Analysis, design and implementation of secure LoRaWAN sensor networks

Bogdan Oniga, Vasile Dadarlat
Department of Computer Sciences,
Technical University of Cluj-Napoca, Romania
Bogdan@Oniga.me, Vasile.Dadarlat@cs.utcluj.ro

Eli De Poorter
Gent University - imec
IDLab, Gent, Belgium
Eli.DePoorter@UGent.be

Adrian Munteanu
Vrije Universiteit Brussel - imec
ETRO Dept., Brussels, Belgium
acmuntea@etrovub.be

***Abstract*—LoRaWAN is a LPWAN (Low Power Wide Area Network) technology used in a large variety of Internet of Things (IoT) applications. The paper addresses the security concerns of data protection and data privacy in sensor networks that make use of the LoRaWAN (Long-Range Wide Area Network) protocol specification. In this context, this paper performs an in-depth analysis of the security aspects in LoRaWAN sensor networks. Additionally, a novel secure LoRaWAN network architecture is proposed. The architecture provides protected data transmission and prevents unauthorized access and data loss. Based on experimental results and security tests, recommendations are made concerning the best practices to be followed in order to provide secure data transmission and data privacy in applications built using the LoRaWAN protocol specification.**

I. Introduction

Internet of Things are becoming ubiquitous and a core business focus in the global market. The IoT concept is widely used in a large variety of applications related to monitoring, surveillance, tracking or environment measurements. In the near future, a fast expansion of IoT is foreseen, the number of IoT devices being expected to reach the value of 50 billion till 2020 [4].

The fundamental requirements and design constraints associated with IoT applications include communication range, battery lifetime, robustness to interference, network capacity, network security, uni- versus bi-directional communication and variety of served applications. At this time, multiple vendors provide communication technologies that can achieve these fundamental requirements such as LoRaWAN, SigFox, NB-IoT, LTE-M, etc.

The technology considered in this work is LoRaWAN, which represents a good LPWAN candidate in terms of transmission capabilities and security features. LoRaWAN is capable to provide all fundamental requirements of IoT including network security, network capacity, battery lifetime and a low cost of implementation. Additionally, LoRaWAN can serve a large variety of applications, such as water and air pollution monitoring, elderly care, home security, tracking, agriculture, precision farming, and many more [5]. Furthermore, LoRaWAN provides solid security features and protections against different types of attacks such as end-node impersonation, replay messages, eavesdropping, tampering or message forgery. Despite of its capabilities, the security of systems deployed over LoRaWAN does depend on the implementation and employed security controls. Moreover, although in principle anyone can become a LoRa operator, no clear guidelines exist on how to create secure LoRaWAN networks.

Security aspects in LoRaWAN have been addressed in the past in [2], which presents an analysis and guidance concerning the security of the Long Range (LoRa) solution and its LoRaWAN protocol. This work advances over [2] by presenting additional security recommendations and best practices that have to be followed in the development of a LoRaWAN application. These recommendations were concluded based on security tests and simulations performed in such applications. Moreover, a novel secure LoRaWAN network architecture is proposed, implementing different security controls in order to protect data transmitted over the network and to establish a strong line of defense for the applications.

The paper is structured as follows. Section 2 presents a brief overview of LoRa. The security aspects of LoRaWAN are investigated in section 3. The proposed secure LoRaWAN architecture is detailed in section 4. Testing scenario results are presented in section 5. Finally, section 6 draws the conclusions of our work.

II. LoRa Technology

LoRa is a long-range wireless communication technology that targets end-nodes with limited energy (battery powered), transmitting small amount of data at a time. The physical layer, provided by Semtech, uses a Chirp Spread Spectrum (CSS) radio modulation technique and operates on the unlicensed radio spectrum in the Industrial, Scientific and Medical (ISM) bands. LoRaWAN is a protocol specification built on top of the LoRa technology to enable low power, secure, bi-directional, wide-area communication between remote sensors and gateways connected to the network [1].

A. *LoRaWAN basic architecture*

A basic LoRaWAN network architecture, follows a star topology, as illustrated in Fig. 1.

