A Secure and Efficient Protocol for LoRa Using Cryptographic Hardware Accelerators

Steven Puckett, *Member, IEEE*, Jianqing Liu, *Member, IEEE*, Seong-Moo Yoo, *Life Senior Member, IEEE*, and Thomas H. Morris, *Senior Member, IEEE*

***Abstract*—Long-range wide-area network (LoRaWAN) is a low-power wide-area network (LP-WAN) protocol developed for low-bandwidth, battery-operated long-range sensors. However, the LoRaWAN specification has several security issues and utilizes software cryptography, which is not energy efficient. Our proposed solution combines a modified LoRaWAN protocol with a hardware-based cryptographic coprocessor that includes secure on-chip key storage for encryption, decryption, and digital signature creation. This design reduces the energy utilization of the wireless nodes while addressing several of the LoRaWAN threat surfaces. This article provides a security analysis of the reduced threat surfaces and demonstrates the improved energy efficiency of the LoRaWAN nodes with the integrated solution.**

***Index Terms*—Cryptographic hardware acceleration, elliptic curve cryptography, encryption, energy efficiency, long range (LoRa), long-range wide-area network (LoRaWAN).**

I. Introduction

THE LONG-RANGE (LoRa) wireless protocol was created in 2009 and designed to connect battery-operated devices that need to send small amounts of data over long distances with reliable transmission even in radio-frequency (RF)-noisy environments. It operates in the 433/800/900-MHz unlicensed bands using frequency shift chirp modulation [1] and has the capability to operate at distances of several kilometers. The LoRa specification only defines the RF physical layer and leaves the upper OSI layers and all security algorithms up to developers or protocol specifications.

In 2015, the LoRa Alliance was created to support the long-range wide-area network (LoRaWAN) protocol and to ensure the interoperability of all LoRaWAN products and technologies. Recently, the LoRaWAN Alliance has enhanced the specifications with additional security features, "over-the-air" (OTA) updates, and added new functionality into the Specification version 1.1 [2]. However, there are newer technologies that have not been adopted into the specification including cryptographic hardware coprocessors which would enhance security further.

A. Research Problem

In 2019, a security audit of the LoRaWAN specifications v1.0.2 and v1.1 found that vulnerabilities in the LoRaWAN specification allowed DoS attacks, join-accept replay attacks, downlink routing vulnerability, and an end-device activation vulnerability [3]. This study found that the LoRaWAN specifications have several security vulnerabilities that could be exploited. Chantis et al. [4] in their book "Practical IoT Hacking" discussed several vulnerabilities, such as bit-flipping in data packets if the message integrity code (MIC) is not properly utilized or becomes compromised, key extraction, replay attacks, eavesdropping, and ACK spoofing. Adefemi Alimi et al. [5] found that LPWAN technologies, specifically the LoRaWAN architecture, were susceptible to over eight LPWAN attacks. There could be other vulnerabilities that have not been identified in the LoRaWAN specifications yet.

Our proposed solution combines a modified LoRaWAN protocol with a hardware-based cryptographic coprocessor that includes secure on-chip key storage for encryption/decryption and digital signature creation. The private/public key pairs are set up during manufacturing and would no longer require additional processing (e.g., rekeying) and specialized encryption libraries. This reduces the overall power usage of the node by using the coprocessor's hardware-accelerated encryption/decryption and digital signature generation. We hypothesize that embedding a cryptographic coprocessor on each LoRaWAN node would be a very low-cost solution to provide enhanced security while reducing the energy usage of the nodes.

B. Paper Organization

This article is organized into six parts including Section I. Section II explains the current LoRaWAN communication and security protocols, vulnerabilities, and provides an overview of hardware cryptographic coprocessors. Section III discusses related works to securing LoRaWAN nodes. Section IV describes the proposed solution and explains the hardware configuration, communication packet modifications, and new

Manuscript received 30 April 2022; revised 27 April 2023 and 21 July 2023; accepted 27 July 2023. Date of publication 10 August 2023; date of current version 7 December 2023. This work was supported in part by the National Science Foundation under Grant ECCS-2312738 and Grant CNS-2247273. *(Corresponding author: Jianqing Liu.)*

Steven Puckett is with the Department of Electrical and Computer Engineering, The University of Alabama in Huntsville, Huntsville, AL 35899 USA, and also with the Department of Management and Marketing, The University of North Alabama, Florence, AL 35632 USA (e-mail: sl0114@uah.edu).

Jianqing Liu is with the Department of Computer Science, North Carolina State University, Raleigh, NC 27606 USA (e-mail: jliu96@ncsu.edu).

Seong-Moo Yoo and Thomas H. Morris are with the Department of Electrical and Computer Engineering, The University of Alabama in Huntsville, Huntsville, AL 35899 USA (e-mail: yoos@uah.edu; thm0009@uah.edu).

Digital Object Identifier 10.1109/JIOT.2023.3304175

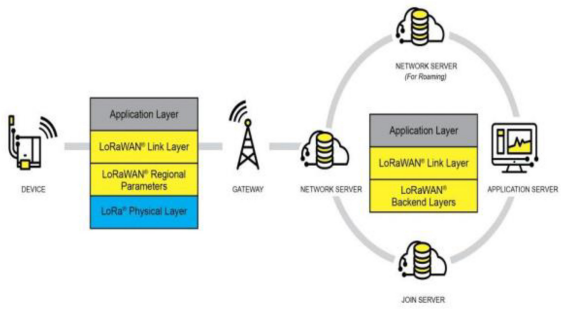


Fig. 1. LoRaWAN network architecture [2].

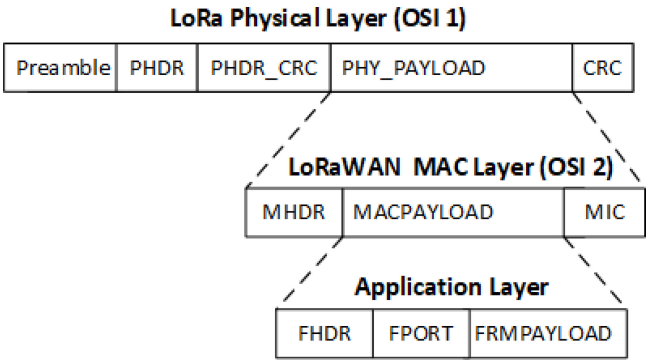


Fig. 2. LoRaWAN data packet diagram.

security protocols to enable a low-power, highly secure LoRaWAN deployment. Section V will compare a nonsecure LoRa system, the LoRaWAN specification, and the proposed solution looking at power consumption, security risks, data validation and authentication, and protocol efficiencies. Section VI will conclude this work.

