Analysis of IoT Authentication Over LoRa

Derek Heeger
Department of Electrical and Computer Engineering
University of New Mexico
Albquerque, NM, USA
heegerds@unm.edu

Jim Plusquellic
Department of Electrical and Computer Engineering
University of New Mexico
Albquerque, NM, USA
jimp@ece.unm.edu

***Abstract*—Implementing a full set of security features within IoT devices is challenging because of constraints on the available resources and power consumption. Nevertheless, such devices must be capable of carrying out mutual authentication with gateways and servers before exchanging data. There are a wide variety of authentication methods that can be used including those based on physically unclonable functions (PUFs), PKI, encryption, and secure hash elements such as MD5 and SHA-3. This work assesses the time and energy associated with authentication protocols in the context of Long Range (LoRa), which is an emerging low-power wide-area network (LPWAN) technology used in IoT devices. LoRa has a set of configurable settings that affect the bandwidth and transmission range. We assess the energy performance of different authentication techniques over a variety of LoRa configurations and address the level of security provided by the authentication protocols. Our finding suggest that PUF-based authentication is well suited for RF devices operating within an energy and data rate constrained LoRa environment. We propose a PUF-based authentication protocol called PARCE that significantly reduces the RF transmissions for IoT devices.**

Keywords—LoRa, IoT, Authentication, HELP, PUF, PARCE

I. Introduction

The integration of IoT devices continues to expand into personal and industrial applications, including home automation, health care and agriculture. The resource constrained nature and unsupervised operating environment of IoT devices increases their vulnerability to adversarial attacks. Although it is possible to build in security techniques in this environment, there are many practical considerations that reduce their attractiveness. Characteristics such as battery life, ease of use, size, and cost weigh heavily into the constraints associated with IoT system design, and directly impact the footprint and computational complexity of acceptable security solutions.

Adversaries can engage in a wide variety of attacks including packet sniffing of transmitted information, denial of service (DoS), man-in-the-middle, password guessing, impersonation, etc. Security functions including authentication, encryption, and secure key exchanges are the primary defense mechanisms against attacks. Many techniques exist that implement these functions providing enhanced levels of security at the expense of additional computation and transmissions. Resource constrained embedded systems are often only able to leverage weaker security functions, making them less resilient to attacks.

In this paper, we focus on evaluating mutual authentication techniques because of the central role they play in establishing the authenticity of devices and servers (entities) at the onset of communications. Although not the focus of this work, authentication is also commonly used to validate messages, using message authentication codes (MAC). In either case, authentication leverages lower-level security primitives such as secure hash, PKI, encryption, and physically unclonable functions (PUFs) to accomplish these goals [1,2,3].

We evaluate authentication techniques in the context of a specific network technology. Many network protocols exist, each providing a specific transmission range, energy profile and data rate. Short range network communication technologies include Bluetooth, WiFi, and Zigbee, while longer range, low-power wide-area network (LPWAN) technologies include NB-IoT, LoRa, Sigfox, and LTE-M [4]. We use LoRa as the network technology for our evaluation.

LoRa is a closed source protocol using chirped spread spectrum (CSS) optimized for long range communications with low power. The LoRa physical layer is highly configurable and includes parameters such as spreading factor (SF), error coding rates, bandwidth, and header types. Devices that communicate using LoRa commonly use LoRa Wide Area Network (LoRaWAN) which is an open source protocol established by the LoRa alliance. The use of LoRaWAN allows for efficient communications to LoRa gateways. A LoRaWAN security protocol is assessed in [5] which we use as a benchmark for the evaluations carried out in this work.

Some applications use LoRa as the physical layer and create their own ad-hoc network. This type of approach is taken because the LoRaWAN specification cannot be optimized to meet all application needs. Applications that need bi-directional communications such as mesh networks and/or those that require in-field firmware updates are likely to adopt an ad-hoc approach.

This work evaluates the RF performance and energy consumption of authentication protocols within a LoRa network architecture, while considering practical implementation issues. The specific contributions of this work are summarized as follows:

- The time and energy overhead of secure hash-based, encryption-based and PUF-based authentication protocols are evaluated within LoRa networks configured with different sets of parameters.

- A comparative analysis is carried out against the LoRaWAN authentication standard (AES-CMAC) to determine which of these authentication methods is best suited for IoT applications.
- A novel protocol called PUF authentication for resource constrained environments (PARCE) is proposed and used in our evaluation. PARCE is based on the hardware embedded delay PUF (HELP) and is optimized for IoT networks.

II. Background

A. Related Work

To the best of our knowledge, no previous work exists on modeling the energy consumption associated with security functions within a LoRa network. There have been investigations on modeling power consumption within LoRa and LoRaWAN, but they are focused on power versus communication range [6, 7]. Other survey-based papers exist that evaluate IoT authentication and security techniques but the analysis is carried out at a high level omitting important lower-level details covered in this work [8]. The most closely related work is described in [9] but the focus is on 802.15.4.

