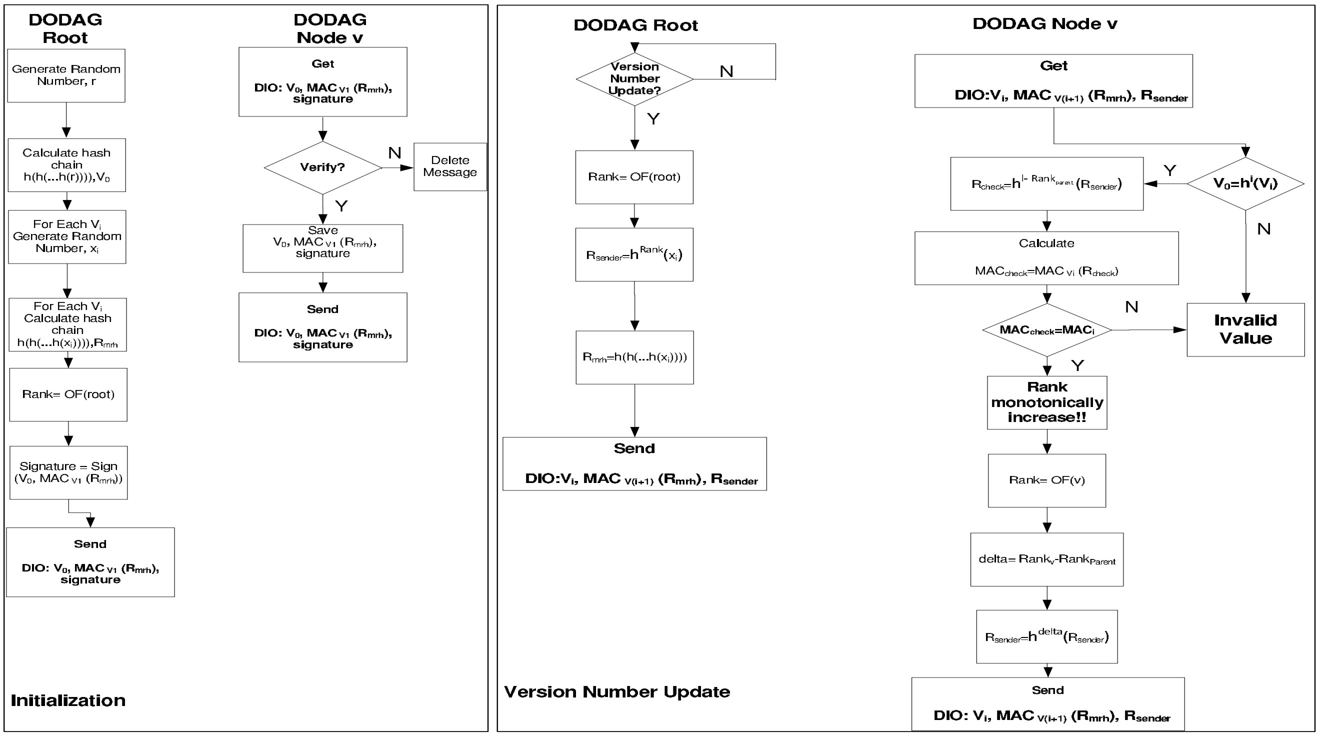
DODAG Root
DODAG Node v
Generate Random Number, r
Calculate hash chain h(h(...h(r)))),V_0
For Each V_i Generate Random Number, x_i
For Each V_i Calculate hash chain h(h(...h(x_i)))),R_{mrh}
Rank= OF(root)
Signature = Sign (V_0, MAC $_{V1}$ (R_{mrh}))
Send
DIO: V_0, MAC $_{V1}$ (R_{mrh}), signature
Get
DIO: V_0, MAC $_{V1}$ (R_{mrh}), signature
Verify?
N
Delete Message
Y
Save
V_0, MAC $_{V1}$ (R_{mrh}), signature
Send
DIO: V_0, MAC $_{V1}$ (R_{mrh}), signature
Initialization
DODAG Root
DODAG Node v
Version Number Update?
N
Y
Rank= OF(root)
R_{sender}=h^{Rank}(x_i)
R_{mrh}=h(h(...h(x_i))))
Send
DIO:V_i, MAC $_{V(i+1)}$ (R_{mrh}), R_{sender}
Get
DIO:V_i, MAC $_{V(i+1)}$ (R_{mrh}), R_{sender}
V_0=h^i(V_i)
Y
N
R_{check}=$h^{l-Rank_{parent}}$(R_{sender})
Calculate
MAC_{check}=MAC $_{Vi}$ (R_{check})
MAC_{check}=MAC_i
N
Invalid Value
Y
Rank monotonically increase!!
Rank= OF(v)
delta= $Rank_v$-$Rank_{Parent}$
R_{sender}=h^{delta}(R_{sender})
Send
DIO: V_i, MAC $_{V(i+1)}$ (R_{mrh}), R_{sender}
Version Number Update

Fig. 3. Flow Chart of the Security Scheme, MAC_i is the MAC function value from the previous Version Number update.