The end-nodes are communicating data over LoRaWAN through the Gateways (G). Once data are received, the Gateways are communicating the LoRa packages to the Network Server (NS), typically via 4G/Ethernet. The Network Server provides MAC commands and network control at end-node

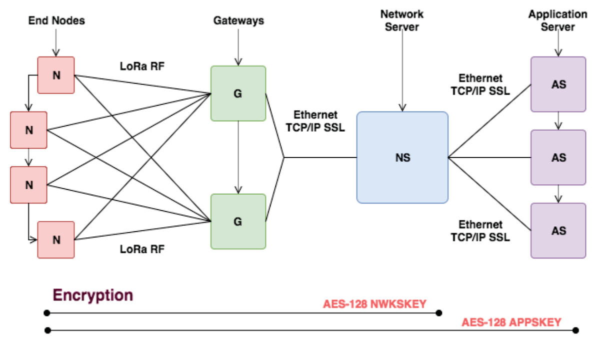

Fig. 1. LoRaWAN diagram.

(N) level. The Application Server (AS) is responsible for end-node key management and payloads sent or received by end-nodes over the network.

LoRaWAN classifies end-nodes in three different device classes (class A, B, C). Class A allows for bi-directional communication, each uplink transmission being followed by two downlink receive windows. Class B allows for bi-directional communication and scheduling of receiving slots; synchronized beacons are used by the Network Server to synchronize the listening time of end-nodes and schedule downlink transmissions. End-nodes from Class C have nearly continuously open receive windows, the exception being the uplink periods when they are sending data to the network.

B. *End-node configuration*

In order to participate in a LoRaWAN network, an end-node must be activated. LoRaWAN provides two ways to activate an end-node: Over-The-Air Activation (OTAA) and Activation By Personalization (ABP).

The information given to an end-node during the activation process is summarized in Tab. I.

III. Security aspects of LoRa

LoRaWAN provides encryption and signing of packets sent over the network. For encryption, symmetric keys known by end-nodes, Network Server and Application Server are used, these being distributed in two different ways, depending on the employed activation method.

A. *End-node activation*

The OTAA and ABP end-node activation methods are detailed and analyzed next.

1) OTAA: In OTAA activation, the LoRa device and the network server are first provisioned with a unique 128bit AppKey. Each end-node uses this key to send a join-request message, the message being signed with the AppKey. The join-request message contains the AppEUI and DevEUI of the end-device followed by a 32 nonce of 2 octets (DevNonce). These values are signed with a 4 byte Message Integrity Code (MIC).

The server calculates the MIC using the AppKey and sends a join-accept message to the end-node if the result is valid. For a valid result, the Network Server generates a nonce

TABLE I
Information given to an end-node during the activation process [1].

Identifiers	**Properties**	**Entity**	**Provisioning**
DevAddr	32bit identifier of the end-node	end-node, Application Server	Generated for OTAA; Hardened for ABP
DevEUI	64bit identifier of the end-node	end-node, Application Server	Hardened
AppEUI	Application ID - that uniquely identifies the owner of the end-node	end-node, Application Server	Hardened
NwkSKey	Key used by the Network Server and the end-node to calculate and verify MIC	end-node, Network Server, Application Server	Generated for OTAA; Hardened for ABP
AppSKey	Key used by the Application Server and end-node to encrypt and decrypt the payload of data messages	end-node, Application Server	Generated for OTAA; Hardened for ABP
AppKey	Key used by the end-node to join a network via OTAA. The AppKey is used to derive the session keys NwkSKey and AppSKey	end-node, Application Server	Hardened and exists only for OTAA

value (AppNonce) and calculates two 128bit keys, i.e., the network session key (NwkSKey) and application session key (AppSKey), based on the values received in the join-request message.

The join-accept message contains the AppNonce value generated by the Network Server, the end-node address (DevAddr), RF delays (RX Delay), and the list of channels that can be used (CFList).

The join-accept message is encrypted with the AppKey, the end-node being able to decrypt the data using the AppKey stored on the device. The AppNonce received in the encrypted message is used to locally calculate AppSKey and NwkSKey which are subsequently used in next message exchanges.

2) ABP: For ABP, each end-node is provided with DevAddr, NwkSKey and AppSKey, hence, a process of join message exchanges is not necessary for end-node activation. DevAddr, NwkSKey and AppSKey should be unique for each end-node.

3) Security aspects: Both ways of activation provide a strong level of security using symmetric encryption in message exchanges between end-node and the server. A potential attacker may inject malicious end-nodes in the network in an attempt to take advantage of the network. However, this is impossible, the provided mechanisms eliminate any possibility of an attacker to take advantage of the network in the activation process without knowing the end-node's keys.

B. *Message encryption*

LoRa provides a symmetric encryption of messages during communication. After an end-node has joined the network, the messages are encrypted and signed using a combination of NwkSKey or AppSKey.