Many PUF-based authentication protocols have been proposed since 2002 [3]. The HELP PUF was first proposed in [10] and then used with a novel authentication protocol in [11]. Reference [12] evaluates the entropy, while [13] demonstrates resilience to model building attacks. The application of HELP in an authentication protocol for RF devices with severely constrained data rates has not been assessed in prior work.

B. LoRa

The LoRa physical layer is highly configurable with parameters such as spreading factor (SF), error coding rates, bandwidth, preamble length, and header types [14]. The SF is the ratio of the symbols to chips with higher ratios increasing range and SNR. SF can be configured to a setting in the range between 6 and 12, with higher setting corresponding to increased power consumption. Error coding can be configured to incorporate between 25% to 100% redundant bits, with higher redundancy improving the reliability of packet delivery. Bandwidth can be configured from 10 kHz to 500 kHz with higher bandwidths increasing data rates. The preamble length can be configured to include between 6 and 65535 symbols, with higher values improving the synchronization with other devices. LoRa packets can be sent with an optional header that specifies the packet size and error coding rates.

C. Authentication Methods

The purpose of authentication is to verify the identity of an entity and to prevent impersonation by a malicious actor. A naïve form of authentication is to accept the claim to an identity in plain text form, i.e. "I am Bob". There is no evidence that Bob is who he claims he is and therefore, a malicious actor could easily make this same claim. A more secure form of authentication is based on Bob providing a password response to Alice, that requires Bob to have knowledge of a shared secret:

- Bob: "I am Bob"
- Alice: "What is your password?"
- Bob: "Password is security1sFun55"

The problem with this scheme is that the password can only be used once securely, otherwise Alice is subject to a replay attack. This occurs when a malicious actor listens in on the first exchange and provides Bob's password on subsequent authentications. The most secure approach is based on a challenge-response protocol, where each challenge-response is used only once. Such a scheme proves knowledge of a shared secret and prevents a replay attack:

- Bob: "I am Bob"
- Alice: "What is password 123141?"
- Bob: "Password is definitelySecureNow98"

A challenge-response form of authentication requires many shared secrets in practice or a one-way function that produces many responses to a single shared secret. The latter is provided by cryptographic primitives such as keyed secure hash functions. The former is possible if the device has a strong PUF which implies a large set of shared secrets.

There are two common mechanisms for establishing a shared secret(s) between Alice and Bob:

1) Through the use of public key infrastructure (PKI) and a trusted third party. This approach allows devices to authenticate without any prior knowledge of each-other.

2) Through a process called enrollment, where each device produces secrets that are recorded in a secure database. Here we assume that the enrollment process is carried out in a secure facility immediately following device manufacturing.

D. Mutual Authentication With Cryptography

Mutual authentication is a process whereby two devices authenticate each other. The steps involved in mutually authenticating enrolled entities proceeds as follows:

- Bob sends "Hello I am Bob" to Alice
- Alice sends back a nonce: "Prove it with challenge number 1234567"
- Bob combines the nonce with a shared secret to define a challenge and computes a response
- Bob transmits his response to Alice "Here's my proof"
- Alice carries out the same process with her copy of the shared secret and authenticates Bob if the two responses match.
- Bob authenticates Alice in a similar fashion, i.e., by sending Alice a nonce, which she combines with her shared secret to define a new challenge, and then computes and transmits her response back to Bob.
- Bob verifies her response with his own and mutually authenticates Alice if they match.

The encryption-based and secure hash mutual authentication schemes use a similar sequence of message exchanges. They differ primarily by how the challenge-response pair is constructed which will be addressed in the following sections.

E. *Secure Hash*

Authentication by secure hash generates the response from a challenge by concatenating the nonce with the secret and then performing a hash operation. A secure hash is a cryptographic primitive that transforms data through a one-way function into a digest. A one-way function implies that it is easy to compute the digest but extremely difficult to reverse the process, i.e., to compute the secure hash function input from the digest.

Common secure hash functions used for authentication include MD5, SHA-2, and SHA-3. MD5 was widely used in the late 1990's but later was found to possess security flaws and subsequently retired as a cryptographic function [15]. It is used today only to detect unintentional data corruption. SHA-2 was released in the early 2000's and is based on the Merkle-Damgard structure [16]. It replaced its predecessor, SHA-1, which was shown to have security vulnerabilities in 2017 [17]. Despite relying on the same mathematical principles, SHA-2 is still trusted and used in bitcoin, TLS, and SSL. The SHA-3 function is based on the Keccak algorithm which does not suffer from the same weaknesses as the Merkel-Damgard structure [18]. The SHA-2 and SHA-3 functions have been standardized to generate digests of lengths 224, 256, 384, or 512 bits.

Hash algorithms leverage standard logic operations including XOR, AND, and NOT operations and can be readily implemented in a microcontroller, FPGA or application-specific integrated circuit (ASIC). ASICs represent the fastest and most energy efficient implementations and are commercially available. For example, Maxim Integrated sells an ASIC that implements the SHA-3 algorithm [19].

