RPL Authenticated Mode Evaluation: Authenticated Key Exchange and Network Behavioral

Arif Burak Ordu[1,2], Mehmet Bayar[1,2], and Berna Örs[1]
[1]Faculty of Electrical and Electronics Engineering, Istanbul Technical University, Istanbul, Turkey
[2]Aselsan Inc., Ankara, Turkey
Email: ordua@itu.edu.tr, bayar18@itu.edu.tr, siddika.ors@itu.edu.tr

***Abstract*—RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) is a standardized routing protocol that can organize thousands of resource constraint routers. Although it is an indispensable protocol with its energy-efficient, scalable, and autonomous structure, it is vulnerable to numerous attacks with its sensitive data and mechanisms. In this paper, we intensely analyzed the standard statements of the authenticated security mode of RPL and designed a comprehensive authenticated key establishment scheme extending the BKE (Bilateral Key Exchange). We formally verified our scheme using the Scyther tool. This study will help researchers contribute more to the authenticated security mode of RPL.**

***Index Terms*—IoT, security, standard, authentication, 6LoWPAN, RPL**

I. Introduction

The rapid development in the Internet of Things (IoT) applications has come up with systematic approaches and solutions for various edge devices and network technologies in the scope of many interoperability and managerial problems. At the beginning of this millennium, Internet Protocol Version 6 (IPv6) has been proposed and then developed by the Internet Engineering Task Force (IETF) to provide internetworking solutions for a growing number of devices connected to the internet [1], [2]. With the advent of the IoT technology and its recent advances, an adaptation solution for transmission of IPv6 packets over IEEE 802.15.4 has been deployed to enable resource constraint devices to be accessed directly from the internet by using IPv6 and created the term 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) [3]. Routing Over Low-Power and Lossy (ROLL) networks working group analyzed the routing requirements in urban, industrial, home, and building automation applications and defined a dynamic and self-healing protocol RPL (IPv6 Routing Protocol for Low Power and Lossy Networks) [4] compliant with 6LoWPAN. Nevertheless, RPL's self-organization and self-recovery mechanisms make it open to abuse with poisonous data contents. Any malicious usage, modification, fabrication, or deficiency on these control messages and fields makes the nodes detract from their expected behaviors. Many research articles [5], [6], [7], [8] have already discussed RPL message forging attacks and possible countermeasures. Any countermeasure step to prevent or detect RPL attacks most likely will consume extra resources (memory, process power, energy, bandwidth), which are already seen as constraints [6]. For that reason, the methods that will be deployed on the system should be carefully managed in detail.

RPL has its security features *pre-installed* and *authenticated* operation modes [4]. As the pre-installed mode, employed with the static keys, has significant weaknesses [7], [9]; the authenticated mode provides the infrastructure to solve this situation but with many shortcomings and is open to improvements.

This paper proposes a standard-compliant authenticated mode of RPL, including authentication and key establishment mechanism, comprehensive evaluations, and analysis with behavioral modeling and formal verification. The rest of the paper is organized as follows. Section II provides a brief overview of the RPL mechanisms. Section III reviews some studies on RPL security mechanisms, specifically focusing on cryptology-based and standard-compliant solutions. Section IV describes the proposed approach and system model with our motivation. It also includes behaviors of network equipment (management server, border router, routers, and hosts), some necessary assumptions, RPL control message format, and proposed security scheme, with detailed executions. Section V presents formal verification of our proposed security protocol using the Scyther tool. Finally, Section VI concludes with some evaluations of the proposed work.

II. Overview

The topology constructed by RPL is called Destination-Oriented DAG (DODAG), consisting of one or more Directed Acyclic Graph (DAG) and terminates with a single point called DODAD root. DODAG construction begins from the DODAG root with transmitted DODAG Information Object (DIO) messages and propagates to further nodes. The position of the nodes is represented by the *rank* values located in the DIO message that monotonically increase towards the outer edges of the network. During the construction, every node except the DODAG root chooses a parent. This selection creates the upward route paths, and the route paths create the general picture of the topology. The parent selection and the rank value calculation are determined by Objective Function (OF), which uses specific constraints and metrics. Depending on the configuration, RPL supports point-to-point (P2P), point-to-multipoint (P2MP), and multipoint-to-point (MP2P) traffic flows. Downward route enabled RPL has two main modes; *non-storing* mode and *storing* mode. In non-storing mode, only the root creates a routing table, and every downward packet is forwarded through to the root. In storing mode, every parent node has a routing table consisting of its child nodes and can route downward packets towards its children. These routing tables are created by signaling Destination Advertisement Object (DAO) and DAO Acknowledgment (DAO-ACK) control messages. In case of any inconsistencies such as DAG Loop detection, DAO inconstancies, link failure, etc., depending on the size of the error, RPL has the healing mechanisms called *local repair* and *global repair*. As the local repair can be handled by any router in any local DAG, the global repair can only be initiated by the DODAG root by increasing the DODAG Version Number located in the DIO message. It results in the whole network re-construction. Aforementioned RPL mechanisms [4] and configurations are dynamically managed by RPL timers (e.g., DAO delay timer, Trickle timer) [10], RPL control messages, and RPL option fields attached to datagram messages defined in [11].