Message encryption is performed using AES128 [3], the key used for encryption being set according to FPort value. If the value is set to 0 then the NwkSKey is used, otherwise the AppsKey is used. Encryption with NwkSKey is performed when a MAC command is sent and the AppSKey is used to encrypt the payload of data messages.

The end-node encrypts the payload using AppSKey and signs the message using NwkSKey. The purpose of message signing is to prevent the manipulation of the chiper-text, or of other values included in the message by calculating a Message Integrity Check (MIC) and validating the value at Network Server level. If the MIC is valid, the Network Server forwards the chiper-text to the Application Server which decrypts it using AppSKey to obtain the payload.

C. End-node security aspects

1) Impersonation: During our security assessment of LoRaWAN, multiple tests were performed attempting to impersonate an end-node. These tests were performed by manipulating the relevant information used by end-nodes in communication with the server.

Any attempt to manipulate the relevant information generates errors at Network Server level. This generated "noise" can be an indication of suspicious activity.

In a first scenario in which the end-node is impersonated using the DevEUI value, messages sent by a malicious end-node are rejected by the Network Server based on an invalid MIC value. The MIC value is calculated using NwkSKey, and for a malicious end-node the key is different than the key stored in the server.

In a second scenario, an end-node impersonates a session of another end-node and starts the OTAA activation process. We consider in this case that the malicious end-node is configured with the same DevEUI and AppKey as an end-node already activated in the network. Once the the malicious end-node starts the activation process, a new session is generated by the server. In this case, messages sent by the initial end-node in the network are signed using the old network session key (NwkSKey). When receiving the messages, the Network Server rejects them based on an invalid MIC value. The malicious end-node will continue to send legit messages.

2) Replay attacks: LoRaWAN offers protection against replay attacks by incrementing a frame counter, FCntUP or FCntDown, for each message sent or received by an end-node. The FCntUP counter is used for uplink messages and the FCntDown counter is used for downlink messages. The values of the counters are maintained by both end-node and Network Server, and they are initialized to 0 every time the node joins the network.

When it receives messages, the Network Server compares the FCntUP value stored on the server with the FCntUP value received from the end-node. If the received value is less than or equal to the local value, the Network Server rejects the message. This approach offers strong protection against replay attacks.

In practice, LoRa devices have a maximum value limit of these counters. According to LoRa specification, FCntUP/FCntDown are allowed to use either 16-bits or 32-bits. This limitations can introduce weaknesses in the implementation.

For a secure implementation that prevents replay attacks, the Network Server should implement a re-activation mechanism for end-nodes when FCntUP/FCntDown reach the maximum value. One notes that this mechanism can be implemented only for OTAA activation.

3) Session and key management: end-nodes should store only necessary keys required for communication. Every end-node which is part of the network should hold unique, random keys such that a compromised end-node should not be able to compromise other end-nodes.

Possessing the keys, an attacker is able to produce legit, signed and encrypted messages. Messages received from end-nodes should not be treated as trustworthy.

The keys are hardened during the chip manufacturing process, which incurs security risks. It is a risky process and the security depends on manufacture providers, if they are compromised then all the keys are exposed. Another risk represents the method used to transmit the keys to the manufacture provider.

The management of end-node sessions is handled by the Network Server, which is responsible for storing, removing and modifying end-node sessions during communication. Revoking an end-node is a simple process, removing its entry from the Application Server. If the end-node is not registered at server level, the messages are not processed.

During the end-node activation process, a new session is being generated and stored by the Network Server in its database. Each session should be stored along with an expiration time to enforce generation of a new session when the session time expires. When the Network Server stores a new session for a given end-node, it must also ensure that an old session does not exist anymore for that end-node. In case in which an old session is active, the Network Server should replace that old session with the new one.

Generating a large number of sessions and storing them in a database without replacing or removing old sessions is considered to be a security weakness. This can lead to a system out-of-memory status, in which case no additional memory can be allocated for use by local programs or the operating system. The database represents a single point of failure and in case of out-of-memory status, the network will not be functional anymore.

4) Security Recommendations: As presented above, either for impersonation or replay attacks, a lot of "noise" is generated at Network Server level. The presence of rejected messages at Network Server level should be an indication of suspicious behavior of end-nodes. A security recommendation is to implement a monitoring mechanism to determine the suspicious behavior of end-nodes based on rejected messages.

To avoid out-of-memory failures at Network Server level, the recommendation is to replace each old session with a new session when a new activation is made. Also, an expire time is necessary in order to remove inactive sessions after a period of time.