II. Background

A. LoRaWAN Overview

LoRa is a proprietary LPWAN standard managed by the LoRa Alliance. It is a physical-layer specification that operates in the sub-GHz unlicensed bands. It utilizes a "spreading factor" and different channel bandwidths to optimize the system's throughput based on RF noise and distance from node to gateway. Bandwidth can vary from 300 bps up to 50 kb/s. For channel access, it uses the ALOHA algorithm with a single-hop topology.

The control of the LoRaWAN architecture focuses on a network server that controls communications and security and identifies which nodes should communicate to which gateways. Fig. 1 shows the overall architecture of the LoRaWAN v1.1 specification.

B. LoRaWAN Packet Format

Fig. 2 outlines the LoRaWAN packet by breaking it out into three layers. The first layer is the LoRa RF wireless physical layer and defines the radio interface and modulation scheme along with an optional CRC. It encapsulates all the other LoRaWAN layers as the payload. The second layer is the MAC layer which defines the message type (contained in MHDR), the MIC, and the MAC-layer payload. The application layer contains the device address (defined in FHDR and FPORT) and the data payload.

TABLE I
LoRa Physical-Layer Packet Format (Uplink)

	Field Size	Encrypted?
Preamble	8 symbols	No
PHDR & PHDR_CRC	4 bytes	No
PHY PAYLOAD	x bytes	No
CRC	2 bytes	No

TABLE II
LoRa MAC-Layer Packet Format (Uplink)

	Field Size	Encrypted?
MHDR	1 byte	No
MAC PAYLOAD	x bytes	No
MIC	4 bytes	No

TABLE III
Application-Layer Packet Format (Uplink)

	Field Size	Encrypted?
FHDR	7 to 22 bytes	No
FPORT	1 byte	No
FRMPAYLOAD	x bytes	Yes

Table I breaks out the PHY-layer format of an uplink packet according to the LoRa specification. The preamble is a simple leading sequence of modulated RF "chirps" that are used to notify listening devices that a LoRa message is being sent. The PHDR contains the length of the PHY-layer packet, error correction coding rate, and a flag to indicate if the optional PHDR CRC exists.

The LoRaWAN MAC-layer uplink packet as shown in Table II consists of the MAC header (MHDR), payload, and MIC.

MHDR specifies the message type and major version of the LoRaWAN frame format. The frame payload (FRMPAYLOAD) is encrypted before the MIC is calculated. The MIC is computed using AES-CMAC of the MAC Header, frame header (FDHR), and Payload to detect packet tampering and to further protect the security of the message.

The FHDR, contained in the application-layer uplink packet, as shown in Table III, consists of the short device address, the frame control octet, counter, and frame options. The frame port (FPort) identifies if the FRMPAYLOAD contains MAC commands only or data as well. The FRMPAYLOAD is encrypted utilizing the IEEE 802.15.4/2006 Annex B generic algorithm based on 128-bit AES [6].

C. LoRaWAN Security Overview and Vulnerabilities

The LoRaWAN Specification v1.1 uses a unique 128-bit AES key (AppKey) and a unique identifier (EUI-64-based DevEUI) to identify each LoRaWAN end device as shown in Fig. 3. These are used during the device authentication and to generate two session keys (NwkSKey, AppSKey) [6]. The

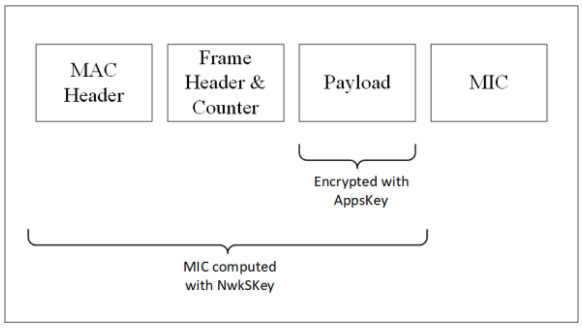


Fig. 3. LoRaWAN packet protection.

NwkSKey is used to verify the packets' integrity and authenticity and gets distributed out to the LoRaWAN network. The AppSKey is used to provide end-to-end encryption of the application payload and is hidden from the network and gateways to allow third-party gateways to be used for network access. The AppSKey is also utilized by the application server to encrypt/decrypt the application payload. However, these keys can be discovered if the remote nodes are compromised by an attacker. The LoRa Alliance has even acknowledged this possibility [7] and the problem has been well described by Aras et al. [8]. This introduces the potential risk of malicious attacks on the remote nodes. For example, an attacker could update remote nodes with malicious firmware, send a configuration to tell the node to never broadcast, or even cause a broadcast storm of all nodes to completely saturate the gateway causing a complete failure of the system [3]. A more secure protocol can be designed to provide the needed functionality while enhancing security by utilizing hardware-based cryptographic coprocessors. Our solution aims to address the following types of vulnerabilities while improving energy efficiency.

1) Bit-Flipping Attack: It can be used to change values in parts of the message [3]. Thomas et al. [9] showed that since the Counter (CTR) mode of AES encryption is used with the plaintext being XOR'ed to encrypt the data, thus making the bit-flipping attack simple and very possible.

2) Key Extraction: All unicast uplink and downlink messages are encrypted at the LoRaWAN MAC Layer using the end device's AppSKey [2]. These keys are generated from the AES128 McKEKey and are stored locally on the remote nodes in accessible memory locations. It is likely that an attacker could read the keys directly if they were able to physically access the node.

3) Replay Attacks: It was found that self-replay attacks posed a high-risk to the LoRaWAN devices [6] [10].

1) *Uplink Replay:* For OTAA and ABP-activated nodes, the 16-bit frame counter (i.e., a sequence number) is in plaintext and resets to zero when power cycled or when the counter wraps around.
2) *Downlink (ACK) Replay:* A captured ACK message could be replayed to let a node know that the message it sent was acknowledged, even if the message was being blocked before the gateway or between the gateway and the network server.

4) Key Reuse: LoRaWAN ensures confidentiality using AES in counter mode (AES-CTR) which relies on a counter to encrypt streams of data instead of a random number called a nonce. LoRaWAN does not set the counter as a nonce, but instead uses the frame counter as input [11]. During a reset of the node, the counter value is reset and the key remains the same which means the block cipher will create the same key material. This is called a keystream reuse [12]. A process called "crib dragging" can be used to guess parts or all of the plaintext [13].

5) Beacon Vulnerabilities: LoRaWAN gateways broadcast beacons on a regular interval to synchronize Class B devices for initiating downlink traffic. These beacons are not encrypted or protected and manipulation of the timing references in the LoRaWAN beacon could cause a desynchronization of Class B devices, thus causing a Denial-of-Service attack [3].

D. Cryptographic Coprocessor Overview

The motivation for using a cryptographic coprocessor instead of software-based cryptographic libraries and memory is as follows. First, the keys could potentially be read from memory if an attacker was able to physically obtain a node. The attacker could also perform a replay attack, ACK spoofing, or it-flipping attack [4]. These malicious attacks could be prevented by utilizing a hardware-based cryptographic coprocessor since the keys are pregenerated, stored securely, and all messages are encrypted within the secure hardware chip.