III. Related Work

Traditional cryptography algorithms can provide a high level of protection. However, they are known as computation-intensive [6]. Therefore, cryptographic methods are avoided as they require extra memory and a large number of CPU cycles, as they will affect the performance of the constrained device [7]. On the contrary, researchers are searching for a lightweight solution. This section briefly presents several exemplary studies that apply cryptographic methods for RPL security. An inspirational study proposed in [9] Version Number and Rank Authentication (VeRA) and an upgraded version in [12], is based on a hash chain mechanism to protect version number and rank against spoofing attacks. However, there is more data in RPL control messages that are sensitive and must be protected.

The authors proposed a standard-compliant implementation of RPL's pre-installed security mode, including a replay protection mechanism in [13]. They analyze the performance of their implementations by comparing three types of RPL, unsecured mode, pre-installed mode, and pre-installed mode with replay protection, in terms of network formation time, node power consumption, and RPL control message overhead, with different network sizes. Their simulations on the Cooja simulator [14] showed that the replay protection with its signaling overhead adds significant increments to the mentioned performance metrics, especially with larger networks. Merging the replay protection signalling with our proposed scheme's authentication and re-keying mechanism may be a novel solution for reducing the communication overhead. The authors provide the evaluation of RPL's pre-installed mode under four common routing attacks in [15] .

The pre-installed mode may become ineffective if the keys of the nodes are compromised. To overcome this drawback, an authenticated key distribution is needed. However, with the complexity of asymmetric cryptographic methods and the overhead of the key exchange mechanisms, employing this on resource constraint devices will be challenging. A group key establishment scheme proposed in [16] is based on the elliptic curve cryptography (ECC), pre-agreement keys, and cryptographic one-way accumulators. They distribute keys via multicast communication, considering the group key features; backward and forward secrecy. They implemented their proposal in the Contiki operating system [14] and analyzed it simulating on the Cooja simulator. Energy consumption per node and communication overhead are acceptable, but, on the contrary, the size of the multicast key distribution message increases as the number of nodes increases. In [17], the authors propose a two-phase authentication scheme to improve the authenticated mode of RPL that serves the same goal as our approach. Nevertheless, nodes authenticate with both their neighbors and the Trusted Party (TP), resulting in much more signaling overhead. Moreover, nodes that store the other node's hashed ID and public keys cast apprehension on memory limitations with more extensive networks.

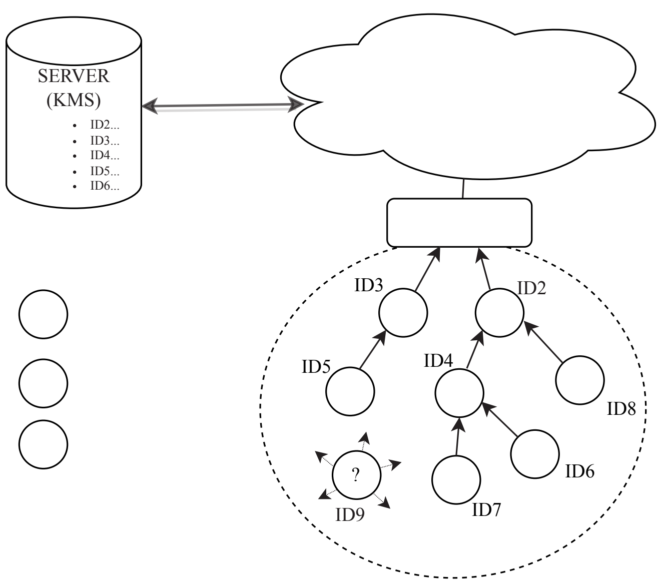


Fig. 1: System model with authenticated mode RPL

IV. Proposed Approach and System Model

RPL pre-installed security mode already provides confidentiality, integrity, and authenticity and supports reply or delay protection [4]. However, the authenticity feature of this mode depends on the Message Authentication Code (MAC) mechanism and does not guarantee the originator of the message. When RPL attacks are analyzed well, it can be seen that if we could verify whether the incoming message was actually from a legitimate or adversary, which corresponds to source authentication, and vice versa, the outgoing message could be decrypted only by legitimate nodes, we could have prevented many attacks. These kinds of mechanisms are only supported by asymmetric algorithms.