D. Gateway compromise

The Gateway is a component outside of the controlled network area and should be considered as an untrustworthy component. The Gateway is communicating directly with the internal network, which represents the most sensible point of the network.

Consider an attacker that gains access to a Gateway: in this case, the attacker is able to manipulate all radio frequency parameters used in the communication with the end-nodes. The manipulation of these parameters can lead to a higher power consumption of end-nodes and/or faster battery depletion, which in turn determines sensor breakdown and higher costs. In the worst case scenario, an attacker would be able to block all data in transit between the end-nodes and the server.

1) Security Recommendations: The Gateway is maintaining the communication between end-nodes and the server, playing the role of a packet forwarder, each message received being forwarded to the server. Message filtering is made at server level, which accepts only messages received from end-nodes listed in the database, and rejecting all other messages. To minimize the workload of the server, a best practice is to implement message filtering at the Gateway level based on end-node white-listing.

Additionally, the Gateway should not open other connections than those which are necessary to connect to the end-nodes through LoRaWAN and to the network via 4G/Ethernet.

IV. Proposed LoRaWAN secure network architecture

Besides the standard security features provided by the LoRaWAN communication protocol, a LoRaWAN network should also implement a *secure* network architecture, which applies security controls and monitoring over the network in order to provide confidentiality, integrity and availability. In this context, we propose a novel secure LoRaWAN sensor network architecture, as detailed next.

The proposed architecture, depicted in Fig. 2, implements a Public Key Infrastructure (PKI) model. In this model, each component of the infrastructure holds a certificate signed by its own Certificate Authority (CA) implemented at network level. All entities use these certificates to communicate with relevant entities inside the network. This approach provides encryption, authentication and integrity in message exchanges.

A. Gateway-network communication

The Gateway cannot be a trustworthy source, as it lies outside of the controlled network area. As depicted in Fig. 2, the Gateway is communicating directly with the Broker component of the internal network. If an attacker gains control of a Gateway, he/she is able to access directly the internal network. For this reason, the Gateway is deemed to be the most sensible point of the network.

To provide secure data transmission and alleviate potential exposure to security attacks originating from the Gateway, the proposed LoRaWAN network implements a Remote Access Virtual Private Network (VPN) server, which is a secure tunnel that provides encryption and authentication for Gateways connected to the internal network. In order to access the internal network, each Gateway holds a certificate generated by the VPN server, preventing in this way any Gateway impersonation attempts. Also, the VPN server maintain a white list of Gateway IPs, avoiding in this manner the access of multiple potentially malicious Gateways or Gateway replication.

Once connected to the internal network, the Gateways should communicate data only with Brokers in a secure way, over Datagram Transport Layer Security (DTLS) - see Fig. 2; this provides security for User Datagram Protocol (UDP) communications and prevents attacks such as eavesdropping, tampering, or message forgery.

B. Network traffic control

1) Broker level: The Broker's role is to transport end-node messages received from Gateways to upper network levels using MQTT messaging protocol. The Broker maintains two different connections, one with the Gateways over UDP, respectively with the Network Server over Transmission Control Protocol (TCP). These connections should be secured to prevent data exposure in clear text.

As mentioned, in order to secure the connection between the Gateway and the Broker, DTLS should be used, which is a protocol that secures the datagram transport. Also, to provide a secure connection between the MQTT server, implemented at Broker level, and the Network Server, the message exchanges should be made over Transport Layer Security (TLS), a protocol that is capable of securing the transport over TCP. Both entities, namely, the Network Server and the Broker, are part of a PKI infrastructure which means that each entity uses its certificate in message exchanges providing encryption and authentication of messages.

For improved security, the Broker should restrict all unnecessary communications and allow only connections with authorized Gateways and the Network Server by implementing Firewall rules at Broker level.

2) Network Server level: The Network Server plays an important role in the network, being responsible for network control and MAC commands, main actions being to manage end-node sessions, schedule acknowledgements, manage receiving windows of end-nodes, eliminate duplicate packets and adapt data rates. The Network Server is a sensible component in the LoRaWAN network, hence, it should implement rigorous security controls.

