Scientia Iranica D (2018) 25(6), 3442–3460

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
http://scientiairanica.sharif.edu

Practical provably-secure authenticated encryption schemes using lattice-based pseudorandom function SPRING

A. Boorghany[a], S. Bayat-Sarmadi[b], and R. Jalili[a,*]

a. *Department of Computer Engineering, Data and Network Security Lab (DNSL), Sharif University of Technology, Tehran, Iran.*
b. *Hardware Security and Trust (HST) Lab, Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.*

Received 29 December 2016; received in revised form 8 April 2018; accepted 29 September 2018

KEYWORDS
Authenticated encryption;
Lattice-based cryptography;
Post-quantum cryptography;
Provable security.

Abstract. Lattice-based cryptography has received significant attention from security practitioners in the past decade. It exhibits attractive properties, including being a major post-quantum cryptography candidate, enjoying worst-case to average-case security reductions, and being supported by efficient implementations. In this paper, we propose three practical lattice-based Authenticated Encryption (AE) schemes. These schemes are provably secure assuming hardness of basic lattice problems. The proposed schemes have remarkable motivations and advantages over widely-used AEs as follows. These schemes are alternatives to current conventional and post-quantum AE schemes in the post-quantum era. Moreover, composing the proposed AEs with a lattice-based asymmetric key distribution scheme results in a hybrid encryption, which depends only on one (type of) security assumption. The implementation of such hybrid encryption can make use of specific optimizations regarding, e.g., code size in software, and gate equivalent or FPGA area usage in hardware. That is because the symmetric and asymmetric algorithms have some common primitive computations. To evaluate the performance of the proposed AEs, we implement them on current Intel CPUs and benchmark them to encrypt messages of various sizes. The most efficient proposed scheme is only 12% slower than AES-256-GCM for 40-byte messages on Sandy Bridge, and 34% faster for 1500-byte messages.

1. Introduction

Lattice-based cryptography is one of the main candidates for post-quantum cryptography [1]. Cryptographic schemes based on hard lattice problems are conjectured to resist against attacks by a large-scale quantum computer [2], while the schemes based on the hardness of integer factorization, integer discrete logarithm, and elliptic curve discrete logarithm are completely vulnerable in this setting [3,4]. Other advantages of the lattice-based schemes, in comparison to widely-used ones based on number theory (such as RSA and ECDSA), are as follows. Most of the state-of-the-art lattice-based schemes enjoy a worst-case to average-case security reduction (e.g., [5-7]). Thus, random instances of these schemes (i.e., with random keys and/or parameters) are provably as hard as *worst-case* instances of basic lattice problems. Finally, computations in lattice-based schemes usually deal with simple operations on small integer vectors. In doing so, engineers can design efficient implementations on hardware and software, and take more advantage from hardware parallelism, processor pipelines, mul-

*. *Corresponding author.*
E-mail addresses: boorghany@ce.sharif.edu (A. Boorghany); sbayat@sharif.edu (S. Bayat-Sarmadi); jalili@sharif.edu (R. Jalili)

doi: 10.24200/sci.2018.21044

tiple cores, and the single-instruction multiple-data (SIMD) feature. As a result, modern implementations of the lattice-based schemes are much more efficient than the traditional alternatives [7-12].

Another line of research focuses on Authenticated Encryption (AE). AE is a symmetric two-in-one solution for secure communications. It provides both confidentiality and authenticity of the encrypted message. AE has recently received significant attention in the academy and industry. The CAESAR competition [13], first started in 2013, is ongoing to select the most secure, applicable, and robust AEs. AE has many applications in data and network security. For instance, AE can provide confidentiality and data-origin authentication of network packets. This is also standardized in the Encapsulating Security Payload (ESP) protocol of the IPSec standard. RFC 7231 [14] proposes AES-GCM [15] and AES-CCM [16] as two AEs suitable for ESP.

A traditional approach to building an AE is to compose a simple (privacy-only) symmetric encryption scheme and a Message Authentication Code (MAC). This approach is called generic composition. For instance, AES-CCM [16] is the composition of AES block cipher in Counter mode of operation, and CBC-MAC message authentication code. Another widely-used generically-composed AE scheme is AES-GCM [15], which uses GMAC message authentication code instead. Note that composing two encryption and MAC schemes should be done very carefully, as there are many subtleties to build a secure composed scheme [17]. MAC schemes that make use of a block cipher as the building block are good candidates to be employed in generic composition. For instance, CBC-MAC [16], TMAC [18], OMAC [19], and PMAC [20] apply the block cipher encryption on each input block, while GMAC [15] performs a GF(2^{128}) multiplication on each input block, and XORs achieve the output of one block-cipher encryption to mask the multiplications result. Schemes built by the generic composition technique are categorized as two-pass AE schemes. This type of AE processes the input message twice (e.g., invoking the block cipher encryption twice per input block) in order to output the ciphertext and authentication tag. The counterpart AE type is a single-pass AE. IAMP [21], OCB [22-24], and OTR [25] are some well-known single-pass AE schemes; however, all of them are protected by a few patents (or patents pending).

The final related work for now is the effort on a lattice-based pseudorandom function (PRF). Banerjee et al. [26] introduced a PRF with a security proof based on lattice problems. PRFs are widely used in the design of symmetric cryptosystems. Subsequently, Banerjee et al. [27] proposed an efficient instantiation of this PRF called SPRING. They used SPRING in the counter (CTR) mode of operation to build a lattice-based symmetric encryption. Their report of the SPRING implementation shows that, using the SIMD feature in high-end processors, the SPRING performance is interestingly comparable to the performance of AES.

In this paper, three authenticated encryption schemes are introduced with security proofs based on the worst-case hardness of lattice problems. The security theorems and proofs of these schemes are exact or concrete. That is, the proved security bound exactly (not asymptotically) determines the maximum success probability of the attacker (To be precise, this bound is usually relative to the security bound of an underlying primitive construction). This approach in provable security is called exact or concrete security, in contrast to the traditional asymptotic security. It is introduced in [28-30], which is referred to as the paradigm of practice-oriented provable security. Almost all provably-secure lattice-based schemes in the literature are supported by an asymptotic security proof, though.

The proposed schemes, referred to as LAE1, LAE2, and LAE3, use the lattice-based pseudorandom function SPRING as the building block. The proposed AEs based on SPRING have the following particular advantages over the previous AE constructions.