Another reason for utilizing cryptographic coprocessors is the power savings. In existing practices, LoRaWAN nodes must utilize software algorithms for key generation, encryption/decryption of data packets, and for signature generation/verification. This requires the node processor to perform the calculations which consumes valuable battery power. It has been shown that hardware-based cryptography outperforms software-based algorithms by over 100% [7]. This performance enhancement can increase the device responsiveness and reduce the overall duty cycle of the device, thereby prolonging a device's battery life. Most hardware accelerators use a few microamps with some of the more current ones only using nanoamps of current in their deep sleep mode. During the encryption/decryption process, the chips use only a few milliamps of current for a very short amount of time. This adds valuable battery life and could possibly reduce the required battery size for the LoRaWAN nodes, thus lowering the overall cost of the node. Moreover, low-end cryptographic chips are only a few tens of cents in cost. With the additional security enhancements, there is now a movement by developers and manufacturers to begin utilizing hardware-based cryptographic accelerators on IoT systems due to their low-cost, secure key storage, and ease of use [14].

Typically, these cryptographic hardware coprocessors include secure key storage, elliptic curve digital signature algorithm (ECDSA), 128-bit or 256-bit AES encryption, secure random number generators, and hash and key generation functions. Table IV shows a few of the encryption hardware System-on-a-Chip (SoC) that are being manufactured for IoT devices. These models were evaluated and the Microchip

TABLE IV
CRYPTOGRAPHIC HARDWARE COPROCESSOR CHIPS FOR IOT APPLICATIONS

Manufacturer	Model	Price ($)	Size (mm)	Interface	Encryption Capabilities	Key Size	Secure Key Storage	Sleep Current	Standby Current	Active Current
Microchip Technology	ATECC508A	0.36	2x3	I2C	ECC-P256 (ECDH and ECDSA), SHA256	256	16	<150 nA	0.03 uA	1 mA
Microchip Technology	ATECC608B	0.93	2x3	I2C	ECC-P256 (ECDH and ECDSA), SHA256, AES128-GCM	256	16	<150 nA	0.03 uA	< 1 mA
Analog Device	DS28S60	1.85	3x3	SPI	SHA-256 MAC, HMAC Hash, AES-128 with GCM, ECDSA-P256, ECDHE-P256 Key Exchange	256	3.6 KB Flash Array	100 nA	0.35 mA	1.62 mA
NXP	SE050B2HQ1	3.50	3x3	I2C	AES and DES, HMAC, CMAC, SHA-224/256/384/512 AES Modes: CBC, ECB, CTR HKDF, MIFARE KDF, PRF (TLS-PSK), RSA and ECC	224,256 384,512	Secured flash memory up to 50 kB	< 5 uA	< 0.5 mA	4.4 to 16.1 mA

ATECC608A coprocessor was selected for testing due to its energy efficiency and functionality. This coprocessor is also now being incorporated by IoT microcontroller manufacturers as part of their SoC [15].

III. RELATED WORK

Gunathilake et al. [16] did a review of utilizing lightweight cryptography (LWC) and found that it could reduce the power used by encryption by as much as 26% while also using less memory in the nodes. However, they found that LWC was less secure than existing LoRaWAN specifications. Thomas et al. [9] validated that man-in-the-middle attacks based on using AES-CTR were not only possible, but could also be used as a valuable attack tool. They determined that using AES in the Galois Counter Mode with an authenticated decryption function would defeat a man-in-the-middle attack while also authenticating and protecting the data. The cryptographic coprocessor used for our evaluation was chosen based on this analysis.

Shahjalal et al. [17] introduced the idea of integrating the LoRaWAN protocol with blockchain to secure the application-layer data specifically while otherwise leaving the LoRaWAN protocol untouched. They also create a randomized *SyncWord* symbol that is assigned to each node and uses the LoRa Layer 1 preamble to broadcast the *SyncWord* to the gateway. The gateway then authenticates the node based on the *SyncWord*. Their solution significantly increases latency and encryption time of the data at the gateway and would require additional power and processing time at the nodes and gateway. Also, the *SyncWord* could easily be captured for replay attacks. Zhang et al. [18] proposed combining the Chinese remainder theorem and received strength signal indicator (RSSI) to protect the transmission bits from eavesdroppers which would require modifications of the RF Layer of the LoRa protocol.

LoRaWAN key management is a major issue since all keys are generated by the AppKey which never changes and is accessible at the node. Hayati et al. [19] utilized a Join Server to create new AppKeys periodically and transmitting them to the nodes which requires the nodes to transmit and receive four additional messages per key update and would only work for LoRa nodes that could transmit in the higher bandwidths. Utilizing a secure hardware-based key storage cryptographic coprocessor would alleviate the need for new AppKey generation altogether.

The LoRaWAN specifications do not cover server-to-server security, such as between the application, join and network servers. Tsai et al. [20] proposed a secure key generation by combining AES-128 and elliptic curve cryptography to create session keys between servers. Integrating the hardware cryptographic coprocessor within the gateways and servers would also simplify secure communications between the servers as well as the nodes. However, this is outside the scope of this article. The summary of the above works is shown in Table V.

IV. PROPOSED SOLUTION

The goal of the enhanced protocol is to provide a highly secure communications pathway between remote nodes and the gateway platform. An analysis of the existing security issues with the LoRaWAN specifications and the research of others was utilized to develop a simple method to provide the mechanisms needed for enhanced security. For this work, Microchip Technology's ATECC608A CryptoAuthentication device was chosen [21]. It is a cryptographic coprocessor with secure hardware-based key storage. The private key is securely stored in the chip itself and never revealed to the outside world. The chip has numerous physical protections to prevent anyone from getting access to the private key storage area. The ATECC608A has the capability to store 16 different 256-bit keys with support for ECDH, elliptic curve digital signature

TABLE V
RELATED WORKS IN LORAWAN SECURITY

	Encryption / Eavesdropping	Bit-Flipping Attack	Key Extraction	Replay Attacks	ACK Spoofing	Beacon Vulnerability
[18] A Novel Physical Layer Encryption Algorithm for LoRa	✔	-	-	-	-	✔
[9] Man in the Middle Attack Mitigation in LoRaWAN	✔	-	-	✔	✔	-
[16] Next Generation Lightweight Cryptography for Smart IoT Devices: Implementation, Challenges and Applications	✔	-	-	-	-	-
[17] Implementation of a Secure LoRaWAN System for Industrial Internet of Things Integrated with IPFS and Blockchain	✔	-	-	-	-	-
[20] Secure Session Key Generation Method for LoRaWAN Servers	✔	-	✔	-	-	-
[19] A Novel Secure Root Key Updating Scheme for LoRaWANs Based on CTR AES DRBG 128	✔	-	✔	✔	-	-
Proposed Solution	✔	✔	✔	✔	✔	✔

✔ = corrects given threat

(ECDS), AES-126/256 with GCM, as well as SHA-256 with HMAC [22]. It also provides extremely low-power consumption (e.g., less than 150 nano-amperes (nA) sleep current), making it ideal for low-end IoT devices. Moreover, since the security algorithms are performed by the ATECC608A in hardware, the overall duty cycle of the system is reduced which saves additional battery life for LoRaWAN nodes. Encryption and decryption would take place internally inside the cryptographic coprocessor further increasing the security of the system while also reducing processor cycles on the nodes and gateways; no software encryption libraries are thus needed. The generated private key and stored public keys are never exposed outside of the chip or on any internal communication buses making key extraction infeasible.