The Network Server interacts with multiple entities such as the Broker, Application Server and the database (see Fig. 2). The Network Server together with these entities are part of a PKI infrastructure model. Each connection with these entities

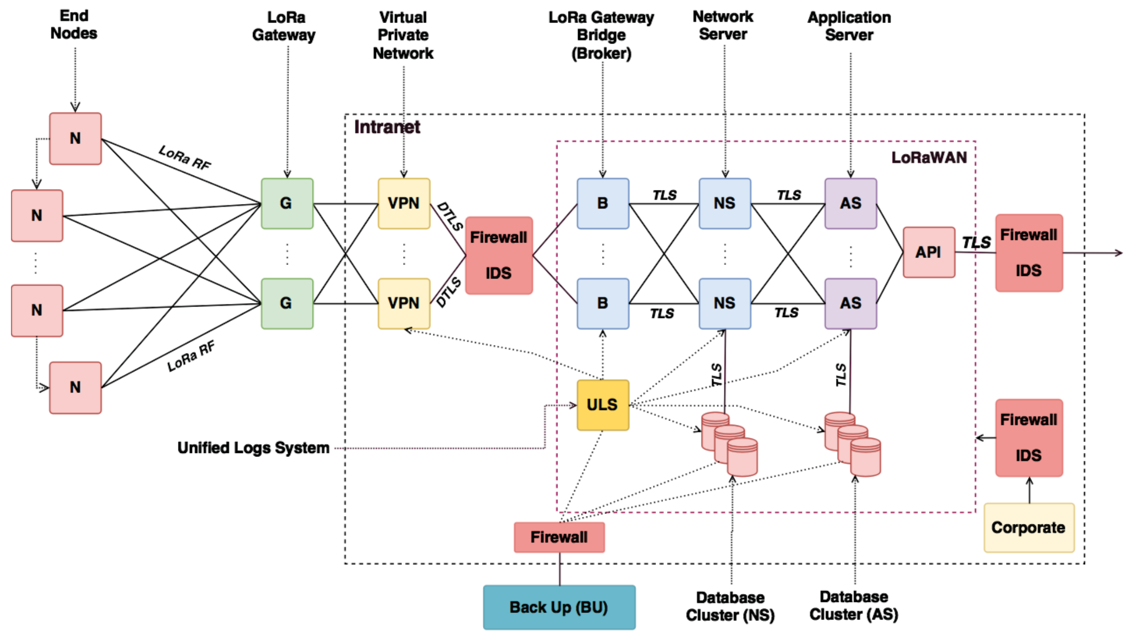

Fig. 2. Proposed LoRaWAN secure network architecture.

is made over TLS, using certificates generated by the CA. This approach provides encryption and message authentication.

The Network Server uses the network keys (NwkSKey) to sign and verify messages sent between end-nodes and the server in order to prevent data processing of altered messages. The key management mechanism should be carefully implemented because the Network Server represents a single point of failure in the LoRaWAN network.

The keys should be stored in a secure way and accessible only by the Network Server. The keys should not be read or altered by unauthorized parties.

For improved security, the Network Server should implement Firewall rules that restrict all unnecessary communications and allow only connections with the Broker, Application Server and the database.

3) Application Server level: The Application Server is also a sensible component and plays a main role for generating and storing the keys used for authentication and message encryption. Also, the Application Server stores and exposes data received from the end-nodes. It is therefore required to have rigorous security controls implemented at each level in order to protect these data.

The Application Server interacts with two entities: the Network Server and the database used to store sensitive data about the end-nodes. Each connection with these entities is made over TLS, using certificates generated by the CA part of the PKI infrastructure.

The Application Server is responsible for management of all keys in the network, generating and storing the keys used by the end-nodes in the activation process and message encryption. It is the most important component in the network. In a worst-case scenario in which the Application Server is compromised, an attacker would be able to access all end-nodes keys, all messages, all payloads and all information from the network.

The keys must therefore be stored in a secure way and accessible only by the Application Server for encryption/decryption purposes, the only exception being the network keys (NwkSKey), which have to be transmitted to the Network Server. As already indicated in section 3, these keys are used by the Network Server for network message control.

The Application Server exposes an Application Programming Interface (API), which is used for application management, end-node management and to expose end-node data. This interface is an important element of the network because it gives access to sensitive data outside of the controlled network area for other applications built based on these data.

Usually, the API exposes data through a web server. Therefore, the web server must implement a secure connection by using Hyper Text Transfer Protocol Secure (HTTPS), based on a TLS certificate signed by a global Certificate Authority (e.g. Comodo or Let's Encrypt).

The payloads received from end-nodes must be validated and sanitized by the Application Server before making any operations. The Application Server must implement functions that check if the payload meets a set of criteria (e.g. testing for the length, format, range, and allowable characters), accepting only expected payload format. These techniques are used to provide in-depth defence at application level.

Additionally, to improve security, the Application Server should implement Firewall rules that allow connections with the Network Server and its database, allow connections that come from the API and restrict all other unnecessary connections.