- Firstly, the proposed schemes are alternatives to the conventional AE schemes based on a block cipher (e.g., AES) in the post-quantum era. The best-known quantum attack to current AE schemes is a corollary of the work of Grover [31]. In this attack, any generic symmetric encryption with an n-bit key can be broken in time $O(2^{n/2})$ using a quantum computer. Thus, a traditional AE that uses AES-256 has 128 bits of post-quantum security. Furthermore, recent work [32,33] proposes quantum attacks to some symmetric cryptosystems; however, they use superposition queries, which assume a very strong attack model. Instead of traditional AEs, the proposed lattice-based schemes are supported by a security proof against quantum attackers, which may help if a better quantum algorithm is discovered to break AES or other AEs based on a block-cipher.
- Secondly, in most applications, such as IPSec or Transport Layer Security (TLS), an AE is utilized within a form of hybrid encryption. Hybrid encryption makes use of an asymmetric scheme to distribute a symmetric session key. After that, the session key is provided to the AE scheme to encrypt bulk data. The asymmetric schemes used in hybrid encryption include Public-Key Encryption (PKE), Key Encapsulation/transport Mechanism (KEM), and Authenticated Key Exchange (AKE). By using a traditional number-theoretic asymmetric scheme (e.g., based on RSA, Diffie-Hellman, or Elliptic curve Diffie-Hellman – DH and ECDH), the entire system becomes vulnerable to a quantum attack.

Thus, for a quantum-resistant hybrid encryption, a post-quantum variant of the asymmetric part is required. Integrating the proposed AEs with a lattice-based PKE, KEM, or AKE (e.g., the ones proposed in [34]) has an important advantage that the resulting hybrid encryption depends only on one (type of) security assumption. Otherwise, the security of the system is being compromised if there is a flaw either in the assumption of the AE part or in the assumption of the asymmetric scheme used for the key distribution.

- Thirdly, utilizing lattice-based schemes in both the AE part and the asymmetric part of a hybrid encryption has efficiency benefits, especially for hardware level implementations. The Fast Fourier Transform (FFT) over finite rings is a major computation task of the proposed AEs and most asymmetric schemes based on (ideal) lattices. Considerable amount of silicon area or FPGA slices can be saved if the common computations are refactored. Therefore, the proposed AEs should be advantageous in post-quantum cryptographic (co-)processors.
- Fourthly, as presented in Section 5, the performance of the proposed AE schemes is comparable with that of conventional provably-secure AEs. For instance, one of the proposed schemes, LAE2, has a competitive performance in IP packet encryption. Assuming a small packet to be 40 bytes and a large packet to be 1500 bytes, it is only 12% and 24% slower than AES-256-GCM for small IP packets on Sandy Bridge and Haswell CPUs, respectively. On the other hand, it is 34% and 15% faster than AES-256-GCM for large IP packets, respectively, on the targeted microarchitectures.

The proposed schemes have also some drawbacks over traditional AEs based on block ciphers. These constructions are designed over ideal lattices [35]. An ideal lattice has more algebraic structure than a general one. This algebraic structure helps design more efficient cryptographic schemes, both in terms of speed and key size. However, the hardness assumption of lattice problems over ideal lattices is stronger. Specifically, SPRING's security proof relies on the hardness of the Ring Learning With Errors (Ring-LWE) problem [35], which is the version of LWE problem over an ideal lattice. Another issue is the very large sized key of the proposed schemes. This is detailed below and in Section 5 along with the implementation results. One solution to the issue of large keys is to utilize a pseudorandom number generator (PRNG). The PRNG is seeded with a short key to generate the larger key of each lattice-based AE.

The first proposed AE scheme is a nonce-based Encrypt-and-MAC (E&M) composition. On the encryption part, SPRING is used in a nonce-based variant of CTR mode. The authentication part, which ensures the authenticity of both nonce and message, is a CBC-MAC similar to XCBC [36], TMAC [16], and OMAC [19]. There are some challenges to adopt SPRING to be used in such cryptographic schemes. Most of these schemes are designed to use a (strong) pseudorandom permutation (PRP), which is the formal model of a block cipher. Some schemes, such as OCB mode of operation [24], are fundamentally dependent on the bijection property of PRPs and cannot directly accept a PRF (Note that a PRP can be built from any PRF by a Feistel-like construction [37]; however, it makes performance worse). Another challenge to utilizing SPRING in cryptosystems is that the SPRING input and output sizes are not the same. It accepts 128 bits as input and provides 127 bits as output. That introduces a number of padding and truncation operations, complicating the security proof. Moreover, the output is not a multiple of bytes and causes some obstacles in the implementation (see Section 5). Another challenge affecting the design of a scheme based on SPRING is that most significant optimizations can be achieved only by a certain configuration of SPRINGs (see Section 3.2).

The second proposed AE scheme is an Encrypt-then-MAC (EtM) composition. The encryption part of the second scheme is similar to the previous one; however, the use of SPRING in the authentication part is minimized. Both of these AEs are two-pass, i.e., they are as inefficient as running encryption and MAC separately.

The third proposed scheme is designed to be single-pass. Until recently, all single-pass AEs could not essentially use a PRF instead of a PRP. For instance, IACBC, IAPM [21], OCB variants [22-24], and stateful XECB and ECBC [38] cannot accept a PRF. The design of the third scheme is inspired by OTR [25]. Minematsu [25] proposed two variants of OTR, which accept both PRF and PRP. However, the third scheme cannot directly follow the PRF-compatible version of OTR because it requires a Variable-Input-Length PRF (VILPRF), while the SPRING's input length is fixed. In this scheme, a tweakable PRF is created using SPRING.

The implementation results of the Intel Sandy Bridge and Haswell microarchitectures show that the second proposed scheme, LAE2, is efficient enough to be used in practice and compete widely-used AEs. On Sandy Bridge, although this scheme is 12% slower than AES-256-GCM for encrypting 40-byte messages, it becomes faster for the messages of length 64 bytes or longer. Particularly, for a 1500-byte message, this scheme is 34% faster than AES-256-GCM. A similar result is also achieved on Haswell. LAE2 is 24% slower for 40-byte messages in comparison with AES-256-GCM; however, it becomes faster for 100-byte messages and longer. Additionally, it is 15% faster

than AES-256-GCM for 1500-byte messages. The main comparison is performed with AES-256-GCM because the security of this scheme is 128 bits in the post-quantum setting. The performance comparison with AES-128-GCM is also presented in Section 5. Similar to most lattice-based cryptographic schemes, the key size of these schemes is very large. The size of the keys for LAE2 and LAE3 is around 6 KB, and the key size of LAE1 is around 12 KB.