F. *Encryption*

The response to the challenge using the encryption method is simply the nonce encrypted with the secret key. The advanced encryption standard (AES) is commonly used as the encryption algorithm, with either 128- or 256-bit keys [20]. This approach is referred to as AES-CMAC which can also be used to create message authentication codes (MACs). Encryption-based authentication can be implemented with other encryption techniques such as data encryption standard (DES) or Triple DES [20].

Encryption methods can be implemented in software or in hardware, e.g., ASICs. Some microcontrollers have built in AES modules. Alternatively, dedicated external AES hardware security modules can be added to the system architecture if desired.

III. PUF Based Authentication

A. *PUFs*

A wide variety of PUF architectures have been proposed, including the Ring Oscillator, SRAM, and Arbiter PUFs [21]. The terms strong PUF and weak PUF refer to distinct classes of PUFs characterized according to the number of unique bit strings they can generate. The number of bits produced by a strong PUF is a large exponential, typically greater than 2^{64}, which enables them to serve both authentication and encryption key generation roles. Weak PUFs can only generate a small number of fixed size bit strings, and therefore are restricted to encryption key generation functions.

There are two basic strategies that have been proposed for using PUFs in authentication protocols:

1) The PUF is used to generate an encryption key, which is used as the shared secret in the authentication techniques described previously in Section II. A strong or weak PUF is applicable here because the shared secret is not revealed outside the device and can therefore be reused in multiple authentication operations.

2) The PUF is used to generate the challenge-response pairs directly, without the need for a shared secret or cryptographic primitive. The PUF receives the challenge, and then computes and transmits the response. Given a new challenge-response pair is needed for every authentication operation, this strategy requires a strong PUF.

The second strategy provides a low power form of authentication, making the approach attractive for resource constrained devices running the LoRa protocol. In the following, we investigate this second challenge-response form of PUF based authentication in the context of a previously proposed strong PUF called HELP [10].

B. *HELP PUF*

The hardware-embedded delay PUF (HELP) is designed with a large source of entropy to enable it to generate an exponential number of challenge-response pairs. HELP leverages variations in propagation delay as a source of randomness. Tolerances in the manufacturing process cause delay to vary along logic paths within microprocessors, FPGAs and ASICs. HELP measures the delay of paths within a functional unit, e.g., a multiplier, using a high precision timing engine and then processes the digitized representations of those delays into a random, unique and reproducible bit string or key. HELP can accept challenges and produce responses directly within authentication protocols, i.e., without the need to obscure the interface with cryptographic primitives as is true for most other PUFs. This is only possible if the strong PUF is resilient to model building attacks (a second condition for a strong PUF). Preliminary work showing that HELP is resilient to model building is given in reference [13].

C. *PARCE Authentication*

We refer to the authentication protocol proposed here as PUF authentication for resource constrained environments or PARCE. Unlike the authentication protocol described in [11], the PARCE enrollment process and authentication protocol leverage a specialized key-encryption-key (KEK) mode of operation. KEK mode is normally used in secure boot processes, which require a stand-alone device to boot up securely without assistance from a remote server. To accomplish this, HELP utilizes challenges and helper data that

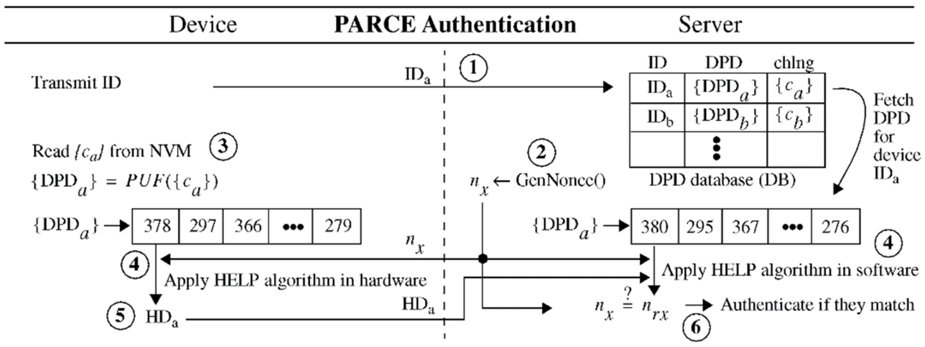

Fig 1.Message exchanges used for PARCE device authentication.

are stored locally on the device. During enrollment, the manufacturing facility applies the challenges to the PUF embedded within the device, and the PUF generates the secret key for the first time. The internally generated key remains on the device and is used to encrypt boot images. In order to enable the PUF to regenerate the same key in the field, the PUF also produces helper data during enrollment. This is stored in a non-volatile-memory (NVM) on the device, along with the challenges. These challenges and helper data are used to regenerate the decryption key, possibly under adverse environmental conditions. The helper data enables the key generation process to avoid bit flip errors adding resilience.