4) Access management: Access management is an important aspect in a secure infrastructure being responsible to granting access and privileges to authorized users. Each entity, part of the proposed infrastructure, should allow a Secure Shell (SSH) connection, which is necessary for system updating and maintenance processes. The access should be made only from the internal network for authorized persons based on user and IP whitelisting.

C. Availability

Availability is one of the most important security aspects that should be taken into consideration in every network implementation.

Availability is known as maintaining a correct functioning of the system and providing adequate communication bandwidth to keep data and resources available for authorized use. To enable availability, the following mechanisms were proposed and practically implemented.

1) Load balancing: Load balancing is a mechanism that distributes the network traffic to multiple services. Load balancers increase the capacity of a system to support concurrent connections. Implementing load balancing in a system is a solution to enhance system strength when facing attacks such as Denial of Service (DoS).

In the proposed secure architecture, load balancing should be implemented on each entity of the infrastructure, including the Virtual Private Network, the Broker, the Network Server, the Application Server, and the Application Programming Interface (API). By implementing load balancing for each entity, the system keeps data transmission continuity between components even if a service or more services are facing functional failures.

2) Database clustering: Database clustering is a solution used to increase database availability. A database cluster consists of a set of connected databases that provide database mirroring/replication and fail-over strategies.

In the proposed LoRaWAN network architecture, at least two databases are present, that is, a database connected to the Network Server which keeps the end-node sessions, and a database connected to the Application Server which keeps the data provided by the end-nodes and all encryption keys used to communicate with end-nodes. It is mandatory to use database clustering and to develop fail-over strategies for each database used in the network in order to provide continuity and availability of data transmission.

3) Data Backup: To support availability, a data backup solution is mandatory to be implemented in the network. Data backups are used in the process of restoring the network state in case of disaster or if hacking activities render the network unreachable.

In the proposed architecture, data backups are required for the databases used by the Network Server and Application Server. These databases handle all information about network state, end-node sessions and encryption keys. Frequently preserving data backups allows for network state restoration and data recovery in case of severe network failures.

Data backups are also required for the Unified logging system (see Fig. 2), which is a system that preserves each entity's logs from the network.

From a security perspective, the data backups should be preserved in a secure environment, outside of the internal network. In case of an event that renders the internal network unreachable, the persons responsible for restoring the network state should have access to data backups.

D. Monitoring and Intrusion detection system

In a secure architecture model, the presence of an intrusion detection system (IDS) is required in order to monitor network traffic for malicious activity or policy violations. Any detected suspect activity or violation is reported to the network administrator or to persons responsible with security management.

In a LoRaWAN network, there are several sensible points that require additional security controls such as an Intrusion detection system. The most sensible points are the entry points of the internal network; at these points, one has to implement firewall rules to restrict connections other than the authorized connections required by the application's functionalities.

As mentioned, one sensible point of the network is the connection of the Gateways with the internal network. In a LoRaWAN network, the inbound and outbound traffic to and from the Gateways can be predicted by calculating the traffic generated by the end-nodes. Knowing the number of end-nodes and their application configuration, the amount of regular inbound and outbound traffic can be coarsely approximated with some safety margins. By establishing such a network traffic baseline, a good practice is to implement an anomaly-based intrusion detection technique which monitors the network traffic and compares it against established baselines.

As baselines one identifies the bandwidth generally used (with some margins), the employed protocols, generally connected ports and devices; such a system generates alerts when the detected traffic is significantly different than the recorded baseline.

Another sensible point of the network is the API which supports connections from outside to the internal network. Network traffic control is provided by the implemented Firewall rules. These rules allow traffic only on the port(s) onto which the API exposes the application's data. This connection should be monitored by implementing a signature-based intrusion detection technique which compares network packets against a database of signatures or attributes from known malicious threats. If malicious threats are detected, the system generates alerts and notifies the network administrators. An anomaly-based detection technique can also be used to complement the traffic monitoring process.

The access from corporate to the internal network should also be taken into consideration and should be permitted only for authorized persons. This traffic should be monitored by implementing security mechanisms that are able to detect suspicious traffic activities and unauthorized access attempts to the internal network's entities.

V. Testing & Implementation

A. Testing scenarios

During security assessment of the proposed LoRaWAN network architecture, multiple testing scenarios were performed. Table II presents testing results and recommendations concluded from each scenario.