RPL standard offers the opportunity to use digital signatures and asymmetric encryption, but signing or encrypting every RPL control message with public key cryptography-based algorithms will cause redundant resource consumption on constraint nodes. In this direction, the digital signature or asymmetric encryption must be used in rare and special conditions, for example, to initialize the global and local repair, to broadcast rank update, and to inform parent nodes for updating the routing table, etc. Besides this, the RPL standard also defines an authenticated mode with shortcomings.

In this work, by sticking to the absolute requirement, absolute prohibition, and recommendations of the RPL standard, we propose an RPL authenticated mode solution that completes the points that are remained currently out of scope in the standard.The system model illustrated in Fig. 1 consists of four main actors. The first one is the Key Management Server (KMS) responsible for authority and key management and has a database with all necessary information about the proposed scheme. The second one is the root (i.e., Border Router or DODAG root) responsible for the 6LoWPAN network construction and connecting 6LoWPAN networks to external IP network (i.e., internet). More, the root also plays a leading role in the proposed authentication scheme. The next one is routers (R); they have been registered to the database by the system administrator and are allowed to use the RPL protocol to construct the network. Nevertheless, they need to authenticate themselves and obtain the second key. The last one is the hosts (H); they join the network for application data purposes only and do not influence the formation of the network routing topology.

A. Notations and Description

The following notations and their description are used in the proposed solution.

TABLE I: NOTATIONS USED IN THIS PAPER

Notation	Description
K_{pik}	Pre-installed symmetric key
$Kpri_r$,$Kpri_i$	Private key of root, node *i*
$Kpub_r$,$Kpub_i$	Public key of root, node *i*
K_{sec}	Second symmetric key
$^*K_{sec}$	K_{sec} with *Key Source* and *Key Index*
$K_{sec'}$	Updated second symmetric key
$^*K_{sec'}$	Updated $^*K_{sec}$
ID_r,ID_i	Identity of root, node *i*
n_i	Nonce generated by node *i*
n_r,$n_{r'}$	Nonces generated (by root) for *challenge*
$n_{i'}$,$n_{i''}$	Nonces generated (by root) for *trust chain*
$h(.)$	A cryptographic one-way hash function
$\{plaintext\}_K$	Encryption of *plaintext* with key *K*

B. Assumptions and Preliminaries

- The primary assumption of the RPL authenticated mode is that; routers that have the second key $Ksec$ are assumed to be legitimate and authenticated. RPL protocol is run between the routers having the second key.
- Root is always trustworthy and the security of wired connection concerns are out of the scope of this study. Any communication between the root and KMS is assumed to be authenticated and secure. Therefore, the root represents both server and itself in the scope of the proposed scheme.
- Root has more abundant resources compared to the other network nodes.
- Root knows or can obtain the database information; identification data ID_i, public keys $Kpub_i$ of the other nodes, and the relation between IDs and Extended Unique Identifier (EUI-64) [2] addresses of the legitimate nodes through the wired channel.
- All nodes have a unique EUI-64 address.
- Root knows its own private key $Kpri_r$ and pre-installed key $Kpik$. Keys of the root are assumed to be non-obtainable by the adversaries.
- Root knows the current second symmetric key $Ksec$ before it triggers the network formation and can obtain the further second symmetric keys (i.e., $Ksec'$) through the secure wired channel when the re-keying is needed.
- Nodes know their own private keys $Kpri_i$ and pre-installed symmetric key $Kpik$ as pre-installed before the deployment on the field.
- Nodes know the public key of the root $Kpub_r$ as a way; pre-installed or learned from network control plane signaling, e.g. embedded in a reserved field of the secure RPL control messages.
- Root and nodes both believe that all nonces (n_i, $n_{i'}$, $n_{i''}$, n_r, $n_{r'}$) generated during the whole signaling scheme process are fresh and also may optionally include timestamps to ensure that old communications cannot be reused in replay attacks.
- Second symmetric group keys ($Ksec$ and $Ksec'$) were generated as secure, random, and strong symmetric encryption keys.

C. Messages Formats

RPL control messages are Internet Control Message Protocol Version 6 (ICMPv6) messages with the *Type* of 155 and identified by the *Code* field located in the ICMPv6 header. Pre-installed and authenticated modes both utilize secure versions of control messages named Secure DIS, Secure DIO, Secure DAO, and Secure DAO-ACK with Codes of 0x80, 0x81, 0x82, and 0x83, respectively. In the content of these messages, a *Security Field* is inserted between the ICMPv6 header and control message body, which defines the security and encryption key configuration. The security field consists of the subfields; Counter is Time Flag (T), Security Algorithm (Algorithm), Key Identifier Mode (KIM), Security Level (LVL), Counter/Timestamp (determined by T flag), Key Identifier (Key Source and Key Index), some reserved flags and subfields. [4]

The authenticated mode employs two different keys for securing the control messages. The first one is called the pre-installed key $Kpik$ to join the network, and the other