The KEK enrollment process proposed here for authentication differs in several ways from the process used for secure boot, and is shown graphically in Fig. 2. As discussed earlier, enrollment is carried out in a secure facility. A secure server shown on the right generates a set of challenge vectors $\{c_a\}$ that are then transmitted to device *a* on the left. Challenge vectors are used to produce transitions on specific logic paths in the functional unit, which is the source of entropy for HELP. The propagation delays of these transitions are measured and are referred to as digitized path delays or $\{DPD_a\}$. DPD are integer values that represent the relative delay of the path. Example DPD are shown on the left side of Fig. 2. The $\{DPD_a\}$ are transmitted to the manufacturer's server and stored in a database, along with the device ID and challenge vectors. The $\{DPD_a\}$ represent the 'shared secrets' and therefore must be stored securely.

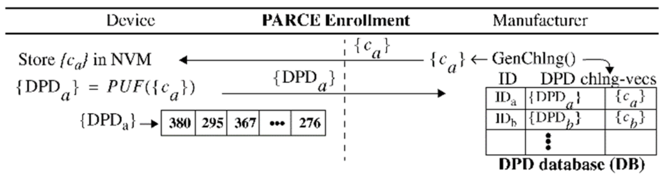

Fig 2: Message exchanges for PARCE enrollment process.

The PARCE protocol is shown in Fig. 1. Authentication begins with device *a* requesting authentication and transmitting its ID to the server (see circled '1' in the figure). The ID is looked up in the DPD database and the device timing data $\{DPD_a\}$ is retrieved. In step 2, the server generates an 8-byte nonce, n_x, and transmits it to the device. n_x is combined with the challenge vectors to define the complete challenge for HELP. In step 3, the device reads the challenge vectors $\{c_a\}$ from NVM and applies them to the PUF to reproduce $\{DPD_a\}$. Note that the digitized timing values are likely different than those stored in the database during enrollment by the server. Measurement noise, as well as changes in the temperature and supply voltage (referred to as TV noise), causes minor DPD variations.

In step 4, a hardware version of the HELP algorithm is run by the device, which processes the $\{DPD_a\}$ through a series of three mathematical operations, called *Diffs*, *TVComp* and *Modulus*. The overall effect of these operations is to first transform the $\{DPD_a\}$ such that delay variations introduced by adverse TV conditions are significantly reduced (*Diffs* and *TVComp*) and then to remove path length bias effects (*Modulus*) [11]. Each of these operations accepts input parameters that significantly expand the response bit string space. Previous work shows that for a given set of challenge vectors, the HELP algorithm parameters enable the $\{DPD_a\}$ to generate approximately 1 million unique bit strings for authentications. The 8-byte nonce n_x is used to specify the input parameters to the *Diffs*, *TVComp* and *Modulus* operations and represents the component of the challenge that changes from one authentication to the next. Note that an identical set of operations are carried out by the server in software as shown by step 4 on the right side of the figure.

The most important contribution of the PARCE protocol is related to the response bit string generation phase, which is identified as step 5 in Fig. 2. It is referred to in the following as secure key encoding (SKE). SKE is graphically depicted in Fig. 3 using a subset of the $\{DPD_a\}$ produced by the PUF, numbered from 1 to 26 along the top of the figure. The y-axis label DPD' represents the DPD after *Diffs*, *TVComp* and *Modulus* are applied. In this example, a modulus of 20 is applied, which wraps the DPD into the range between 0 and 20. The goal of the SKE process is to produce a helper data bit string (HD_a) that is small in size to conserve transmission power, e.g., 8 bytes, but also allows the server to securely authenticate the device. The SKE process incorporates reliability enhancement techniques to improve the probability of a successful authentication for genuine devices.

The graph in Fig. 3 illustrates the process of converting the DPD' into a helper data bit string and a response bit string. The solid horizontal lines at y-axis positions 0, 10 and 20 delineate

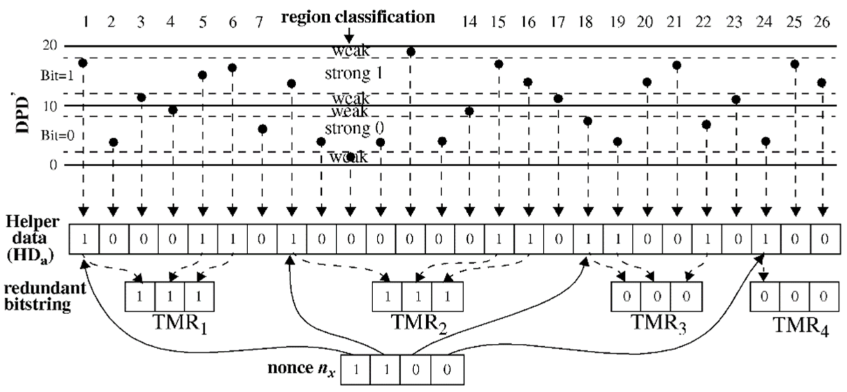

Fig 3: SKE process applied to subset of device {DPD}, with Modulus = 20 and Margin = 2.

the '0' and '1' bit assignment regions, with DPD' < 10 assigned '0' and those >= 10 assigned '1'. These lines are referred to as the bit flip lines because regenerated DPD' that differ from the enrollment values will be assigned different bit values by the device and server if they appear on opposite sides of these lines.