B. Implementation

For implementing and testing the proposed secure architecture, we used a server with Intel Core i7 CPU 860 2.80GHz quad core, 16GB of RAM, running the Proxmox VE operating system, an open source solution for virtualization [6].

1) LoRaWAN sensor network: The testing environment consists of a LoRaWAN application implemented using an open-source solution for the back end server [7], the open-source software provided by LoRa Alliance for implementing the gateways [8], [9], and an open-source solution for implementing the end-nodes [10], [11]. The employed hardware components include: Rasbperry Pi 3 and IMST iC880A SPI for the Gateways, and Raspberry Pi 3 and Semtech SX1276 for the end-nodes.

2) VPN: The software utilized for implementing the VPN server is OpenVPN [12]. Each Gateway holds a digital certificate generated by the Certificate Authority and signed with the server's certificate.

The configuration file (/etc/openvpn/server.conf) includes:

- ”cipher AES-128-CBC” - to facilitate a good level of encryption
- ”auth SHA256” - to select hash message authentication code (HMAC)
- push ”route 10.10.10.0 255.255.255.0” - to facilitate the access of Gateways into the internal network

A detailed tutorial to install and configure OpenVPN can be found in [13].

3) IDS: The software utilized in the IDS implementation is called Snort [14]. Snort is an open source intrusion detection system used to perform real-time traffic analysis and packet logging on IP networks.

In the proposed architecture, Snort is configured as a network intrusion detection, monitoring network traffic and analyzing it against different rules.

A first rule is implemented to alert any UDP traffic on a port different than 1700 between any network, in our case only the VPN network, and HOME_NET:

```
alert udp any any -> \$HOME_NET !1700 (sid
    :12345678; msg: "UDP traffic other than on
     port 1700"; classtype: bad-unknown;)
```

HOME_NET represents the internal network and its address is set to 10.10.10.0/24. In a normal flow, the Gateways send data to the internal network only on UDP port 1700. Any other traffic should be alerted and reported as a sign of hacking activity.

The second implemented rule has the main scope to alert if the size of a packet sent on UDP port 1700 between VPN and HOME_NET is over 263 bytes:

```
alert udp any any -> \$HOME_NET 1700 (sid:
    499124124; msg: "Large LoRa Packet"; dsize
    : > 263; classtype: bad-unknown;)
```

The threshold is set to 263 bytes as this represents the maximum size of a LoRaWAN packet. Any packet transmitted by the Gateways with a size over 263 bytes can be interpreted as a sign of hacking activity.

With the third rule, any attempt of TCP traffic that originates from the VPN network and has the target the internal network (HOME_NET) is generating an alert and is reported as a suspicious activity:

```
alert tcp any any -> \$HOME_NET any (sid
    :124124215; msg: "TCP traffic over the
    network"; classtype: bad-unknown;)
```

The rules implemented below generate alerts for any attempt of SSH connection or any attempt of UDP traffic on port 1194 that target the host 10.8.0.1 (OpenVPN server). Any violation of these rules is reported as suspicious activity:

```
alert tcp any any -> 10.8.0.1 22 (sid
    :214143444; msg: "SSH attempt on OpenVPN
    Server"; classtype: bad-unknown;)

alert udp any any -> 10.8.0.1 1194 (sid
    :214143445; msg: "UDP traffic on port 1194
     to OpenVPN Server"; classtype: bad-
    unknown;)
```

4) Network traffic control: Each entity implements Iptables rules which allow only network traffic necessary for a functional LoRaWAN application and for system maintenance, any other unnecessary connections being denied.

5) PKI: The proposed architecture implements a PKI model using the open-source software solution provided by Cloudflare [15]. Each entity of the architecture holds a certificate signed by the Certificate Authority (CA) and it uses it to initiate secure communications with other entities.

VI. Conclusions

This paper addresses the security concerns of data protection and data privacy in applications based on LoRaWAN. LoRaWAN's transmission capabilities and security features render it as one of the best technologies satisfying the fundamental requirements imposed by IoT applications.

LoRaWAN provides strong security features and protections against different types of attacks such as end-node impersonation, replay messages and prevents eavesdropping, tampering or message forgery by implementing end-to-end message encryption using AES128. However, these standard security features are not sufficient for secure end-to-end LoRaWAN sensor networks and applications. A LoRaWAN sensor network is a complex system assembling multiple entities which renders the process of securing it to be expensive both in terms of cost and time.

The paper presents a set of best practices and considerations that have to be followed in order to build an end-to-end secure LoRaWAN sensor network. The paper also propose and implements a secure LoRaWAN network architecture model

TABLE II
TESTING SCENARIOS, RESULTS AND RECOMMENDATIONS.