For the sake of simplicity, we have not considered common complementary AE features in the three proposed AE schemes, such as the support of associated data (AEADs) [39], nonce misuse-resistance [40], and online environment requirements.

1.1. Organization

Section 2 introduces the preliminary cryptographic definitions and notations used in this paper. The proposed two-pass AE schemes and the single-pass one are described in Sections 3 and 4, respectively. These sections also include the design rationale, security theorems, and proofs of the constructions. Section 5 is devoted to the implementation of the proposed schemes. The efficiency and performance results are presented in this section. Finally, Section 6 concludes the paper.

2. Preliminaries

2.1. Notations

$\text{msb}_\ell(x)$ is the string consisting of the ℓ most significant bits of x. $\text{pad}_n(x)$ is the n-bit string obtained by concatenating a 1 and optionally a few 0's to the right of string x. The notation $\xleftarrow{n}$ is the operator that splits the right-hand string into a list of n-bit strings, except the last one, which may be shorter. The concatenation of two bit strings a and b is denoted by $a\|b$. By $x \xleftarrow{\$} \mathcal{S}$, we mean a uniform sampling of x from finite set $\mathcal{S}$. $\text{Func}(n,\ell)$ is the set of all functions from n-bit to ℓ-bit strings. Similarly, $\text{Perm}(\ell)$ is the set of all permutation functions from ℓ-bit to ℓ-bit strings. A family of functions $F = \{F_K\}$ is a set of functions indexed by the key K. $\mathcal{A}^{\mathcal{O}_K(\cdot,\cdot)}$ denotes a machine $\mathcal{A}$ with oracle access to $\mathcal{O}_K$. Oracle $\mathcal{O}_K(\cdot,\cdot)$ has two inputs under the control of $\mathcal{A}$. However, she cannot access the the parameter K embedded inside.

Polynomial additions and multiplications in $\text{GF}(2^{128})$ and $\text{GF}(2^{127})$ are performed modulo $x^{128} + x^{127} + x^{126} + x^{121} + 1$ and $x^{127} + x^{126} + 1$, respectively, which are typical irreducible polynomials. The elements of $\text{GF}(2^n)$ may be represented in three ways and be interchanged frequently. An n-bit string $Y = y_{n-1}...y_1y_0$ is equivalent to the polynomial $y_{n-1}x^{n-1} + ... + y_1x + y_0$, and is also equivalent to the number obtained by base-2 interpretation of Y. For instance, $2Y$, $3Y$, and $4Y$ are the multiplication of polynomials x, $x+1$, and x^2, respectively, with the polynomial equivalent to string Y. Multiplication of a small constant with a field element can be performed very efficiently using a few shift and XOR operations.

The success probability or advantage of an adversary $\mathcal{A}$ is denoted by $\text{Adv}^{\text{G}}_{S[\mathcal{F}]}(\mathcal{A})$, where S is the cryptographic scheme being attacked by $\mathcal{A}$, $\mathcal{F}$ is the (family of the) building block used in S, and G is the security model or security game in which $\mathcal{A}$ is modeled. $\$(\cdot)$ and $\$(\cdot,\cdot)$ are oracles that, on each query, return a fresh random bit string of specified length.

2.2. Pseudorandom functions and message authentication codes

For the integers $n \geq \ell \geq 1$, a family of functions $F = \left\{F_K : \{0,1\}^n \to \{0,1\}^\ell\right\}$ is a pseudorandom function (PRF) if for any adversary $\mathcal{A}$, with ordinary number of queries and resources, the following advantage is small [41, Definition 2.1]:

$$\text{Adv}^{\text{prf}}_F(\mathcal{A}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{F_K(\cdot)} \Rightarrow 1] - \Pr[\rho \xleftarrow{\$} \text{Func}(n,\ell) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1]. \quad (1)$$

A pseudorandom permutation (PRP) is the family $P = \left\{P_K : \{0,1\}^\ell \to \{0,1\}^\ell\right\}$ of one-to-one and onto functions, for which the security is defined similarly to the PRF with the following advantage [41, Definition 2.2]:

$$\text{Adv}^{\text{prp}}_P(\mathcal{A}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{P_K(\cdot)} \Rightarrow 1] - \Pr[\rho \xleftarrow{\$} \text{Perm}(\ell) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1]. \quad (2)$$

A variable-input-length pseudorandom function (VILPRF) family $F = \left\{F_K : \{0,1\}^* \to \{0,1\}^\ell\right\}$ is an special PRF in which the input can be a string of arbitrary length [42, Section 3.1].

A message authentication code (MAC) $\mathcal{MA} = (\mathcal{K}, \mathcal{T}, \mathcal{V})$ consists of a randomized key generation algorithm $\mathcal{K}$, and two functions $\mathcal{T}_K(N,M) : \{0,1\}^w \times \{0,1\}^* \to \{0,1\}^\tau$ and $\mathcal{V}_K(N,M,T) : \{0,1\}^w \times \{0,1\}^* \times \{0,1\}^\tau \to \{\texttt{true}, \texttt{false}\}$ for message tagging and tag verification, respectively. Unforgeability under chosen message attack (uf-cma) is the model to define the security of $\mathcal{MA}$. This scheme is a secure MAC if, for any adversary $\mathcal{A}$, with ordinary number of queries and resources, the following advantage is small:

$$\text{Adv}^{\text{uf-cma}}_{\mathcal{MA}}(\mathcal{A}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{T}_K(.,.),\mathcal{V}_K(.,.,.)}\text{forges}]. \quad (3)$$

A forgery is successful when $\mathcal{A}$ makes a query to $\mathcal{V}_K$ with two conditions. Firstly, inputs N, M, and T given to $\mathcal{V}_K$ have not appeared before in any query-response pair of $\mathcal{T}_K$. Secondly, $\mathcal{V}_K$ does not return `false` on the query [41, Definition 2.6].