The LoRaWAN Layer 1 PHY_PAYLOAD packet would be the only modification needed to the LoRaWAN specification for the enhanced security. The MAC and Application Layers would be untouched so that the design would still meet the LoRaWAN specifications and be interoperable with other LoRaWAN systems if so desired, though the additional encryption is not required. The gateway would require slight modifications so that the Layer 1 payload can be decrypted and the packets can be authenticated. However, application data encryption algorithms should utilize the cryptographic coprocessor to further improve the power usage of the node.

A. LoRaWAN Protocol Modifications

The proposed modifications to the existing LoRaWAN protocol require the following steps.

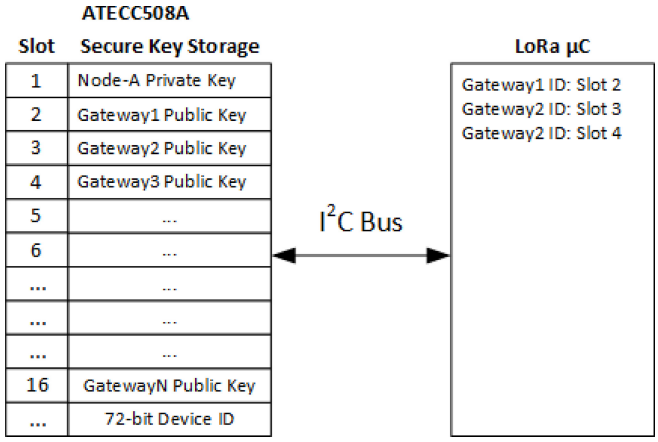


Fig. 4. Gateway key storage on a LoRa node.

1) Asymmetric Key Generation and Storage: Before deployment, an asymmetric public–private key pair is generated on each LoRa node using the cryptographic coprocessor. The LoRa node's private key is securely stored on the chip while its public key and unique ID are stored within the main memory, database, or software of the gateway as a lookup table.

The public keys of the gateways that are available to the LoRaWAN node are loaded into the secure storage of the crypto coprocessor (the ATECC608A has 16-slots as an example) and each gateway's ID is noted as to which secure slot their respective public key is stored. Whenever the sensor node needs to decrypt an encrypted message or verify a message, the selected gateway's ID is used to look up the slot number that the corresponding gateway public key is stored in. The keys never leave the chip to do any encryption or decryption, thus keeping them secure and out of public view. Fig. 4 displays how the keys are stored within the LoRa node.

2) Secure Node Identification: The ATECC608A has a built-in, guaranteed unique 72-bit serial number. This unique ID is used as part of the encryption key input and used to validate each message. The device ID is also encrypted as part of the data payload as it is transported to the LoRaWAN gateway to prevent eavesdropping of the device ID. The remote gateway can then decrypt and validate that the remote node is a valid and authentic node. This provides extra protection against known plaintext attacks and identifies any compromised nodes. It also prevents the introduction of rouge LoRaWAN nodes from being introduced into the network.

3) Transport Layer: The transport layer will reside on the LoRa physical layer (OSI Layer 1) as the PHY PAYLOAD. Inside the payload will reside the header (HDR), the unique node device ID, the Message Data includes the LoRaWAN MAC and Application Layers, and an HMAC of the device ID and message data. Fig. 5 shows the layout of the packet. The HDR (header) contains the 8-bit gateway ID so that a receiving gateway can immediately determine if the message was being sent to it before deciding to decode the rest of the message. If the message is addressed to the gateway, the gateway then authenticates the message to ensure that no data bits were changed or lost. If the signature checks out, then the

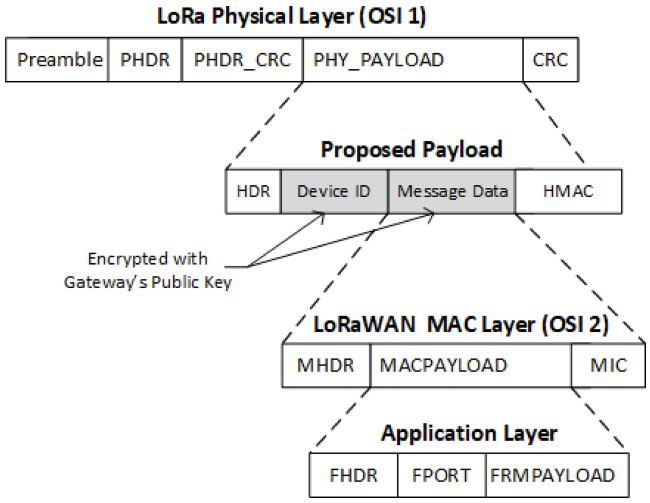


Fig. 5. Proposed data packet diagram.

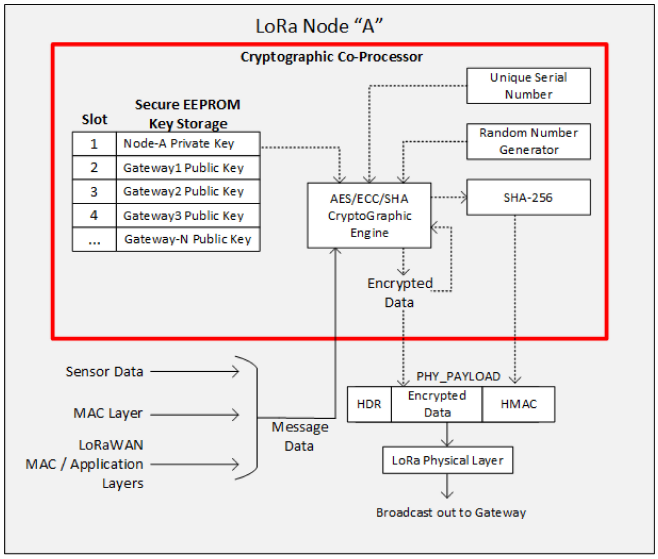


Fig. 6. LoRa node data flow diagram.