TABLE III
Estimated Time Equations of the Security Scheme

	Root	Intermediate
VNA	$ECC_G + (n+1)\cdot SHA$	$ECC_V + \frac{(2\cdot(n+1)-I)\cdot I}{2}\cdot SHA$
RA	$I\cdot(MAC+(l+1)\cdot SHA)$	$I\cdot(MAC+((l+1)-MHRI\cdot level)\cdot SHA)$

the DODAG root while in the intermediate one MAC operation (MAC) and $((l+1) - MHRI \cdot level) \cdot SHA$ operations per Version Number update is required, $level$ is the hop count from the DODAG root in the DAG.

Note that, the evaluation time of each intermediate node in the VNA part it equal while in RA it depend on the $level$ of the node. Table III summarizes the general estimate time overhead of our new scheme while Table IV gives an example of a LLN network with $n, l = 99$, $I = 50$, $level = 1$, $MHRI = 1$ ($level = 1$, one of the nodes next to the DODAG

TABLE IV
ESTIMATED TIME EXAMPLE

	Root	Intermediate
VNA	2040	7650
RA	7280	7210

root).

VI. CONCLUSION

In this paper, we identified illegitimate Version Number increases and Rank value decreases as two powerful attacks against RPL which eavesdropping the entire LLN network traffic and exhaust the nodes' batteries. Moreover, we proposed solutions that prevents both of the attacks based on stable mechanisms. Using evaluation we showed that the time overhead of our security scheme based is sufficient small. As future work, we will deploy our new security scheme in RPL implementation and in real wireless sensor deployment.

ACKNOWLEDGEMENT

The work described in this paper is based on results of the WSAN4CIP project (`http://www.wsan4cip.eu`), which receives research funding from the European Community's 7th Framework Programme. Apart from this, the European Commission has no responsibility for the content of this paper. Amit Dvir has also been supported by the Marie Curie Mobility Grant, OTKA-HUMAN-MB08-B 81654 and Levente Buttyan has been supported by the Hungarian Academy of Sciences through the Bolyai Janos Research Fellowship. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

REFERENCES

[1] IETF, "The internet engineering task force," http://www.ietf.org/.

[2] IETF-ROLL, "Routing Over Low power and Lossy networks (ROLL) - Working Group," http://datatracker.ietf.org/wg/roll/.

[3] "Multicast extensions to OSPF," SRI Networks Information, RFC 1584, March 1994.

[4] J. Parker, "Recommendations for interoperable networks using intermediate system to intermediate system," Internet RFC 3719, Feb. 2004.

[5] T.Pusateri, "Ad hoc on demand distance vector (AODV) routing," Internet draft, draft-ietf-manet-aodv-04.txt, Oct. 1999.

[6] T. Clausen and P. Jacquet, "Optimized link state routing protocol," RFC 3626, 2003.

[7] A. Brandt, J. Buron, and G. Porcu, "Home Automation Routing Requirements in Low-Power and Lossy Networks," in *RFC 5826*, Apr. 2010.

[8] J. Martocci, P. D. Mil, N. Riou, and W. Vermeylen, "Building Automation Routing Requirements in Low-Power and Lossy Networks," in *RFC 5867*, June. 2010.

[9] M. Dohler, T. Watteyne, T. Winter, and D. Barthel, "Routing Requirements for Urban Low-Power and Lossy Networks," in *RFC 5548*, May. 2009.

[10] K. Pister, P. Thubert, S. Dwars, and T. Phinney, "Industrial Routing Requirements in Low-Power and Lossy Networks," in *RFC 5673*, Oct. 2009.

[11] T. Winter, P. Thubert, A. Brandt, T. Clausen, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, and J. Vasseur, "RPL: IPv6 Routing Protocol for Low power and Lossy Networks," in *draft-ietf-roll-rpl-19 (work in progress)*, March 2011.

[12] T. Tsao, M. Dohler, V. Daza, and A. Lozano, "A security framework for routing over low power and lossy networks," Internet Draft, draft-ietf-roll-security-framework-06 (work in progress), June 2011.

[13] A. Francillon and C. Castelluccia, "Code injection attacks on harvard-architecture devices," in *ACM conference on Computer and communications security*, Oct. 2008, pp. 15–25.