2.3. Authenticated encryption

An authenticated encryption scheme $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is a tuple of three algorithms. $\mathcal{K}$ is the key generation algorithm. The encryption function $\mathcal{E}(K, N, M) : \{0,1\}^k \times \{0,1\}^\omega \times \{0,1\}^* \rightarrow \{0,1\}^* \times \{0,1\}^\tau$ receives the key, nonce, and message, and returns the ciphertext and tag (C, T). The decryption function $\mathcal{D}(K, N, C, T) : \{0,1\}^k \times \{0,1\}^\omega \times \{0,1\}^* \times \{0,1\}^\tau \rightarrow \{0,1\}^* \cup \{\perp\}$ performs the inverse functionality and returns the plaintext. $\mathcal{D}$ may also return $\perp$ in the case of an error. $\mathcal{E}_K(N, M)$ and $\mathcal{D}_K(N, C, T)$ are the same functions, considering the key as a parameter. The referred AEs in this paper are nonce-based. Therefore, an extra parameter N is given to $\mathcal{E}$ and $\mathcal{D}$ as the nonce.

$\mathcal{AE}$ should have two security properties: privacy and authenticity. Formally, a secure $\mathcal{AE}$ should provide the indistinguishability under chosen plaintext attack (ind-cpa) and maintain the integrity of the ciphertext (int-ctxt). No nonce value N should be repeated in the inputs of encryption function $\mathcal{E}$. Otherwise, the privacy or authenticity may not be guaranteed. Note that a repeated nonce in the inputs of $\mathcal{D}$, or between the inputs of $\mathcal{E}$ and $\mathcal{D}$, is fine. Without loss of generality, we can assume that all adversaries are nonce-respecting, i.e., they never send the same nonce value to $\mathcal{E}$. The privacy advantage of adversary $\mathcal{A}$ is defined as follows:

$$\mathrm{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_K(\cdot,\cdot)} = 1] - \Pr[\mathcal{A}^{\$(\cdot,\cdot)} = 1]. \quad (4)$$

$\$(\cdot,\cdot)$ is an oracle which returns a fresh random string of length $|\mathcal{E}_K(\cdot,\cdot)|$. Note that $\$(\cdot,\cdot)$ is not queried twice on an exact input, as $\mathcal{A}$ is nonce-respecting.

The authenticity advantage of an adversary $\mathcal{A}$ is defined as follows:

$$\mathrm{Adv}^{\text{int- ctxt}}_{\mathcal{AE}}(\mathcal{A}) = \Pr[K \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathcal{E}_K(\cdot,\cdot),\mathcal{D}_K(\cdot,\cdot,\cdot)} \text{forges}]. \quad (5)$$

A forgery is successful when $\mathcal{A}$ makes a query to $\mathcal{D}_K$ with two conditions. Firstly, inputs N, C, and T given to $\mathcal{D}_K$ have not appeared before in any query-response pair of $\mathcal{E}_K$. Secondly, $\mathcal{D}_K$ does not return $\perp$ on the query. If both $\mathrm{Adv}^{\text{ind-cpa}}_{\mathcal{AE}}(\mathcal{A})$ and $\mathrm{Adv}^{\text{int-ctxt}}_{\mathcal{AE}}(\mathcal{A})$ are small respecting any adversary $\mathcal{A}$, with ordinary number of queries and resources, then $\mathcal{AE}$ is a secure authenticated encryption scheme [22, Section 5.1].

2.4. SPRING pseudorandom function

The SPRING pseudorandom function, presented by $\mathrm{SPR}_K : \{0,1\}^n \rightarrow \{0,1\}^\ell$ in this paper, is the SPRING-CRT function in the work of Banerjee et al. [27]. This function is defined as follows:

$$\mathrm{SPR}_K(x_1 \ldots x_n) = \mathrm{MSBs}\left(a \cdot \prod_{j=1}^{n} s_j^{x_j}\right) \quad (6)$$

In this equation, the polynomial $a \in \mathcal{R}_p^n = \mathbb{Z}_p[x]/(x^n + 1)$ is a fixed uniformly-random polynomial and is a parameter of the function. Moreover, $K = \{s_1, \ldots, s_n\}$ is the function key, in which each polynomial s_j is in the $\mathcal{R}_p^n$ ring. All multiplications are also performed in $\mathcal{R}_p^n$. The function $\mathrm{MSBs} : \mathcal{R}_p^n \rightarrow \{0,1\}^\ell$ is an encoding function that concatenates the most significant bit of each coefficient to make an n-bit string. Then, by truncating the last bit, we obtain a pseudorandom output of length $\ell = n - 1$ bits.

3. Two-pass lattice-based authenticated encryption

3.1. Using a CBC-MAC variant

The first proposed authenticated encryption scheme, referred to as LAE1, is illustrated in Figure 1. It is obtained by an Encrypt-and-MAC (E&M) composition of a nonce-based symmetric encryption scheme and a Message Authentication Code (MAC). The encryption part is built using SPRING in the CTR mode, in which the indices given to the SPRINGs have two segments. The first segment is the nonce N, which is fixed for all SPRINGs of the encryption part. The second segment is the block number encoded with a Gray code. The input and output of SPRING are n and ℓ bits, respectively. The length of the nonce is fixed to w bits. The SPRING functions of the encryption part use K_1 as the key. The authentication part is a variant of CBC-MAC [41]. The concatenation of the nonce and message will be fed into a CBC-chain of SPRINGs. The SPRINGs in the authentication part use a different key K_2 in order to ensure a safe composition. In the processing of the last block, another parameter is XORed with the input of SPRING, which depends on the third short key K_3. The multiplication of 2 by K_3, in the full-last-block case, is performed in $\mathrm{GF}(2^\ell)$ using at most one shift and one XOR. The encryption procedure of LAE1 is shown in Scheme 1. The decryption procedure is easily obtained by XORing the given ciphertext C with $X[i]$ to obtain M^* and computing the expected tag T^* from M^* in the forward direction. If T^* is the same as the given tag T, then the successfully decrypted message M^* is returned. Otherwise, only a $\perp$ is returned.