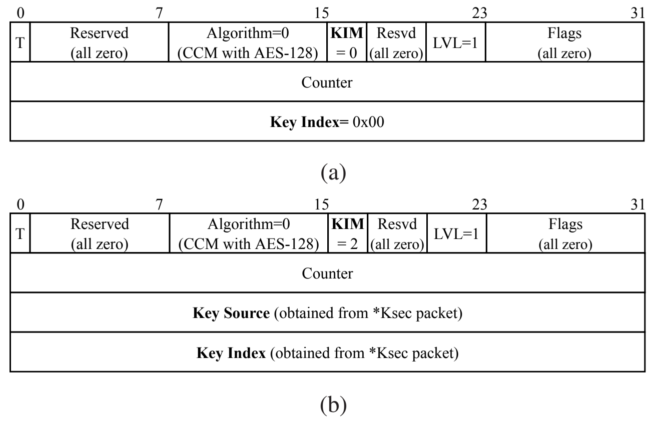

0
7
15
23
31
T
Reserved (all zero)
Algorithm=0 (CCM with AES-128)
KIM = 0
Resvd (all zero)
LVL=1
Flags (all zero)
Counter
Key Index= 0x00
(a)
KIM = 2
Key Source (obtained from *Ksec packet)
Key Index (obtained from *Ksec packet)
(b)

Fig. 2: Security field for (a) Pre-installed Key (b) Second Key

one is second key $Ksec$ to behave as a router. That results in two different configurations (i.e., KIM=1 and KIM=2) of the security field shown in Fig. 2. Both configurations make use of standard defined symmetric algorithm AES-CCM (the Advanced Encryption Standard in ”Counter with Cipherblock chaining Message authentication code” mode) with AES-128.Thus, Algorithm field is set to 0. Next field LVL is set to 1, which means both encryption and authentication (with MAC) are applied for the payload. The Counter field is incremented at each RPL message sent and used by AES-CCM for the replay protection. Key identifier fields define the employed key with Key Index and Key Source subfields. The key index value is a fixed value of 0x00 for the pre-installed key defined by the standard. Key index and Key source values for the second key are defined by the KMS.

In authenticated mode, all RPL control messages must be encrypted regardless of security configuration. Unencrypted messages and messages with MAC mismatch will be discarded by nodes. Besides, Authenticated Enabled (A) bit must be set, which is located in the *Type* of 0x04 *DODAG Configuration Option* attached to secure DIO messages shown in Fig. 3. Notably, in a secure DIO message, encrypted with the pre-installed key, the A bit informs the new node whether the network is deployed as authenticated mode.

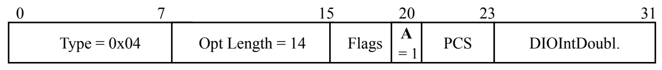


Fig. 3: The A flag in DODAG Configuration Option

D. Joining the Network as a Host-Only

A new node wishing to join a secure network first briefly waits to receive DIOs encrypted with the pre-installed key. If any pre-installed DIS was not received, depending on the neighbor count, the node solicits a multicast or simulcast DIS to trigger DIOs. Receiving node resets its Trickle timer and transmits a secure DIO with the same security configuration as the received secure DIS. The A flag in DODAG configuration options must be set in Secure DIO sent in response to DIS. The requester node processes the received DIO, selects a preferred parent, and joins the network. If the DODAG mode of operation (MOP) is configured as storing mode and the node is willing to stay as a host-only, it sends a secure DAO with RPL *Target Option* including only the IPv6 address itself. Receiving parent adds the host to its routing table. Thus, the node joined the 6LoWPAN network as a host-only and was ready to receive downward and send upward IP data packets. If it resets its Trickle timer for any reason, it is allowed to send only secure control messages with pre-installed configuration, e.g., secure DIOs with only the rank value of $INFINTE\_RANK$, and do not influence the formation of the network routing topology.

E. Joining the Network as an Authenticated Router

A node willing to be a router that requires a second key in authenticated RPL network must be authenticated by an authority. Through the root, an authentication server can authenticate the requester node by controlling whether is allowed to be a router or not before providing it with the second key. Nodes that have a second key become authenticated routers and trust each other's control messages which are utilized for network formation.

For joining the network, candidate routers follow the same procedure as the host nodes. A node gains ability to send upward application data using a standard IP packet through its parents after it selects a preferred parent and joins the network. Moreover, via this ability, it can initialize our proposed authentication and key establishment mechanism shown in Fig. 4. For this purpose; it generates a fresh random number n_i and prepares a plaintext message consisting of n_i and its identification number ID_i. Then this plaintext is encrypted with the public key of the root $Kpub_r$ and generated ciphertext is attached with the unencrypted "KEYREQUEST" word as a representation of the message type. Eventually, the node which already knows the root reachable IPv6 address obtained from the DODAGID field in the DIO message sends the key request message to the root.