gateway decrypts the device ID and message data. The 72-bit device ID is important because it identifies the end node's public key, and it is used as part of the encryption and HMAC encoding for responding messages from the gateway back to the node. To reduce message overhead, only the last 4 bytes of the device ID would be required to identify the LoRaWAN node. In larger networks the full device ID could be utilized if needed

4) Creating Secure Message: The overview of this process is shown in Fig. 6. When the remote node is ready to transmit, the software concatenates the ATECC608A's secure serial number, used as the device ID, with the data to create the Message Data (M). The key storage slot ID of the receiving gateway, along with the message (M) is passed to the ATECC608A. The gateway's public key is used as the encryption key for the message. This creates the encrypted ciphertext which is then used as the data payload for the packet. The ciphertext is passed to the ATECC608A to generate the HMAC that is then added to the end of the PHY packet. The gateway ID is added to the PHY packet as part of the header and encapsulated by the LoRA Layer 1 packet. The entire packet is then sent to the gateway over the RF link. All identifying aspects of the data, device ID, and message authenticity and integrity are protected using this method. After the packet has been broadcasted, the sensor node then waits for a predetermined amount of time for an acknowledgment from the gateway.

5) Gateway Functionality: All gateways receive the Layer 1 LoRa packet, but parse the HDR information to determine if the Gateway ID matches. If so, the HMAC is checked to validate the message. The reason for message validation first is to reduce resource load on the gateway. If the HMAC fails, then decryption is not necessary and a resend of the data is required. If it is validated, then the message data is unencrypted with the gateway's private key. The device ID and message data is then processed by the gateway's software per the LoRaWAN standards. A response message back to the sensor node is then created, encrypted and an HMAC is added in the same way. The response message can contain further instructions for the sensor node, (i.e., setting a new broadcast time interval or setting an interval variable value) and to validate that the message was received and processed by the correct gateway. The sensor node waits for a predetermined amount of time to listen for the response message from the gateway. If the message is not received, a retransmission of the original message by the sensor node might be instigated. Outside of the addition of new software functions to read the device ID, compute the HMAC of the message, and compare the results with the incoming message, no further modifications to the LoRaWAN gateway would be required. Encryption can be performed using standard software libraries or several crypto coprocessors could be integrated with the gateway hardware over the I2C bus to improve its performance as well. Each LoRaWAN gateway RF channel could be assigned a crypto coprocessor to enhance the encryption/decryption and HMAC computations.

B. Security Analysis of Our Solution

The proposed solution utilizing a hardware cryptographic coprocessor and modified LoRaWAN protocol defeats each of the vulnerabilities outlined below.

1) Bit-Flipping Attack: Utilizing the HMAC functions of the cryptographic coprocessor algorithm on the entire message payload before transmitting and adding the HMAC at the end of the message would prevent bit flipping, validate that the data is correct, and authenticate that the message came from the correct node. The node would utilize the generated public/private key pairs for the HMAC algorithm. To reduce the size of the transmitted packets, only the last 4 bytes of the HMAC is utilized. When the message is received by the gateway, the gateway computes the full HMAC and then verifies the last 4 bytes match those in the message. This still provides a high level of confidence and protection from bit-flipping.

2) Key Extraction: The LoRa node's private key and the gateway's public keys are stored in secure memory with no access from outside and all encryption and signature generation are performed inside the crypto coprocessor chip. Most manufacturers also include numerous physical and electrical security features, including Active Shield Circuitry, Internal Memory Encryption, Glitch Protection, and Voltage Tamper

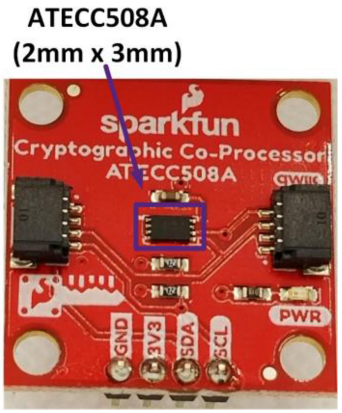


Fig. 7. ATECCX08A cryptographic coprocessor.

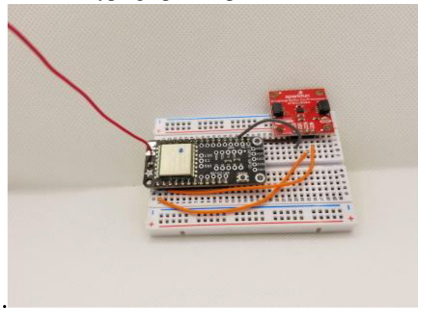


Fig. 8. Adafruit's feather M0 with RFM95 LoRa radio.

Detection. The keys and encryption/decryption algorithms stay within the chip and are never exposed to the bus or device pins. Both the logic clock and supply voltage are internally generated to protect the rest of the device from any attacks using the device pins.

3) Replay Attacks: In the newer LoRaWAN specification, it is recommended that the counters are stored in nonvolatile memory and read after a power cycle of the node to prevent the reset of the counters. The cryptographic coprocessors have on-board counters and protected memory that meet this recommendation.

4) Eavesdropping: The onboard Random Number Generator is utilized to generate a new nonce for AES-128 message encryption and does not utilize a software counter as stated in the current LoRaWAN specifications. However, a solution that follows with the LoRaWAN v 1.1 specifications would be to utilize the counter in the cryptographic chipset so that it would be secure in the event of a power outage or node reset and not exposed to modification or eavesdropping from outside the chip.

5) ACK Spoofing: Each data message sent uses an encrypted and incremental sequence number generated by the hardware encryption coprocessor which securely stores the sequence number even in the event of a power cycle or reset event. A short timeout (100–150 ms) is setup to only accept an ACK response within the time window to further reduce the chances of a replay attack.

6) Beacon Vulnerabilities: Future LoRaWAN specifications should include the use of a hardware cryptographic chipset to digitally sign the synchronization beacon. Only those nodes with the correct key for the beacons would be able to authenticate the beacons.

TABLE VI
PROPOSED TRANSPORT-LAYER PACKET LAYOUT

	Field Size	Encrypted?
HDR	1 byte	No
Device ID	4 bytes	Yes
Message Data (includes LoRaWAN MAC and Application Layers)	x bytes	Yes
HMAC	4 bytes	No

TABLE VII
LoRa PACKET SIZES FOR A 32-BYTE PAYLOAD

	Unsecured LoRa (BYTES)	**LoRaWAN** (BYTES)	**Proposed LoRaWAN Changes** (BYTES)
Layer 1	7	7	7
MAC Layer	-	1	1
Application Layer	-	23	23
Plaintext Data	32	32	32
HMAC + ID	-	4	8
Total	**39**	**67**	**71**

V. PERFORMANCE EVALUATION

All test setups and models utilize Adafruit's Feather M0 Radio with the RFM95 LoRa radio module as shown in Figs. 7 and 8. The microcontroller is an ATSAMD21G18 ARM Cortex M0 processor, clocked at 48 MHz with 256K of FLASH and 32K of RAM. This processor was chosen for its faster clock speeds and higher memory capabilities and for the integrated LoRa 900-MHz radio. For IoT deployments, this system would be typically used for industrial and utility applications where higher security and reliability are more important than with most consumer products. The hypothesis is that if the ATECC608A chipset performs as well as the higher end IoT microcontrollers, then the value-add of the ATECC608A could be justified. As you will see, the ATECCX608A performed better than existing software-based encryption solution.