Design rationale The authentication part of LAE1 is a stand-alone MAC based on SPRING as the underlying primitive. It is similar to TMAC [18] with a few differences. TMAC uses a block cipher instead of a PRF. In addition, TMAC does not have a parameter τ to adjust the tag length. The input and output lengths of the underlying function in TMAC (i.e., the block cipher) are the same, while the SPRING output is shorter than its input. There are some alternatives

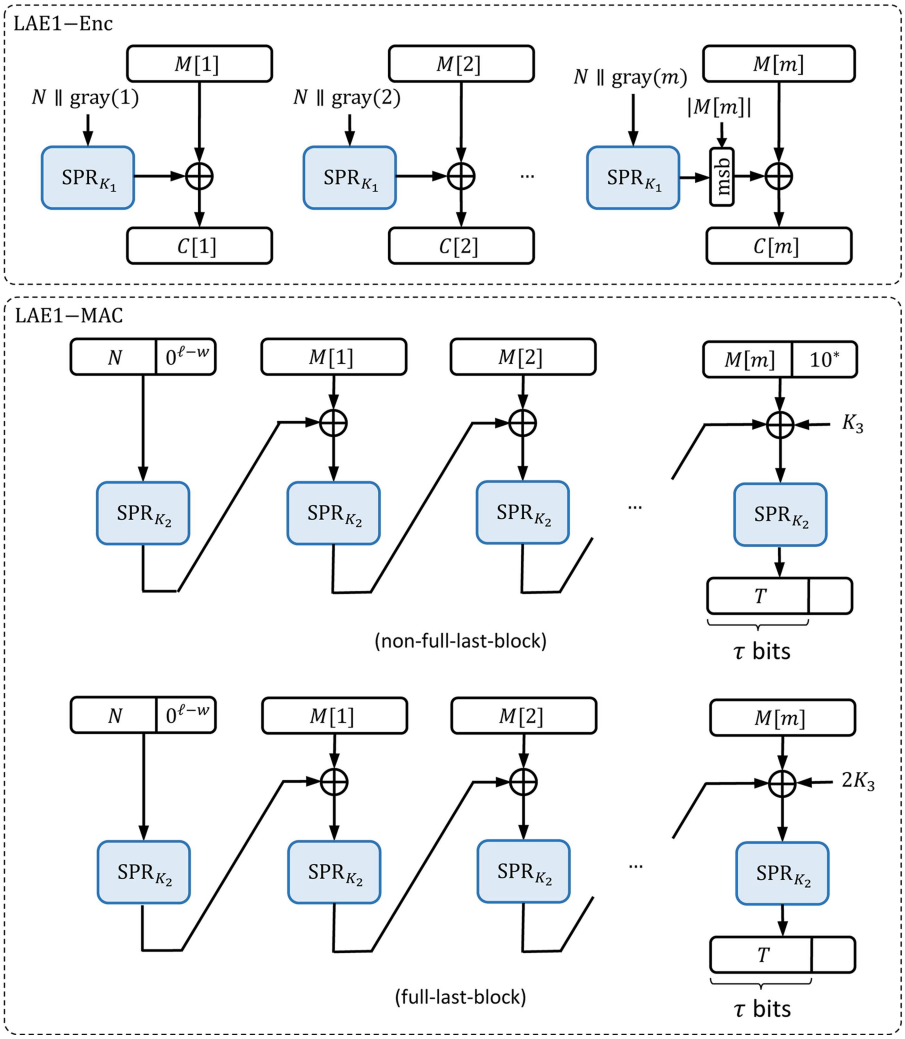

Figure 1. The flow diagram of LAE1 authenticated encryption procedure; top is the encryption part (LAE1-Enc), and bottom is the authentication part (LAE1-MAC), which is divided into non-full-last-block and full-last-block modes.

procedure AUTHENCRYPT(N, M)
  $(M[1], ..., M[m]) \xleftarrow{\ell} M$
  // LAE1-Enc
  for $i \leftarrow 1$ **to** $m-1$ **do**
    $Z[i] \leftarrow \text{SPR}_{K_1}(N \| \text{gray}(i))$
    $C[i] \leftarrow M[i] \oplus Z[i]$
  $Z[m] \leftarrow \text{msb}_{|M[m]|}\left[\text{SPR}_{K_1}(N \| \text{gray}(m))\right]$
  $C[m] \leftarrow M[m] \oplus Z[m]$
  $C \leftarrow C[1] \| ... \| C[m]$
  // LAE1-MAC
  $X[0] \leftarrow N \| 0^{\ell-w}$
  $Y[0] \leftarrow \text{SPR}_{K_2}(X[0] \| 0^{n-\ell})$
  for $i \leftarrow 1$ **to** $m-1$ **do**
    $X[i] \leftarrow M[i] \oplus Y[i-1]$
    $Y[i] \leftarrow \text{SPR}_{K_2}(X[i] \| 0^{n-\ell})$
  if $|M[m]| = \ell$ **then**
    $X[m] \leftarrow M[m] \oplus X[m-1] \oplus 2K_3$
  else
    $X[m] \leftarrow \text{pad}_\ell(M[m]) \oplus X[m-1] \oplus K_3$
  $Y[m] \leftarrow \text{SPR}_{K_2}(X[m] \| 0^{n-\ell})$
  $T \leftarrow \text{msb}_\tau(Y[m])$
  return (C, T)

Scheme 1. LAE1.

to be used as a base to construct a SPRING-based MAC. A structure similar to OMAC [19] may seem to be advantageous as it requires only one key for the authentication (LAE1 uses two keys K_2, and K_3 for this purpose); however, it needs an extra invocation of SPRING which is relatively heavy. Moreover, the length of extra key K_3 is n bits, which is much smaller than the size of SPRING keys K_1 and K_2 (see Section 5 for the efficiency results). Similar to [27], to achieve a better performance, the SPRING input in the encryption part is designed to be a Gray-code counter, consisting of a fixed part N and the Gray-code encoding of the block number.

A SPRING-based parallel MAC can be built using a structure similar to PMAC [20]. This makes the authenticated encryption scheme parallel (the encryption part is already parallel). However, the overall performance is not so different from LAE1, as the SPRING inputs are again non-Gray-code. As mentioned before, the type of the composition used in LAE1 is Encrypt-

and-MAC (E&M). Instead of the Encrypt-then-MAC (EtM) technique in which the ciphertext is given to the MAC algorithm, E&M allows parallel processing of the ciphertext and tag. To build an AE using the EtM generic composition, Bellare and Namprempre [43] claimed that performing a MAC only on the ciphertext was sufficient. However, Namprempre et al. [17] showed that their results could not be applied directly to the once-based authenticated encryption. Thus, we cannot save one SPRING invocation using the EtM technique.

Namprempre et al. [17] proposed secure E&M generic compositions. LAE1 has some differences with the proposed constructions in [17]. Mainly, the authors of this paper insisted on the use of a single key and derived subsequent keys via a PRF. However, LAE1 has three different keys for the sake of running time efficiency.

3.1.1. Security of LAE1

Theorems 1 and 2 show the security of LAE1.