SKE implements two reliability enhancement techniques called margining and TMR to prevent bit flip errors in the device regenerated response. Margining uses a margin parameter to classify the DPD' within the '0' and '1' regions as strong or weak bits. Weak bits are defined as DPD' that are within the margin of a bit flip line. From Fig. 3, the margin is represented as horizontal dotted lines at y-axis positions 2, 8, 12 and 18. DPD' classified as weak are deemed unreliable and are not used in the bit string generation process, i.e., they are skipped. A helper data bit-string is constructed to identify the classification status of the DPD'. DPD' that fall within the weak regions are assigned '0' in the helper data bit-string. For example, weak bits occur at positions 3 and 4 (among others) in Fig. 3. The TMR technique is layered on top of the margining scheme, and leverages redundancy as a means of improving reliability. The scheme shown in Fig. 3 uses TMR or triple modular redundancy, but the method can be extended to any odd integer value >= 3. TMR is a popular fault tolerance technique applied to high reliability computing systems. The underlying principle of TMR is to use 3 consecutive strong bits to encode one response bit, as described in the following.

Step 5 from Fig. 2 indicates that the response transmitted to the server by the PARCE protocol is the helper data bit string, designated as HD_a. The process shown along the bottom of Fig. 3 illustrates how the helper data and redundant bit strings are created. A 4-bit portion of the nonce n_x is shown along the bottom of Fig. 3 and represents the target bit string that SKE will encode. Encoding is accomplished by processing the n_x bits one bit at-a-time, starting with the left-most '1' bit. SKE parses the DHD' constructing the helper data bit-string such that 1) no weak bits are used in the response bit string and 2) three consecutive strong bits of the same value are selected. The left-most DPD' is classified as a strong '1', so a '1' is inserted into the HD_a at this position. The DHD' at position 2 is a strong '0' but it cannot be used because of its value (SKE encoding mismatch) and therefore, a '0' is inserted into the HD_a at position 2. The DPD' at positions 3 and 4 are weak, and are also excluded via margining. The DHD' at positions 5 and 6 are both strong '1's and therefore qualify under the SKE criteria. The first bit of n_x is now fully encoded. The process then begins again but targets the second bit of n_x which is also a '1'. Here, the strong '0' at position 7 is skipped because of a SKE encoding mismatch, while the strong '1' at position 8 qualifies and is assigned a '1' in HD_a. The DHD' at positions 8 through 14 are skipped. The encoding of the second n_x bit completes with the DHD' at positions 15 and 16.

The SKE encoding process continues until the HD_a exceeds a minimum size threshold, e.g. 10-bytes or 80-bits. As discussed earlier, the process of generating the helper data and response bit string for the first time is called enrollment. Therefore, the device carries out enrollment in the PARCE protocol while the server carries out regeneration. Once the server receives the HD_a, it applies a similar process except the HD_a is used to dictate which DPD' are selected to construct the response bit string, labeled as n_{rx} in Fig. 2. A majority voting scheme is applied to the redundant bit string for cases in which the 3 bits in each group do not agree in value. In other words, a mismatch occurs for bits in the n_x and n_{rx} bit strings only when two consecutive redundant bits in the 3-bit groups have bit flip errors. The margining and TMR schemes add significant resilience to bit flip errors. Authentication is deemed successful in step 6 if n_x matches n_{rx}. PARCE authentication can be applied in both directions to achieve mutual authentication.

IV. Packet Modeling

A framework for modeling LoRa interactions was developed and validated in [22] and is applied here for authentication. The time to transmit a single packet is defined in Equation 1.

$$T_{payload} = (8 + \text{ceil}[\frac{8PL-4SF+28+16CRC-20IH}{4(SF-2DE)}](CR + 4)\ )T_s \quad (1)$$

The parameters of this equation represent LoRa configuration options which are described in detail in [14]. Our modeling and analysis in the following focuses on transmit time. Transmit time is implementation independent in contrast to energy consumption because energy consumption depends on the specific power amplifier, operating environment, and antenna used in an application. Transmission energy consumption is proportional to transmit time and can be computed by multiplying the transmit time by transmit power.

A. *Authentication Packets*

This work models the energy usage of LoRa authentication using minimal packet overhead. There are four packets exchanged by any generic authentication protocol:

1. Request to authenticate
2. Challenge
3. Response
4. Acknowledgement

The challenge and response packet length depend on the authentication method being used. The request to authenticate and acknowledgement are 4-byte data packets. It is noted that an actual application might choose to include source, destination, and sequence numbers in packet transmissions but this additional overhead is not being considered here.

V. Transmission Time Analysis

The authentication methods described earlier are modeled and analyzed over a variety of LoRa settings. The configuration parameters used in the analysis and plots are defined in Table 1 unless otherwise noted. The results are presented as a function of transmit time instead of power consumption, recognizing that transmission power can be computed for specific implementations as needed. Note that the computation time of operations associated with the authentication method are negligible compared to the RF transmission time so they are ignored in our analysis.

Table 1: The default settings used for modeling.