A. Message Overhead

Given a plaintext message size of 32 bytes, the total message sizes and encrypted parts of the packet were calculated. Table VI shows the calculated packet sizes for plain LoRa with no encryption or other higher-level functions as a baseline, the LoRaWAN packet size and the proposed LoRaWAN protocol outlined in this article. The existing LoRaWAN specification is 28 bytes larger than the baseline, while the proposed protocol is only 32 bytes larger than the baseline.

For LoRaWAN, only the plaintext data payload is encrypted, which for this example is 32 Bytes. This means that all other information in the packet is sent in plaintext and any RF sniffer can capture this data. The proposed protocol encryption data

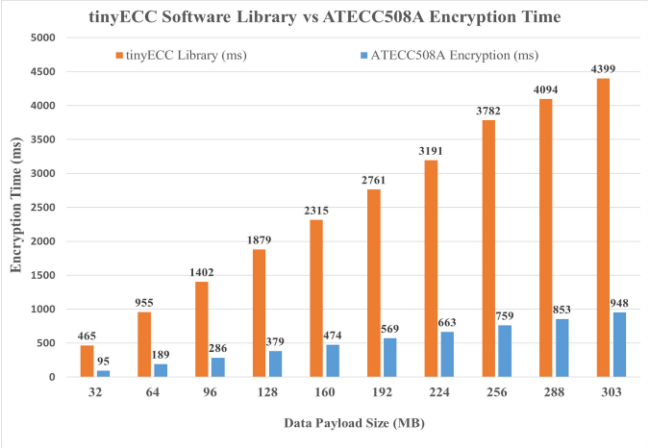


Fig. 9. Encryption time comparison between tinyECC software library and ATECCX08A.

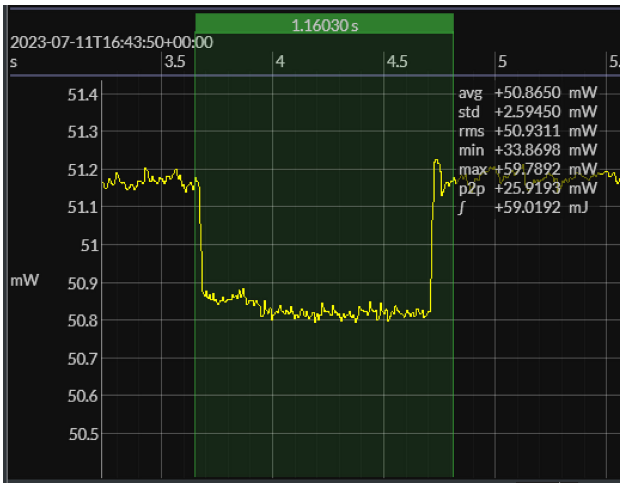


Fig. 10. Crypto library software-based SHA-256 HMAC time and energy graph for a 32-byte data packet. It took 1.16 s and used 59 mJ to complete.

size was 32 bytes (data) plus the 4-byte Device ID and the 4-byte HMAC for a total of 71 bytes which is only 4 bytes larger than the standard LoRaWAN protocol, which is summarized in Table VII. This increase is negligible in the overall design, even at 300 bps for the added security and functionality that is attained. LoRaWAN supports data payload sizes from 11 to 256 bytes.

B. Computation Overhead

The Adafruit Feather M0 Radio was used to test the encryption times and power utilization as a comparison between using the ATECCX08A chipset and software-based encryption libraries. The Joulescope JS220 Precision Energy Analyzer [23] was utilized to measure the processing time and energy used by the encryption and HMAC functions for both software and hardware. The micro-ecc Arduino library created by Ken MacKay [24] was modified and loaded onto the Adafruit Feather M0 to calculate the time required to utilize Elliptic Curve Cryptography to encrypt and decrypt data of different lengths to compare with the ATECCX08A and to create and verify an HMAC (SHA-256) generated signature. These tests are not meant to be a test of the algorithms used between the Arduino libraries and the ATECCX08A platform. The tests were a comparison of the same IoT platform utilizing a software-based security solution versus the addition of the cryptographic hardware accelerator. The execution times for the ATECCX08A are the same for both encryption and decryption as described in the ATECCX08A datasheet.

Fig. 9 shows that the ATECCX08A was nearly five times faster at encryption than the standard ECC software encryption libraries. As part of the proposed LoRaWAN protocol modifications, the LoRaWAN MAC (Layer 2) is encrypted and then an HMAC using SHA-256 is created and added to the PHY_PAYLOAD to authenticate and validate the message at the gateway. A comparison between the CRYPTO library [25] SHA-256 HMAC and the ATECC608A HMAC was performed as shown in Figs. 10 and 11. The ATECC608A completed the HMAC 14 times faster than using the software library.

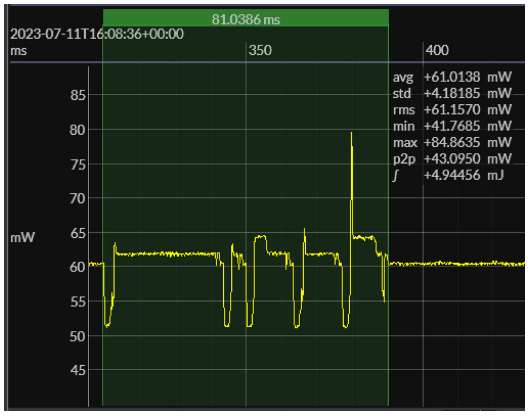


Fig. 11. ATECC608A SHA-256 HMAC time and energy graph for a 32-byte data packet. It took 81 ms and used 4.95 mJ to complete.

C. Power Efficiency

Another key feature of LoRa systems is the need to operate at low power for a long time period which is often measured in years [26]. This requires the system to utilize as little power as possible by working as quickly as possible to reduce the duty cycle required to read a sensor, process the data, secure the data and then transmit the secured data to the gateway. Utilizing higher power, but faster microprocessors is one way to further reduce the overall duty cycle. As shown in Fig. 12, the Adafruit Feather M0 processor uses 5.29 mA during heavy calculations and the ATECCX08A uses 16 mA during ECC command execution.