Theorem 1 (privacy of LAE1)

Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \to \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{A}$ to attack the privacy of $LAE1[\mathcal{F}]$, who runs in time t and asks q oracle queries with a maximum length of m blocks for each query, there exists an adversary $\mathcal{P}$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{ind-cpa}}_{LAE1[\mathcal{F}]}(\mathcal{A}) \leq 2\mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{(4m^2+1)q^2}{2^{\ell+1}} + \frac{q^2}{2^{n+1}}. \quad (7)$$

Moreover, adversary P asks $q' = 2\sigma + q$ oracle queries and runs in time $t' = 2t + \sigma t_F + \alpha n(\sigma + q)$, where t_F is the time to compute F and α is a constant depending on the model of the computation.

Theorem 2 (authenticity of LAE1)

Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \to \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{A}$ to attack the authenticity of $LAE1[\mathcal{F}]$, who runs in time t and asks q_e encryption and q_d decryption queries, there exists an adversary $\mathcal{P}$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{int-ctxt}}_{LAE1[\mathcal{F}]}(\mathcal{A}) \leq \mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{(4m^2+1)q^2+2}{2^{\ell+1}} + \frac{q^2}{2^{n+1}} + \frac{q_d}{2^\tau}, \quad (8)$$

where $q = q_e + q_d$. Moreover, adversary $\mathcal{P}$ asks $q' = \sigma + q$ oracle queries and runs in time $t' = t + \sigma t_F + \alpha n(\sigma + q)$, where t_F is the time to compute F and α is a constant depending on the model of the computation.

The proofs of Theorems 1 and 2 are presented in Section 3.1.2.

3.1.2. Security proofs

To prove Theorem 1, the following lemmas are required.

Lemma 1 (privacy of LAE1-Enc)

Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \to \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{B}$ to attack the privacy of $LAE1\text{-}Enc[\mathcal{F}]$, who asks q queries with total message length of σ blocks, there exists an adversary $\mathcal{P}_1$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{ind-cpa}}_{LAE1-Enc[\mathcal{F}]}(\mathcal{B}) \leq \mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}_1). \quad (9)$$

Moreover, adversary $\mathcal{P}_1$ asks $q' = \sigma q$ oracle queries, and runs in time $t' = t + \alpha n(\sigma + q)$, where α is a constant depending on the model of the computation.

The proof of Lemma 1 is derived from the security proof of [44, Theorem 13].

Lemma 2 (ideal LAE1-MAC is pseudorandom)

Let $\mathcal{R} = \mathrm{Func}(n, \ell)$. For any adversary $\mathcal{A}$ asking q queries, each with at most m blocks, we have:

$$\mathrm{Adv}^{\text{vilprf}}_{LAE1-MAC[\mathcal{R}]}(\mathcal{A}) \leq \frac{(4m^2+1)q^2}{2^{\ell+1}} + \frac{q^2}{2^{n+1}}. \quad (10)$$

Note that this result is independent of the computational resources of $\mathcal{A}$.

Proof of Lemma 2

Suppose that $\mathcal{T}$ is the MAC part of the LAE1 scheme. We introduce *fFCBC* as a variant of *FCBC* [36], which uses pseudorandom functions. $fFCBC[F_1, F_2, F_3]$ is a CBC-MAC which calls F_1 on the intermediate blocks and calls either F_2 or F_3 on the last block, depending on being a full or partial block. By the definition of the VILPRF advantage, we have:

$$\begin{aligned}\mathrm{Adv}^{\text{vilprf}}_{LAE1-MAC[\mathcal{R}]}(\mathcal{A}) = &\Pr[\rho \xleftarrow{\$} \mathrm{Func}(n,\ell), K_3 \xleftarrow{\$} \{0,1\}^n : \\ &\mathcal{A}^{\mathcal{T}[\rho,K_3](\cdot)} = 1] - \Pr[\rho_1, \rho_2, \rho_3 \xleftarrow{\$} \mathrm{Func}(n,\ell) : \\ &\mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} = 1] + \Pr[\rho_1, \rho_2, \rho_3 \xleftarrow{\$} \mathrm{Func}(n,\ell) : \\ &\mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} = 1] - \Pr[\rho \xleftarrow{\$} \mathrm{Func}(*,\ell) : \\ &\mathcal{A}^{\rho(\cdot)} = 1]. \end{aligned} \quad (11)$$

We now consider each of these two pairs of probabilities. The second pair is the information-theoretic security of *fFCBC*, which is analyzed in Section 3.1.3 and bounded as follows:

$$\mathrm{Adv}^{\mathrm{vilprf}}_{fFCBC[\rho_1,\rho_2,\rho_3]}(\mathcal{A}) \leq \frac{(4m^2+1)q^2}{2^{\ell+1}}. \quad (12)$$

Note that $\mathcal{T}[\rho, K_3](\cdot)$ is equivalent to $fFCBC[\rho(\cdot), \rho(\cdot \oplus K_3), \rho(\cdot \oplus 2K_3)]$. Thus, the first pair of probabilities can be written as follows:

$$\Pr[\rho \xleftarrow{\$} \mathrm{Func}(n,\ell), K_3 \xleftarrow{\$} \{0,1\}^n : \mathcal{B}^{\rho(\cdot),\rho(\cdot\oplus K_3),}_{\rho(\cdot\oplus 2K_3)} = 1]$$

$$-\Pr[\rho_1,\rho_2,\rho_3 \xleftarrow{\$} \mathrm{Func}(n,\ell) : \mathcal{B}^{\rho_1(\cdot),\rho_2(\cdot),\rho_3(\cdot)} = 1], \quad (13)$$

where $\mathcal{B}$ simply simulates $\mathcal{A}$ to distinguish the two sets of oracles. Computing the advantage of such adversary is straightforward and is bounded by $q^2/2^{n+1}$.□

Lemma 3 (LAE1-MAC is a VILPRF)

Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \to \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{D}$ to attack the pseudorandomness of *LAE1-MAC*$[\mathcal{F}]$, there exists an adversary $\mathcal{P}_2$ against pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\mathrm{vilprf}}_{LAE1-MAC[\mathcal{F}]}(\mathcal{D}) \leq \mathrm{Adv}^{\mathrm{prf}}_{\mathcal{F}}(\mathcal{P}_2) + \frac{(4m^2+1)q^2}{2^{\ell+1}} + \frac{q^2}{2^{n+1}}. \quad (14)$$

Moreover, adversary $\mathcal{P}_2$ asks $q' = (\sigma+q)q$ oracle queries and runs in time $t' = t+\alpha n(\sigma+q)$, where α is a constant depending on the model of the computation.