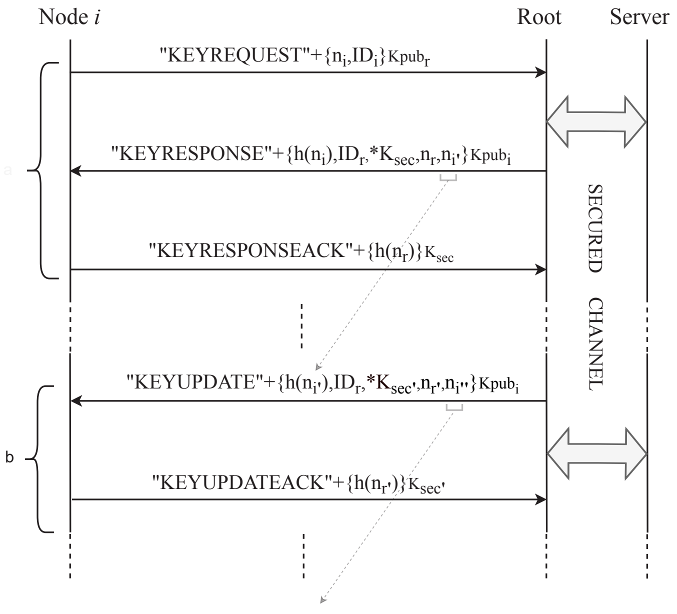

Node i
Root
Server
"KEYREQUEST"+$\{n_i,ID_i\}Kpub_r$
"KEYRESPONSE"+$\{h(n_i),ID_r,{}^*K_{sec},n_r,n_{i'}\}Kpub_i$
"KEYRESPONSEACK"+$\{h(n_r)\}K_{sec}$
"KEYUPDATE"+$\{h(n_{i'}),ID_r,{}^*K_{sec'},n_{r'},n_{i''}\}Kpub_i$
"KEYUPDATEACK"+$\{h(n_{r'})\}K_{sec'}$
SECURED CHANNEL
b

Fig. 4: Proposed (a) Authentication and key establishment (b) Re-keying scheme of node *i*

When the root receives the key request message, it first decrypts the ciphertext with the private key of the root $Kpri_r$ and obtains the incoming ID_i field. Next, it calculates the EUI-64 address of the sender node that can be reversely derived from the source IPv6 address. Using the database, with the relation between the EUI-64 address and ID of node ID_i, the root checks the presence and authenticity of the node to be a router. If the requester node is allowed to be a router, the root starts to prepare a key response message.

For this purpose, the root generates two random numbers n_r and $n_{i'}$. The n_r value is for the *challenge* that will be hashed and replied with in the key response acknowledgment message, and the $n_{i'}$ value is for the starting state of the *trust chain* relation between the root and the authenticated router, which will be used in the following key update. The authenticated routers will check the $n_{i'}$ value in the key update to whether it is from the real root. Next, the root prepares a plaintext message consisting of hash of n_i, ID of Root ID_r , the second key packet $^*Ksec$, n_r and $n_{i'}$. The $^*Ksec$ key packet includes the second key, Key Source, and Key Index values that are necessary for the second key configuration of control messages in authenticated mode. The plaintext is encrypted with the public key of requester node $Kpub_i$. Next, the generated ciphertext is attached with the unencrypted "KEYRESPONSE" word as a representation of the message type. Eventually, the root sends the prepared key response message to the requester node. This message also carries the *challenge* feature for receiving node, with the field of n_r.

When the node receives the key request response message, it decrypts the ciphertext using its private key $Kpri_i$ and checks the $h(n_i)$ and ID_r fields to ensure the message comes from the real root since only the root has known the value of n_i. If the node verifies the incoming key response message, it employs the second key $Ksec$ and becomes an authenticated node. However, one more signal is needed for mutual authentication. The root is still waiting for *challenge-response*. For this purpose, the node encrypts the $h(n_r)$ with the second key $Ksec$ and attaches an unencrypted "KEYRESPONSEACK" word as a representation of the message type. Then, the node sends the prepared key response acknowledgment message to the root. The root receives and verifies the *challenge-response* value of n_r ($h(n_r)$) and is sure that the key request message has come from a real router since only the real node has known the value of n_r.

The authenticated router is now able to be an actor, a part of network formation, and starts the control plane signaling with the configuration of the symmetric second key. If the DODAG Mode of Operation (MOP) is configured as storing mode, it sends a secure DAO with RPL *Target Option*, including the IPv6 addresses itself and its children. Receiving parents process the secure DAO if it is encrypted with the second key. The detailed control plane signaling and filtering mechanism of the authenticated routers are given in Fig. 5.