The energy usage was calculated by measuring the number of cycles needed by the microprocessor and the cryptographic coprocessor to run the same algorithm with the same key and message sizes. This time was then multiplied with the current measured to determine the total energy usage during the ECC algorithm cycle. The measured current was verified with the datasheets for both devices. For encryption, the power savings of utilizing the ATECCX08A is nearly a third of using software-based encryption while HMAC used one-twelfth of the energy.

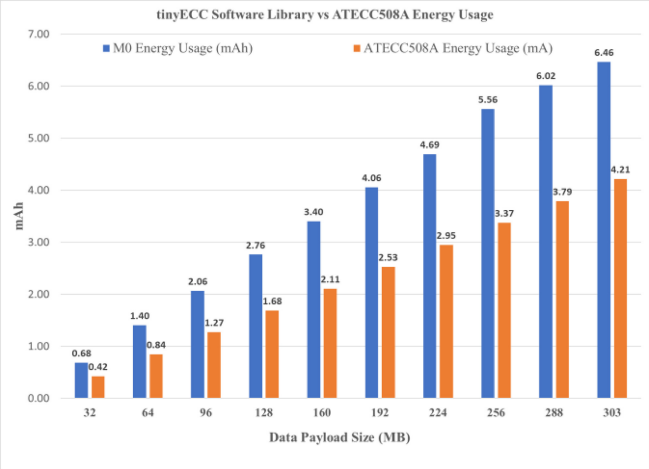


Fig. 12. Energy usage comparison between the tinyECC software library and ATECCX08A.

D. Hardware Integration

A further test of ATECCX08A was performed by integrating a pair of the Adafruit M0 LoRa radios with ATECCX08A to test a data transfer scenario. ATECCX08A easily connects to the Adafruit M0 over the I2C bus and has a very detailed and flexible library of APIs to manage the chipset. Appendices A and B show the state diagrams of the software flow. All libraries compiled and worked on the Adafruit M0 LoRa radios with a few small modifications specific to pin layout and M0 functionality. The M0 LoRa radios utilize the RadioHead RFM9x Library [27] and the ATECCX08A utilizes the SparkFun ATECCx08A Arduino Library [28]. The software examples were utilized with test messages to systematically verify that the ATECCX08A would work for the proposed LoRaWAN secure protocol.

VI. Conclusion and Future Work

It was found that utilizing a cryptographic hardware accelerator drastically increased the security and performance on a LoRa node utilizing our new, more secure protocol while also reducing power use and decreased processing time. It was found that the cryptographic hardware accelerator was nearly five times faster at encryption than the standard ECC software encryption libraries at one-twelfth the energy usage. The suggested protocol modifications for enhanced security only added 4 additional bytes to the overall message length, but added protection against six existing LoRa security vulnerabilities. The power savings of utilizing a crypto-coprocessor is an over one-third improvement of software-based encryption, thus increasing battery life for low-power nodes. The additional cost of a few tenths of a dollar, and the integration of a cryptographic hardware accelerator is well worth the additional investment for the added security for both LoRaWAN and our proposed protocols utilizing the LoRa Physical Layer. A cryptographic hardware accelerator would be a valuable addition to any LoRa end device system. However, the additional overhead of the larger encrypted packet size could cause additional latency and increase bandwidth needs so this solution may not be a fit for all LoRaWAN deployments and are best suited to those requiring better security and energy savings. The number of gateways would need to be possibly increased to handle the larger bandwidth requirements.

The test programs, evaluation boards, and libraries for the ATECCX08A were utilized to develop the protocols and structure of the system outlined in this article. The next steps would be to build a fully integrated LoRaWAN node and gateway by embedding the ATECCX08A/B chipset with a low-cost LoRaWAN enabled microcontroller and include on-chip AES encryption to the protocol. New dedicated APIs would be created to fully utilize the multiple secure key slots for multiple gateways and nodes as well as fully implement the new proposed LoRaWAN protocol outlined in this document.

References

[1] L. Vangelista, "Frequency shift chirp modulation: The LoRa modulation," *IEEE Signal Process. Lett.*, vol. 24, no. 12, pp. 1818–1821, Dec. 2017, doi: 10.1109/LSP.2017.2762960.

[2] N. Sornin and A. Yegin, "LoRaWAN® specification v1.1," LoRa Alliance, Fremont, CA, USA, Oct. 11, 2017.

[3] E. van Es, H. Vranken, and A. Hommersom, "Denial-of-service attacks on LoRaWAN," in *Proc. 13th Int. Conf. Availabil. Reliabil. Secur.*, New York, NY, USA, Aug. 2018, pp. 1–6, doi: 10.1145/3230833.3232804.

[4] F. Chantis, I. Stais, P. Calderon, E. Deirmentzoglou, and B. Woods, *Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things*. San Francisco, CA, USA: No Starch Press, 2021.

[5] K. O. Adefemi Alimi, K. Ouahada, A. M. Abu-Mahfouz, and S. Rimer, "A survey on the security of low power wide area networks: Threats, challenges, and potential solutions," *Sensors*, vol. 20, no. 20, p. 5800, Oct. 2020, doi: 10.3390/s20205800.

[6] S. Wesemeyer, I. Boureanu, Z. Smith, and H. Treharne, "Extensive security verification of the LoRaWAN key-establishment: Insecurities; patches," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Sep. 2020, pp. 425–444, doi: 10.1109/EuroSP48549.2020.00034.

[7] "Crypto chips and LoRaWAN | developer portal." Accessed: Feb. 12, 2022. [Online]. Available: https://lora-developers.semtech.com/learn/lorawan-academy/courses/crypto-chips-and-lorawan/

[8] E. Aras, G. S. Ramachandran, P. Lawrence, and D. Hughes, "Exploring the security vulnerabilities of LoRa," in *Proc. 3rd IEEE Int. Conf. Cybern. (CYBCONF)*, Jun. 2017, pp. 1–6, doi: 10.1109/CYBConf.2017.7985777.

[9] J. Thomas, S. Cherian, S. Chandran, and V. Pavithran, "Man in the middle attack mitigation in LoRaWAN," in *Proc. Int. Conf. Invent. Comput. Technol. (ICICT)*, Feb. 2020, pp. 353–358, doi: 10.1109/ICICT48043.2020.9112391.

[10] N. Yakin, M. Zhitkov, A. Chernikov, and P. Pepelyaev, "Security threats and service degradation detection in LoRaWAN networks," in *Proc. Ural Symp. Biomed. Eng. Radioelectron. Inf. Technol. (USBEREIT)*, May 2021, pp. 455–458, doi: 10.1109/USBEREIT51232.2021.9455123.