Setting	***Value***
Spreading Factor	9 or 512 (chips/symbol)
Bandwidth	62.5 kHz
Implicit Header	Implicit Header On
Preamble Length	6

A. *Secure Hash*

The transit times associated with various secure hash authentication methods are plotted in Fig. 4 as a function of the LoRa spreading factor (SF) configuration parameter. MD5 is the most efficient since it only requires a 56-bit packet size but, as discussed earlier, is not secure. SHA-2 and SHA-3 transmit the same number of bits so they have equivalent transmission times. As expected, the larger digests associated with high security standards increase transmission times. An interesting observation of this is that SHA-224 and SHA-256 take the exact same time to authenticate with SF set to 11. This is due to the ceiling function constraint in the LoRa transmission specification so their packet sizes are rounded up (binned) to the same packet length. This binning effect was validated in [22] and introduces minor non-linearities in the analysis presented in the following.

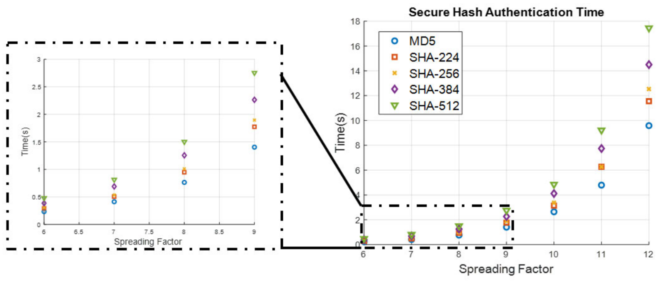

Fig 4: The transmit time versus spreading factor for secure hash techniques.

B. *Encryption*

An analysis of the encryption-based authentication transmit times are plotted in Fig. 5 as a function of LoRa SF. AES-56 is obsolete but is provided for reference. Note the relative spacing of AES-192 and AES-256 for SF=11 and SF=12. Intuitively, one would expect their transmission times to be proportional to the number of bits transmitted but that is not actually the case. Given this discrepancy, it would be inefficient to select AES-192 with SF=11 because the transmission time of AES-256 adds only 50ms or 0.8% to transmission time. AES-192 is very efficient when SF=12 because AES-256 duration is 2s longer in this case, which is approximately a 20% increase. Note that these observations are dependent on the packet structure described in the previous section but the comparison is similar for other packet options.

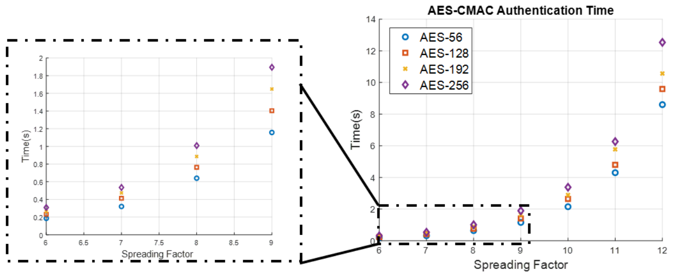

Fig 5: The transmit time versus spreading factor for AES-CMAC.

C. *PUF*

As noted earlier, the size of the response, HD_a is configurable in the PARCE protocol so our analysis considers several possibilities. Note that PARCE is not susceptible to model building because the strong bit string is not transmitted, only the helper data. Moreover, the sequence of '1's in the transmitted HD_a is directly coupled to the strong bit string, and therefore inherits its statistical quality characteristics. In previous work, the strong bit strings were shown to be cryptographic quality [12]. Therefore, a 64-bit HD_a response is theoretically as secure as the other authentication methods of equivalent length. The transit times associated with PARCE are the same as the corresponding bit lengths shown in the previous section for AES.

D. *LoRa Parameter Comparisons*

This section extends the comparative analysis of authentication protocols SHA-3, AES-CMAC, and PARCE over multiple LoRa settings. The analysis is carried out using an 80-bit PARCE response (HELP-80), a 128-bit version of AES-CMAC (note, this is the standard for LoRaWAN uses), and the smallest SHA-3 response of 224 bits. PARCE is more efficient than the other two methods because of the smaller packet size. Fig. 6 compares the transmit times using bandwidths from 50 kHz to 500 kHz. For LoRa, it should be noted that increasing the bandwidth decreases transmission times but also decreases the effective communication range.

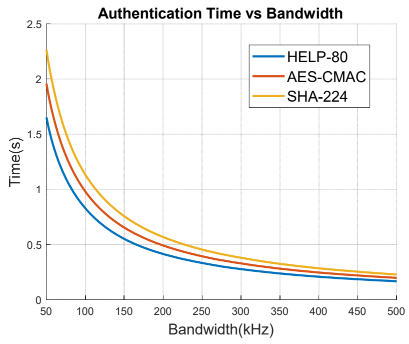

Fig 6: The authentication time versus the bandwidth.

Fig 7 illustrates the differences in error coding rates for schemes that send 25% to 100% extra bits. Transmission times are linearly related to increases in error coding rate.