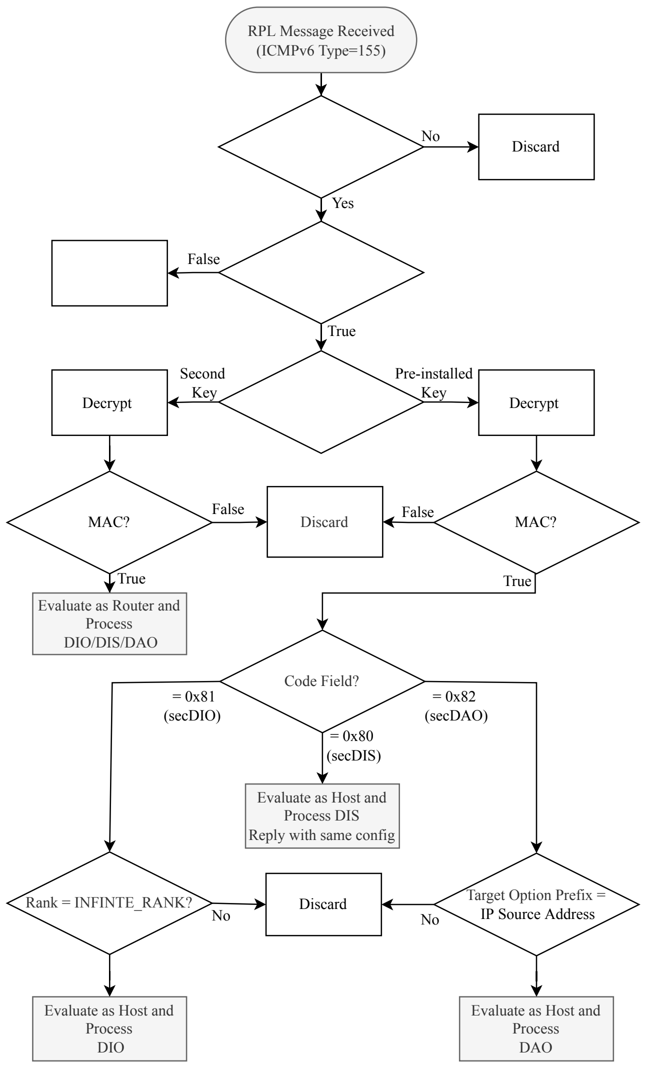

RPL Message Received (ICMPv6 Type=155)
No
Discard
Yes
False
True
Second Key
Pre-installed Key
Decrypt
Decrypt
MAC?
False
Discard
False
MAC?
True
True
Evaluate as Router and Process DIO/DIS/DAO
Code Field?
= 0x81 (secDIO)
= 0x82 (secDAO)
= 0x80 (secDIS)
Evaluate as Host and Process DIS Reply with same config
Rank = INFINTE_RANK?
No
Discard
No
Target Option Prefix = IP Source Address
Evaluate as Host and Process DIO
Evaluate as Host and Process DAO

Fig. 5: Reception of incoming control messages by authenticated nodes (router)

F. Re-Keying Mechanism

The second key used in authenticated mode of RPL is a symmetric group key that can be updated periodically or intentionally at any time to ensure the security requirements of network communication, e.g., forward secrecy, collusion freedom, and confidentiality [18]. Our proposal already adopted the Bilateral Key Exchange (BKE) protocol [19] to RPL as the initial authentication and key exchange procedure for router authentication. Considering the re-keying, we extended the current protocol based on a trust chain mechanism established with preceding mutual authentication. The proposed scheme is given in Fig. 4.