[11] S. J. Philip, J. M. McQuillan, and O. Adegbite, "LoRaWAN v1.1 security: Are we in the clear yet?" in *Proc. IEEE 6th Int. Conf. Dependabil. Sensor Cloud Big Data Syst. Appl. (DependSys)*, Dec. 2020, pp. 112–118, doi: 10.1109/DependSys51298.2020.00025.

[12] X. Yang, E. Karampatzakis, C. Doerr, and F. Kuipers, "Security vulnerabilities in LoRaWAN," in *Proc. IEEE/ACM 3rd Int. Conf. Internet Things Design Implement. (IoTDI)*, Apr. 2018, pp. 129–140, doi: 10.1109/IoTDI.2018.00022.

[13] O. Omolara, A. Jantan, O. Abiodun, and H. Arshad, "An enhanced practical difficulty of one-time pad algorithm for resolving the key management and distribution problem," in *Proc. Int. MultiConf. Eng. Comput. Sci.*, 2018, pp. 409–415.

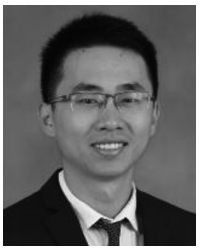
[14] P. Kietzmann, L. Boeckmann, L. Lanzieri, and T. C. Schmidt, "A performance study of crypto-hardware in the low-end IoT," in *Proc. EWSN*, 2021, pp. 79–90.

[15] "Espressif ESP32 Wi-Fi module with microchip ATECC608A secure element, EBV blog news." Feb. 24, 2020. Accessed: Nov. 14, 2022. [Online]. Available: https://blog.ebv.com/espressif-esp32-wi-fi-module-with-microchip-atecc608a-secure-element/

[16] N. A. Gunathilake, W. J. Buchanan, and R. Asif, "Next generation lightweight cryptography for smart IoT devices:: Implementation, challenges and applications," in *Proc. IEEE 5th World Forum Internet Things (WF-IoT)*, Apr. 2019, pp. 707–710, doi: 10.1109/WF-IoT.2019.8767250.
[17] Md. Shahjalal, Md. M. Islam, Md. M. Alam, and Y. M. Jang, "Implementation of a secure LoRaWAN system for Industrial Internet of Things integrated with IPFS and blockchain," *IEEE Syst. J.*, vol. 16, no. 4, pp. 5455–5464, Dec. 2022, doi: 10.1109/JSYST.2022.3174157.
[18] C. Zhang, J. Yue, L. Jiao, J. Shi, and S. Wang, "A novel physical layer encryption algorithm for LoRa," *IEEE Commun. Lett.*, vol. 25, no. 8, pp. 2512–2516, Aug. 2021, doi: 10.1109/LCOMM.2021.3078669.
[19] N. Hayati, K. Ramli, S. Windarta, and M. Suryanegara, "A novel secure root key updating scheme for LoRaWANs based on CTR_AES DRBG 128," *IEEE Access*, vol. 10, pp. 18807–18819, 2022, doi: 10.1109/ACCESS.2022.3150281.
[20] K.-L. Tsai, F.-Y. Leu, L.-L. Hung, and C.-Y. Ko, "Secure session key generation method for LoRaWAN servers," *IEEE Access*, vol. 8, pp. 54631–54640, 2020, doi: 10.1109/ACCESS.2020.2978100.
[21] "ATECC608B | microchip technology." Accessed: Oct. 9, 2021. [Online]. Available: https://www.microchip.com/en-us/product/atecc608b
[22] *ATECC508A Summary Data Sheet*, Microchip Technol., Chandler, AZ, USA, 2017.
[23] "Joulescope JS220: Precision energy analyzer, joulescope store." Accessed: Jul. 11, 2023. [Online]. Available: https://www.joulescope.com/products/js220-joulescope-precision-energy-analyzer
[24] K. MacKay. "micro-ECC." Nov. 23, 2021. Accessed: Nov. 24, 2021. [Online]. Available: https://github.com/kmackay/micro-ecc
[25] "Arduino cryptography library, operator foundation." Jul. 10, 2023. Accessed: Jul. 11, 2023. [Online]. Available: https://github.com/OperatorFoundation/Crypto
[26] M. Mabon, M. Gautier, B. Vrigneau, M. Le Gentil, and O. Berder, "The smaller the better: Designing solar energy harvesting sensor nodes for long-range monitoring," *Wireless Commun. Mobile Comput.*, vol. 2019, Jul. 2019, Art. no. e2878545, doi: 10.1155/2019/2878545.
[27] "RadioHead: RadioHead packet radio library for embedded microprocessors." Accessed: Dec. 1, 2021. [Online]. Available: http://www.airspayce.com/mikem/arduino/RadioHead/
[28] "SparkFun ATECCX08A arduino library, SparkFun electronics." Oct. 27, 2021. Accessed: Dec. 1, 2021. [Online]. Available: https://github.com/sparkfun/SparkFun_ATECCX08a_Arduino_Library

Steven Puckett (Member, IEEE) received the B.S. degree in physics from Auburn University, Auburn, AL, USA, in 1996, and the M.S. degree in electrical engineering from the University of Alabama at Birmingham, Birmingham, AL, USA, in 2003. He is currently pursuing the Ph.D. degree with the ECE Department, The University of Alabama in Huntsville, Huntsville, AL, USA.

He is the Community and Business Outreach Director and a Lecturer of Management with the Sanders College of Business and Technology, University of North Alabama, Florence, AL, USA.



Jianqing Liu (Member, IEEE) received the B.Eng. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2013, and the Ph.D. degree from The University of Florida, Gainesville, FL, USA, in 2018.

He is an Assistant Professor with the Department of Computer Science, North Carolina State University, Raleigh, NC, USA. His research interests are wireless communications and networking, security, and privacy.

Dr. Liu received the U.S. National Science Foundation Career Award in 2021. He is also the recipient of several Best Paper Awards, including the 2018 Best Journal Paper Award from the IEEE Technical Committee on Green Communications and Computing.

Seong-Moo Yoo (Life Senior Member, IEEE) received the B.A. degree in economics from Seoul National University, Seoul, South Korea, and the M.S. and Ph.D. degrees in computer science from the University of Texas at Arlington, Arlington, TX, USA.

He is a Professor Emeritus with the Electrical and Computer Engineering Department, University of Alabama in Huntsville (UAH), Huntsville, AL, USA. Before joining UAH, he was an Assistant Professor with Columbus State University, Columbus, GA, USA. He has coauthored over 130 scientific articles in refereed journals and international conferences.

Thomas H. Morris (Senior Member, IEEE) received the M.S. and Ph.D. degrees from Southern Methodist University, Dallas, TX, USA, in 2008 and 2001, respectively.

He is currently the Eminent Scholar of Computer Engineering, the Professor of Electrical and Computer Engineering, and the Director of the Center for Cybersecurity Research and Education with the University of Alabama in Huntsville, Huntsville, AL, USA.