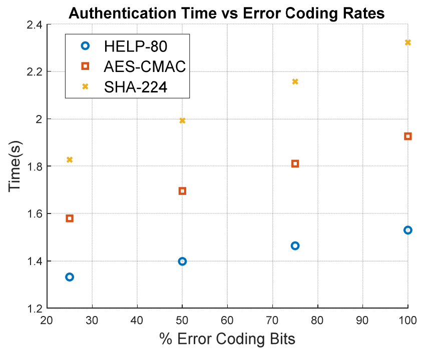

Fig 7: The authentication time versus the error coding rate.

Fig. 8 shows the effect on transmission time when the LoRa implicit header feature is varied. As expected, schemes that do not transmit the header saves between 20-30% of the transmission time and energy. Note that omitting the transmission header limits the flexibility in the system because the packet length and error correction settings are fixed for all transmissions.

E. Single Ended Authentication

All previous analyses compare the transmit time for mutual authentication, but single ended authentication is sufficient for some applications. Fig 9 shows the transmit time relationship between performing single and mutual authentication as a function of the spreading factor plotted along the x-axis. Note that mutual authentication adds approximately 25% to the transmit time over single ended authentication.

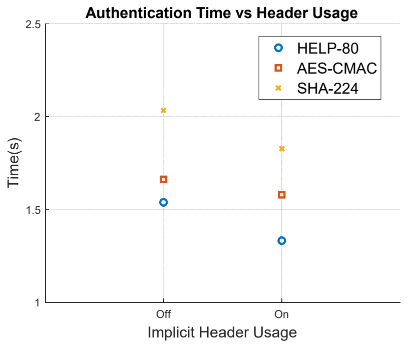

Fig 8: The authentication time when using the implicit header.

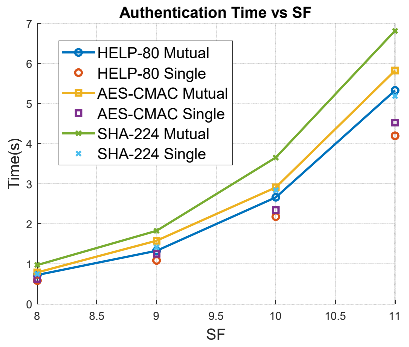

Fig 9: Single and mutual authentication time compared to the spreading factor.

VI. Other Considerations

A. Enrollment Issues

Each of the authentication protocols has different implementation requirements. For protocols that leverage cryptographic primitives, a key must be installed into the device during manufacturing. Common NVM storage mechanisms include flash and battery-backed RAM. The device specific stored key must be recorded during enrollment in a secure database to enable fielded authentication. As discussed, PARCE leverages a strong PUF and requires the challenge vectors to be stored in NVM. An alternative is to eliminate the stored challenges and instead receive them at the onset of authentication from the server, at the cost of significant transmission time and energy. Therefore, strong PUFs offer a trade-off of either including costly NVM on the device or eliminating it.

B. Device Security Issues

The level of security provided by the authentication protocols vary, with some more vulnerable to attacks then others. The cryptographic methods rely on NVM which makes them vulnerable to invasive physical attacks [23]. Secure NVM increases system costs further and cannot be used for lost cost IoT device applications. Battery backed random access memory (RAM) is an alternative, but requires periodic battery maintenance. PUFs offer a distinct advantage in this regard because the secret is no longer stored digitally in an NVM or on-chip. Moreover, a PUF is tamper-evident, i.e., they are easily destroyed by invasive probing attacks, further improving their attack resilience in the field.

For IoT devices, side-channel attacks are also of increasing concern, where adversaries attempt to steal internal secrets while the device carries out cryptographic operations by measuring and analyzing power transients or electromagnetic emissions. Protocols that utilize fixed-size, permanent keys are more vulnerable to side-channel attacks than the PARCE protocol where repeated use of a stored secret does not occur [24].

C. LoRaWAN Authentication

The LoRaWAN standard uses the AES-CMAC to authenticate with 128-bit AES. This approach proves to be superior to secure hash techniques from an RF performance perspective. In addition, computing AES in embedded devices is convenient when using LoRa because the LoRa specific processors include an AES module [25]. LoRa processors do not have built-in SHA modules therefore an external IC is needed or the function would need to be computed in software.

For IoT devices that transmit sensitive information, control sensitive equipment, or require extremely low power consumption, the PARCE protocol is a better alternative because it provides high levels of security, provides resistance to invasive attacks and is energy efficient with respect to internal computation and RF transmission. Although HELP is currently implemented only on FPGAs, which are not suitable for low cost IoT applications, it is possible to build HELP onto a LoRa specific device or dedicated ASIC, making PARCE applicable to a wider range of applications, i.e., beyond medical and defense applications that can afford the cost of an FPGA.

VII. Conclusions

This paper investigates authentication protocols based on secure hash, encryption and PUFs for the use in LoRa networks. An analysis over LoRa configuration parameters is performed to investigate the transmit time which correlates to energy consumption. A novel PARCE protocol is proposed for low cost mutual authentication within resource constrained IoT environments. PARCE leverages a strong PUF called the hardware-embedded delay PUF (HELP) as a source of entropy, and proposes to carry out authentication using helper data bit strings, instead of PUF generated response bit strings. The size of the transmitted helper data can be configured, allowing trade-offs between energy efficiency and the level of security. Future work will use hardware testbeds implemented with these IoT authentication protocols as a means of validating the simulation results presented in this paper.