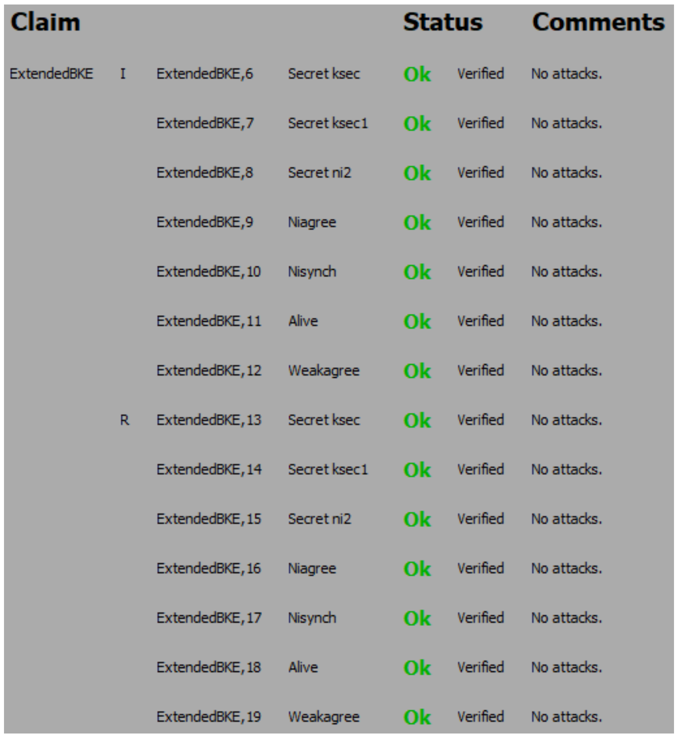
The administrator of the key update is the root in cooperation with the server (i.e., KMS). Depending on network configuration, the root can determine node key update order. For example, in storing mode, the root knows the route map of all routers; the root starts the key update process first beginning from its neighbors to deeper nodes that have lower rank values through to the outside edge of the network. In other configurations, the key update order can be determined according to the joining time already known by the root and server. When the root decides to start the key update, it generates two fresh random numbers. The first random value $n_{r'}$ is a *challenge* value for the receiving router node to use in the key update response message as an acknowledgment of the key update message. The other one is the $n_{i''}$ value used as a trust chain ticket for the next key update process. In the next key update, the root will use the hash value of $n_{i''}$ to give reassurance to receiving node. Afterward, the root prepares a plaintext message consisting of the hash of $n_{i'}$ $h(n_{i'})$, ID of root ID_r the updated second key packet $^*Ksec'$, the new *challenge* nonce n'_r and the trust chain ticket $n_{i''}$. Then this plaintext is encrypted with the public key of the target router $Kpub_i$, and generated ciphertext is attached with the unencrypted "KEYUPDATE" word at the as a representation of the message type. Eventually, the root sends the prepared key update message to the router.

With the receiving key update message, the router first decrypts the ciphertext using its private key $Kpri_i$ and checks the $h(n_{i'})$ value, which was sent by the root in "KEYRESPONSE" message as $n_{i'}$ value at the joining process. If $h(n_{i'})$ matches, the router replaces its second key $Ksec$ with the incoming second key $Ksec'$. On the other side, the root is still waiting for *challenge-response*. For this purpose, the node encrypts the $h(n_{r'})$ with the new second key $Ksec'$ and attaches an unencrypted "KEYUPDATEACK" word as a representation of the message type. Then, the node sends the prepared key update acknowledgment message to the root. The root decrypts the ciphertext and checks the *challenge-response* value of $n_{r'}$ ($h(n_{r'})$) to be sure that the key update acknowledgment message has come from the real router. Since only the real node has known the value of $n_{r'}$. After a successful key update session, the root saves the $n_{i''}$ value to use in the next key update message as a trust chain value (i.e., $h(n_{i''})$) to give reassurance to the target router.

V. Formal Verification

It is not sufficient to have a security protocol that is assumed secure. Instead, a strong guarantee is needed about security requirements stating the purpose of the protocol [20]. It can be fulfilled manually using Burrows-Abadi-Needham (BAN) logic [21] or automatically with security protocol verification tools, e.g., Scyther, ProVerif, Athena, Avispa, Casper/FDR [22].

In the scope of this paper, the proposed secure extended BKE protocol is coded in SPDL (Security Protocol Definition Language) and analyzed by Scyther tool, which provides a graphical user interface that makes it easier to verify and understand the protocol. There are two *roles* in the analyzed "ExtendedBKE"; "R" represents the root, and "I" represents the requester node willing to authenticate itself to have an opportunity of obtaining the current and following second keys. A detailed form of authentication and secrecy terminologies are proposed in [20]. In our model, we integrate the authentication properties aliveness (Alive), non-injective



Claim				Status		Comments
ExtendedBKE	I	ExtendedBKE,6	Secret ksec	Ok	Verified	No attacks.
		ExtendedBKE,7	Secret ksec1	Ok	Verified	No attacks.
		ExtendedBKE,8	Secret ni2	Ok	Verified	No attacks.
		ExtendedBKE,9	Niagree	Ok	Verified	No attacks.
		ExtendedBKE,10	Nisynch	Ok	Verified	No attacks.
		ExtendedBKE,11	Alive	Ok	Verified	No attacks.
		ExtendedBKE,12	Weakagree	Ok	Verified	No attacks.
	R	ExtendedBKE,13	Secret ksec	Ok	Verified	No attacks.
		ExtendedBKE,14	Secret ksec1	Ok	Verified	No attacks.
		ExtendedBKE,15	Secret ni2	Ok	Verified	No attacks.
		ExtendedBKE,16	Niagree	Ok	Verified	No attacks.
		ExtendedBKE,17	Nisynch	Ok	Verified	No attacks.
		ExtendedBKE,18	Alive	Ok	Verified	No attacks.
		ExtendedBKE,19	Weakagree	Ok	Verified	No attacks.