Lemma 3 is the complexity-theoretic counterpart of Lemma 2, which can be proven in a standard way (for example, see [41, section 3.2]).

Proof of Theorem 1

Suppose that $\mathcal{E}$ and $\mathcal{T}$ are the Enc and MAC parts of LAE1, respectively. We derive two adversaries $\mathcal{B}$ and $\mathcal{D}$ from $\mathcal{A}$. Both simulate an ind-cpa game for $\mathcal{A}$. Meanwhile, $\mathcal{B}$ tries to break the privacy of LAE1-Enc, and $\mathcal{D}$ attempts to distinguish LAE1-MAC from a random function. $\mathcal{B}$ forwards $\mathcal{A}$'s request to its own oracle to obtain C. Then, it generates a fresh random T and outputs (C,T). Furthermore, $\mathcal{D}$ generates a random key K_1. Then, $\mathcal{D}$ itself computes C and invokes its oracle to obtain T. Thus, we have:

$$\mathrm{Adv}^{\text{ind-cpa}}_{LAE1[\mathcal{F}]}(\mathcal{A}) == \Pr[\mathcal{A}^{\mathcal{E}(\cdot,\cdot),\mathcal{T}(\cdot,\cdot)} \Rightarrow 1]$$

$$-\Pr[\mathcal{A}^{\$(\cdot,\cdot),\mathcal{T}(\cdot,\cdot)} \Rightarrow 1] + \Pr[\mathcal{A}^{\$(\cdot,\cdot),\mathcal{T}(\cdot,\cdot)} \Rightarrow 1]$$

$$-\Pr[\mathcal{A}^{\$(\cdot,\cdot),\$(\cdot,\cdot)} \Rightarrow 1] = \mathrm{Adv}^{\text{ind-cpa}}_{LAE1-Enc[\mathcal{F}]}(\mathcal{B})$$

$$+\mathrm{Adv}^{\mathrm{vilprf}}_{LAE1-MAC[\mathcal{F}]}(\mathcal{D}) \leq$$

$$2\mathrm{Adv}^{\mathrm{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{(4m^2+1)q^2}{2^{\ell+1}} + \frac{q^2}{2^{n+1}}, \quad (15)$$

where $\mathcal{P}$ can be either $\mathcal{P}_1$ or $\mathcal{P}_2$ which has more advantage. Additionally, the last inequality is provided by Lemmas 1 and 3.□

Proof of Theorem 2

Based on the pseudorandomness of LAE1-MAC (Lemma 3), it is standard to prove the following equation about the unforgeability of LAE1-MAC under chosen message attack:

$$\mathrm{Adv}^{\text{uf-cma}}_{LAE1-MAC[\mathcal{F}]}(\mathcal{B}) \leq \mathrm{Adv}^{\mathrm{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{(4m^2+1)q^2+2}{2^{\ell+1}} + \frac{q^2}{2^{n+1}}. \quad (16)$$

Now, adversary $\mathcal{A}$ can be utilized to build adversary $\mathcal{B}$. This adversary acts as follows. For any encryption queries $\mathcal{A}$ makes, $\mathcal{B}$ uses its oracle to obtain T. Then, it generates a random key K_1 to compute C. Moreover, upon a decryption query from $\mathcal{A}$, it decrypts C to obtain M^* and sends (N, M^*, T) to its forgery (decryption) oracle. As a result, we simply have:

$$\mathrm{Adv}^{\text{int-ctxt}}_{LAE1[\mathcal{F}]}(\mathcal{A}) \leq \mathrm{Adv}^{\text{uf-cma}}_{LAE1-MAC[\mathcal{F}]}(\mathcal{B}), \quad (17)$$

and the theorem is proved.□

3.1.3. Security of fFCBC

In this section, the information-theoretic security of *fFCBC* is presented. At first, suppose that *fCBC* is the plain CBC mode, instantiated with a PRF function. The collision probability of *fCBC* is defined as follows:

$$\mathrm{Col}_{n,\ell}(m,m') = \max_{M\neq M'} \Big\{\Pr\big[\rho \xleftarrow{\$} Func(n,\ell) : fCBC_\rho(M) = fCBC_\rho(M')\big]\Big\}, \quad (18)$$

where messages M, M' are chosen from $\{0,1\}^{m\ell}$ and $\{0,1\}^{m'\ell}$, respectively.

Lemma 4 (collision resistance of fCBC)

Fix $m, m' \geq 1$. The following bound holds for the *fCBC* collision probability:

$$\mathrm{Col}_{n,\ell}(m,m') \leq \frac{(m+m')^2+1}{2^\ell}. \quad (19)$$

Proof of Lemma 4

Fix two messages M, M', where $|M| = m\ell$, $|M'| = m'\ell$, and $M \neq M'$. Moreover, assume that the first k blocks of M and M' are equal (k may be zero). Now, Game 1 presents the computation of two *fCBC* functions. The collision occurs when $Y_m = Y'_{m'}$.

Consider the case that the bad flag is never set to `true`. Then, all Y_i and Y'_i variables in lines 13 and 19 of Game 1 are set to uniform and independent values.

1: bad $\leftarrow$ `false`; Dom(ρ) $\leftarrow \phi$
2: **for** $i \leftarrow 1$ **to** k **do**
3: **if** $i = 1$ **then** $X_i \leftarrow X'_i \leftarrow M_1$
4: **else** $X_i \leftarrow X'_i \leftarrow Y_{i-1} \oplus M_i$
5: **if** $X_i \in$ Dom(ρ) **then** bad $\leftarrow$ `true`
6: **else** $\rho(X_i) \xleftarrow{\$} \{0,1\}^\ell$
7: $Y_i \leftarrow Y'_i \leftarrow \rho(X_i)$
8: **for** $i \leftarrow k+1$ **to** m **do**
9: **if** $i = 1$ **then** $X_i \leftarrow M_1$
10: **else** $X_i \leftarrow Y_{i-1} \oplus M_i$
11: **if** $X_i \in$ Dom(ρ) **then** bad $\leftarrow$ `true`
12: **else** $\rho(X_i) \xleftarrow{\$} \{0,1\}^\ell$
13: $Y_i \leftarrow \rho(X_i)$
14: **for** $i \leftarrow k+1$ **to** m' **do**
15: **if** $i = 1$ **then** $X'_i \leftarrow M'_1$
16: **else** $X'_i \leftarrow Y'_{i-1} \oplus M'_i$
17: **if** $X'_i \in$ Dom(ρ) **then** bad $\leftarrow$ `true`
18: **else** $\rho(X'_i) \xleftarrow{\$} \{0,1\}^\ell$
19: $Y'_i \leftarrow \rho(X'_i)$

Game 1. The game of *fCBC* collision.