Entity	Scenario	Result	Recommendation
End-node	Sniffing LoRa Traffic	LoRaWAN messages are encrypted and signed using AES128 keys during communication, offering a strong protection against intercepting and reading data in transit.	Keys should be stored in a secure place and the access should be allowed only for authorized entities.
End-node	end-node impersonation	LoRaWAN offers strong protection against impersonation based on the sessions handled by the server. Any impersonation attempt generates error messages at Network Server level.	Implement a monitoring mechanism that alerts suspicious activities of end-nodes based on wrong messages received at Network Server level.
End-node	Replay Messages	Replay attacks can be achieved in ABP configuration, only OTAA configuration offers protection against such attacks.	Implement OTAA configuration for end-nodes.
Gateway	Manipulate communication's parameters	An exposed Gateway inevitably exposes communication. An attacker with direct access to the Gateway is able to manipulate communication's parameters which can lead to a higher power consumption on the end-nodes, which in turn determines battery depletion, sensor breakdown and higher costs.	No recommendation.
Gateway	Man-in-the-middle (MITM) attack performed between Gateway and internal network	In such a scenario, an attacker is able to alter the communication between the Gateway and the internal network components, who believe they are directly communicating with each other. The attacker is able to intercept all messages or even inject new ones.	The best option is to implement a VPN server which provides encryption and authentication for Gateways. Using a VPN connection, the internal network's resources can be accessed remotely without exposing them to the public. Extra security controls such as Firewall rules and IDS are necessary to enhance network security.
IDS	Rules violation	Performing multiple tests to violate the implemented IDS rules to test the logging and alerting mechanism	A security and event-management software should be considered to centralize and prioritize the alerts.
Network Server	Denial of Service: database Out-of-Memory	The Network Server stores new sessions and keeps the old sessions of each end-node. Generating a large number of sessions till the database capacity is completely filled yields a database out-of-memory status.	Remove or replace any old end-node session when a new session is generated. Also, each session should have assigned an expiration time.
Application Server	Sending malicious payloads to the Application Server using the end-nodes	No vulnerabilities found, but may differ from one implementation to another.	The payloads received from end-nodes must be validated and sanitized by the Application Server, which should accept only expected payload formats before performing any subsequent operations.

that implements security controls to protect data transmitted over the network and to provide an in-depth defence line for the applications.

Additional challenges in this area include investigating secure cross-network roaming solutions and efficient update mechanisms of the firmware running on the sensors.

ACKNOWLEDGMENT

This research has been supported by Fonds Wetenschappelijk Onderzoek (FWO), Vlaanderen (projects G025615 and G084117).

REFERENCES

[1] LoRa specification provided by LoRa Alliance (2015). [Online]. Available: http://bit.ly/LoRaWAN-specification,Accessedon:Jun.1,2017
[2] R. Miller. (2016, June). LoRa Security Building a Secure LoRa Solution. Presented at BSides 2016 Conference. [Online]. Available: http://bit.ly/mwr-lora-security, Accessed on: Jun. 5, 2017.
[3] J. Daemen, V. Rijmen, *The Design of Rijndael, AES - The Advanced Encryption Standard*. Springer-Verlag, Berlin, Heidelberg, Germany, 2002.
[4] Number prediction of IoT devices. [Online]. Available: http://bit.ly/iot-devices-prediction, Accessed on: Jun. 5, 2017.
[5] List of LoRa Applications provided by Semtech Corporation. [Online]. Available: https://bit.ly/lora-applications, Accessed on: Jun. 5, 2017.
[6] ProxMox website. [Online]. Available: https://proxmox.com/
[7] LoRa Server. [Online]. Available: https://docs.loraserver.io/
[8] LoRa packet forwarder repository. [Online]. Available: https://github.com/Lora-net/packet_forwarder
[9] LoRa gateway software repository. [Online]. Available: https://github.com/Lora-net/lora_gateway
[10] end-node software repository. [Online]. Available: https://github.com/hallard/arduino-lmic/tree/rpi
[11] end-node software repository. [Online]. Available: https://github.com/jeroennijhof/LoRaWAN
[12] OpenVPN website. [Online]. Available: https://openvpn.net/
[13] OpenVPN tutorial. [Online]. Available: https://bit.ly/openvpn-tutorial
[14] Snort website. [Online]. Available: https://snort.org/
[15] Cloudflare SSL software repository. [Online]. Available: https://github.com/cloudflare/cfssl