[14] R. Anderson and M. Kuhn, "Tamper Resistance - a Cautionary Note," in *USENIX Workshop on Electronic Commerce Proceedings*, Nov. 1996, pp. 1–11.

[15] Q. Lampin, D. Barthel, and F. Valois, "Efficient Route Redundancy in DAG-based wireless sensor networks," in *WCNC*, Apr 2010.

[16] J. Tripathi, J. C. de Oliveira, and J. P. Vasseur, "Applicability Study of RPL with Local Repair in Smart Grid Substation Networks," in *First IEEE International Conference on Smart Grid Communications*, Oct. 2010.

[17] J. P. Vasseur and A. Dunkel, *Interconnecting smart objects with ip: The next internet*. Morgan Kaufmann, 2010.

[18] T. H. Clausen and U. Herberg, "Study of multipoint-to-point and broadcast traffic performance in rpl," *Journal of Ambient Intelligence and Humanized Computing (to appear)*, no. Spring.

[19] T. Clausen and U. Herberg, "Some considerations on routing in particular and lossy environments," in *1st IAB Interconnecting Smart Objects with the Internet Workshop*, March 2011.

[20] W. Xie, M. Goyal, H. Hosseini, J. Martocci, Y. Bashir, E. Baccelli, and A. Durresi, "Routing loops in dag-based low power and lossy networks," in *IEEE International Conference on Advanced Information Networking and Applications*, pp. 888–895.

[21] P. Soldati, H. Zhang, Z. Zou, and M. Johansson, "Optimal routing and scheduling of deadline-constrained traffic over lossy networks," in *GLOBECOM*, Dec. 2010, pp. 1–6.

[22] M. Nuvolone, "Stability analysis of the delays of the routing protocol over low power and lossy networks," Masters Degree Project, KTH, Sweden, 2010.

[23] J. G. Ko, O. Gnawali, D. Culler, and A. Terzis, "Evaluating the Performance of RPL and 6LoWPAN in TinyOS," in *IPSN*, April 2011.

[24] L. Buttyan and J.-P. Hubaux, *Security and Cooperation in Wireless Networks*. Cambridge University Press, 2008.

[25] H. Y. Chun and A. Perrig, "A survey of secure wireless ad hoc routing," *IEEE Security Privacy*, vol. 2, no. 3, pp. 28 –39, June 2004.

[26] Y. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," *Ad Hoc Networks*, vol. 1, no. 1, pp. 175–192, 2003.

[27] M. G. Zapata, "Secure ad hoc on-demand distance vector (saodv) routing," Sep. 2006, iNTERNET-DRAFT draft-guerrero-manet-saodv-06.txt.

[28] R. Hauser, M. Consulting, T. Przygienda, and G. Tsudik, "Lowering security overhead in link state routing," *Computer Networks*, vol. 31, pp. 885–894, 1999.

[29] M. Goyal, E. Baccelli, E. Baccelli, A. Brandt, R. Cragie, and J. Martocci, "Reactive discovery of point-to-point routes in low power and lossy networks," Internet Draft, draft-ietf-roll-p2p-rpl-03 (work in progress), May 2011.

[30] J. Vasseur, M. Kim, K. Pister, N. Dejean, and D. Barthel, "Routing metrics used for path calculation in low power and lossy networks," in *draft-ietf-roll-routing-metrics-19 (work in progress)*, March 2011.

[31] O. Gnawali and P. Levis, "The minimum rank objective function with hysteresis," draft-ietf-roll-minrank-hysteresis-of-04 (work in progress), May 2011.

[32] L. Lamport, "Password authentication with insecure communication," *Journal of the ACM*, vol. 24, no. 11.

[33] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko, "The Trickle Algorithm," in *RFC 6206*, March 2011.

[34] "MicaZ," *openautomation.net/uploadsproductos/micaz_datasheet.pdf*, 2007.

[35] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu, "Analyzing and modeling encryption overhead for sensor network nodes," in *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, Sep. 2003.

[36] K. Piotrowski, P. Langendoerfer, and S. Peter, "How public key cryptography influences wireless sensor node lifetime," in *Proceedings of the fourth ACM workshop on Security of ad hoc and sensor networks*, Nov. 2006, pp. 169–176.

[37] R. Roman, C. Alcaraz, and J. Lopez, "A Survey of Cryptographic Primitives and Implementations for Hardware-Constrained Sensor Network Nodes."

[38] D. Eastlake and P. Jones, "Secure Hash Algorithm 1," Internet RFC 3174, Sept. 2001.

[39] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication," Internet RFC 2104, Feb. 1997.

[40] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems."

[41] I. Blake, G. Seroussi, and N. Smart, "Elliptic curves in cryptography," Cambridge Univ. Press, ISBN 0-521- 65374-6, 1999.