Fig. 6: Verification of Extended BKE protocol by Scyther

synchronization (Nisynch), weak agreement (Weakagree), and non-injective agreement (Niagree) for the Roles "R" and "I". Additionally, the secrecy of the crucial parameters, the second keys (ksec, ksec1) and nonce $n_{i''}$ (n2), is checked. The provincial analysis, by default settings of the tool, yielded positive results with the status "OK", yet with the comment "no attacks within bounds". Then, increasing the maximum number of runs (bounding) to 8, the Scyther results shown in Fig. 6 mean the proposed scheme is fully verified with the comment "Verified" under the claims as specified.

VI. Conclusion

In this paper, we have proposed a conceptional and well-established concrete design of a standard-compliant RPL secure authenticated mode mechanism. Our design includes a mutual authentication scheme, including the key establishment and key update mechanism based on asymmetric cryptology and the trust chain mechanism. We transferred some overhead of the public cryptography process to the pre-deployment phase that already takes an essential part in the radio-based IoT systems and applications. A pre-configuration process is applied to IoT devices before sending them to their operation field (i.e.,area of utilization) to update their software, impose necessary network parameters, and install cryptographic keys. At this stage, it is reasonable and applicable to make an installation with a small size (i.e., several bytes) for nodes to diminish overheads arising from communication and computational activities in 6LoWPAN edge environments. The public and private key generation and the installation of them into IoT devices thus prevent a considerable amount of process burden and load on the 6LoWPAN side.

References

[1] D. M. Crawford, "Transmission of IPv6 Packets over Ethernet Networks," RFC 2464, Dec. 1998.

[2] D. S. E. Deering and B. Hinden, "IP Version 6 Addressing Architecture," RFC 4291, Feb. 2006.

[3] G. Montenegro, J. Hui, D. Culler, and N. Kushalnagar, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," RFC 4944, Sep. 2007.

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

[5] A. Raoof, A. Matrawy, and C.-H. Lung, "Routing attacks and mitigation methods for rpl-based internet of things," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1582–1606, 2019.

[6] A. Verma and V. Ranga, "Security of rpl based 6lowpan networks in the internet of things: A review," *IEEE Sensors Journal*, vol. 20, no. 11, pp. 5666–5690, 2020.

[7] 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, May 2016.

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

[9] A. Dvir, T. Holczer, and L. Buttyan, "Vera - version number and rank authentication in rpl," in *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems*, 2011, pp. 709–714.

[10] P. Levis, T. H. Clausen, O. Gnawali, J. Hui, and J. Ko, "The Trickle Algorithm," RFC 6206, Mar. 2011.

[11] J. Hui and J. Vasseur, "The Routing Protocol for Low-Power and Lossy Networks (RPL) Option for Carrying RPL Information in Data-Plane Datagrams," RFC 6553, Mar. 2012.

[12] M. Landsmann, M. Wahlisch, and T. C. Schmidt, "Topology authentication in rpl," in *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2013, pp. 73–74.

[13] P. Perazzo, C. Vallati, A. Arena, G. Anastasi, and G. Dini, "An implementation and evaluation of the security features of rpl," in *Ad-hoc, Mobile, and Wireless Networks*, A. Puliafito, D. Bruneo, S. Distefano, and F. Longo, Eds. Cham: Springer International Publishing, 2017, pp. 63–76.

[14] N. Tsiftes, J. Eriksson, and A. Dunkels, "Low-power wireless ipv6 routing with contikirpl," in *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2010, pp. 406–407.

[15] A. Raoof, A. Matrawy, and C.-H. Lung, "Enhancing routing security in iot: Performance evaluation of rpl's secure mode under attacks," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 536–11 546, 2020.

[16] N. Ferrari, T. Gebremichael, U. Jennehag, and M. Gidlund, "Lightweight group-key establishment protocol for iot devices: Implementation and performance analyses," in *2018 Fifth International Conference on Internet of Things: Systems, Management and Security*, 2018, pp. 31–37.

[17] M. Razali, M. Rusli, N. Jamil, and S. Yussof, "Two phases authentication level (tpal) protocol for nodes authentication in internet of things," *Journal of Fundamental and Applied Sciences*, vol. 10, no. 2S, pp. 190–200, 2018.

[18] R. Mridula and S. Rajesh, "Group key management techniques," *Global Journal of Computer Science and Technology*, 2013.

[19] J. Clark and J. Jacob, "A survey of authentication protocol literature," 1997.

[20] C. Cremers and S. Mauw, "Operational semantics," in *Operational semantics and verification of security protocols*. Springer, 2012, pp. 13–35.

[21] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," *ACM Trans. Comput. Syst.*, vol. 8, no. 1, p. 18–36, feb 1990.

[22] N. Dalal, J. Shah, K. Hisaria, D. Jinwala *et al.*, "A comparative analysis of tools for verification of security protocols," *Int'l J. of Communications, Network and System Sciences*, vol. 3, no. 10, p. 779, 2010.