References

[1] L. Chi and X. Zhu, “Hashing Techniques,” *ACM Computing Surveys*, vol. 50, no. 1, pp. 1–36, 2017.

[2] R. Poovendran, J. Lee, and T. Iwata, “The AES-CMAC Algorithm,” 2006.

[3] G. E. Suh and S. Devadas, “Physical Unclonable Functions for Device Authentication and Secret Key Generation,” *2007 44th ACM/IEEE Design Automation Conference*, 2007.

[4] K. Mekki, E. Bajic, F. Chaxel, F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment", ICT Exp., vol. 5, no. 1, pp. 1-7, Mar. 2019

[5] M. Eldefrawy, I. Butun, N. Pereira, and M. Gidlund, “Formal security analysis of LoRaWAN,” *Computer Networks*, vol. 148, pp. 328–339, 2019.

[6] T. Bouguera, J.-F. Diouris, J.-J. Chaillout, R. Jaouadi, and G. Andrieux, “Energy Consumption Model for Sensor Nodes Based on LoRa and LoRaWAN,” Sensors, vol. 18, no. 7, p. 2104, Jun. 2018

[7] L. Casals, B. Mir, R. Vidal, and C. Gomez, “Modeling the Energy Performance of LoRaWAN,” Sensors, vol. 17, no. 10, p. 2364, Oct. 2017

[8] M. El-Hajj, A. Fadlallah, M. Chamoun, and A. Serhrouchni, “A Survey of Internet of Things (IoT) Authentication Schemes,” *Sensors*, vol. 19, no. 5, p. 1141, Jun. 2019.

[9] D. Altolini, V. Lakkundi, N. Bui, C. Tapparello and M. Rossi, "Low power link layer security for IoT: Implementation and performance analysis," *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, Sardinia, 2013, pp. 919-925.

[10] J. Aarestad, P. Ortiz, D. Acharyya and J. Plusquellic, HELP: A Hardware-Embedded Delay-Based PUF, Design and Test of Computers, Mar., 2013, pp. 17-25

[11] W. Che, M. Martin, G. Pocklassery, V. K. Kajuluri, F. Saqib and J. Plusquellic, A Privacy-Preserving, Mutual PUF-Based Authentication Protocol, Cryptography, Vol. 1, Issue 1, 2016.

[12] W. Che, F. Saqib and J. Plusquellic, Novel Offset Techniques for Improving Bitstring Quality of a Hardware-Embedded Delay PUF, Trans. on VLSI, 2017.

[13] W. Che, M. Martinez-Ramon, F. Saqib and J. Plusquellic, Delay Model and Machine Learning Exploration of a Hardware-Embedded Delay PUF, Hardware-Oriented Security and Trust, 2018.

[14] Semtech, “SX1276/77/78/79 - 137 MHz to 1020 MHz Low Power Long Range Transceiver”, SX1276/77/78/79 datasheet, August 2016

[15] *MD5 considered harmful today*. [Online]. Available: https://www.win.tue.nl/hashclash/rogue-ca/. [Accessed: 26-Apr-2020].

[16] R.C. Merkle. Secrecy, authentication, and public key systems. Stanford Ph.D. thesis 1979, pages 13-15.

[17] “SHAttered,” *SHAttered*. [Online]. Available: https://shattered.it/. [Accessed: 26-Apr-2020].

[18] J. R. C. Cruz and J. R. C. Cruz, “Keccak: The New SHA-3 Encryption Standard,” [Online]. Available: https://www.drdobbs.com/security/keccak-the-new-sha-3-encryption-standard/240154037. [Accessed: 26-Apr-2020].

[19] Maxim Integrated, “DS28E16: 1-Wire Secure SHA-3 Authenticator”, Jan. 2020

[20] N. Aleisa, “A Comparison of the 3DES and AES Encryption Standards,” *International Journal of Security and Its Applications*, vol. 9, no. 7, pp. 241–246, 2015.

[21] T. Mcgrath, I. E. Bagci, Z. M. Wang, U. Roedig, and R. J. Young, “A PUF taxonomy,” *Applied Physics Reviews*, vol. 6, no. 1, p. 011303, 2019.

[22] D. Heeger, M. Garigan, J. Plusquellic, “Adaptive Data Rate Techniques for Energy Constrained Ad Hoc LoRa Networks,” IN PRESS, 2020 Global IoT Summit

[23] S. Skorobogatov, “Flash Memory ’Bumping’ Attacks”, Cryptographic Hardware and Embedded Systems, 2010

[24] O. Lo, W. J. Buchanan, and D. Carson, “Power analysis attacks on the AES-128 S-box using differential power analysis (DPA) and correlation power analysis (CPA),” Journal of Cyber Security Technology, vol. 1, no. 2, pp. 88–107, 2016.

[25] Microchip, “SAM R34/R35 Low Power LoRa® Sub-GHz SiP Datasheet”, 2019.