Thus, if $k < m$ and $k < m'$, or if either $k = m$ or $k = m'$ (but not both, because $M \neq M'$), then Y_m and $Y'_{m'}$ are uniformly distributed and independent. Thus, we have:

$$\Pr[Y_m = Y'_{m'} \mid \text{bad} = \texttt{false}] = \frac{1}{2^\ell}, \tag{20}$$

where the probability is taken over random function ρ. Moreover, the following equations hold:

$$\begin{aligned}\text{Col}_{n,\ell}(m,m') &= \Pr[Y_m = Y'_{m'}] \\ &= \Pr[Y_m = Y'_{m'} \wedge \text{bad} = \texttt{false}] \\ &\quad + \Pr[Y_m = Y'_{m'} \wedge \text{bad} = \texttt{true}],\end{aligned} \tag{21}$$

$$\begin{aligned}&\Pr[Y_m = Y'_{m'} \wedge \text{bad} = \texttt{false}] \\ &\leq \frac{\Pr[Y_m = Y'_{m'} \wedge \text{bad} = \texttt{false}]}{\Pr[\text{bad} = \texttt{false}]} \\ &= \Pr[Y_m = Y'_{m'} \mid \text{bad} = \texttt{false}] = \frac{1}{2^\ell},\end{aligned} \tag{22}$$

$$\Pr[Y_m = Y'_{m'} \wedge \text{bad} = \texttt{true}] \leq \Pr[\text{bad} = \texttt{true}]. \tag{23}$$

Thus, we have:

$$\text{Col}_{n,\ell}(m,m') \leq \Pr[\text{bad} = \texttt{true}] + \frac{1}{2^\ell}. \tag{24}$$

Now, to bound the probability of setting the bad flag, note that it is set to true in lines 5, 11, and 17 of Game 1. Just before these lines X_i or X'_i are assigned and if it is in the domain of ρ, the bad flag is set. For the case in which the bad flag has not been yet set to true, the value assigned to X_i or X'_i is uniformly-random and independent. That is because the value of Y_{i-1} or Y'_{i-1} is uniformly-random and independent, and it is XORed with the message to form X_i or X'_i. If there are $i-1$ values in the domain of ρ, the probability of a collision for the first time for a new random value is $(i-1)/2^\ell$. Thus, the following equation holds:

$$\Pr[\text{bad} = \texttt{true}] \leq \sum_{i=1}^{m+m'} \frac{i-1}{2^\ell} \leq \frac{(m+m')^2}{2^\ell}. \tag{25}$$

Finally, we have:

$$\text{Col}_{n,\ell}(m,m') \leq \frac{(m+m')^2+1}{2^\ell}.\square \tag{26}$$

Theorem 3 (security of *fFCBC*)

For any adversary $\mathcal{A}$ that asks q oracle queries, each with a maximum length of m blocks, we have:

$$\begin{aligned}&\Pr[\rho_1,\rho_2,\rho_3 \xleftarrow{\$} \text{Func}(n,\ell) : \mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} \Rightarrow 1] \\ &\quad - \Pr[\rho \xleftarrow{\$} \text{Func}(*,\ell) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1] \leq \frac{(4m^2+1)q^2}{2^{\ell+1}}.\end{aligned} \tag{27}$$

Proof of Theorem 3

Let us define Col as the event of the occurrence of a collision in the outputs of ρ_2 or in the outputs of ρ_3 (yet not between them). Thus, we can break the left side of the theorem equation to:

$$\begin{aligned}&\Pr[\rho_1,\rho_2,\rho_3 \xleftarrow{\$} \text{Func}(n,\ell) : \mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} \\ &\quad \Rightarrow 1 \mid \text{Col}]\Pr[\text{Col}] + \Pr[\rho_1,\rho_2,\rho_3 \xleftarrow{\$} \text{Func}(n,\ell) : \\ &\quad \mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} \Rightarrow 1 \mid \overline{\text{Col}}]\Pr[\overline{\text{Col}}] \\ &\quad - \Pr[\rho \xleftarrow{\$} \text{Func}(*,\ell) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1].\end{aligned} \tag{28}$$

Using the following simple equations:

$$\begin{aligned}&\Pr[\rho_1,\rho_2,\rho_3 \xleftarrow{\$} \text{Func}(n,\ell) : \mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} \\ &\quad \Rightarrow 1 \mid \text{Col}] \leq 1,\end{aligned} \tag{29}$$

$$\Pr[\overline{\text{Col}}] \leq 1. \tag{30}$$

We can bound the left-hand side of the theorem equation as follows:

$$\begin{aligned}&\Pr[\text{Col}] + \Pr[\rho_1,\rho_2,\rho_3 \xleftarrow{\$} \text{Func}(n,\ell) : \mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)} \\ &\quad \Rightarrow 1 \mid \overline{\text{Col}}] - \Pr[\rho \xleftarrow{\$} \text{Func}(*,\ell) : \mathcal{A}^{\rho(\cdot)} \Rightarrow 1].\end{aligned} \tag{31}$$

If there is no collision, it is apparent that:

$$\Pr[\rho_1,\rho_2,\rho_3\xleftarrow{\$}\mathrm{Func}(n,\ell):$$

$$\mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)}\Rightarrow 1\mid \overline{\mathrm{Col}}]$$

$$=\Pr[\rho\xleftarrow{\$}\mathrm{Func}(*,\ell):\mathcal{A}^{\rho(\cdot)}\Rightarrow 1], \quad (32)$$

and, therefore, we have:

$$\Pr[\rho_1,\rho_2,\rho_3\xleftarrow{\$}\mathrm{Func}(n,\ell):\mathcal{A}^{fFCBC[\rho_1,\rho_2,\rho_3](\cdot)}\Rightarrow 1]$$

$$-\Pr[\rho\xleftarrow{\$}\mathrm{Func}(*,\ell):\mathcal{A}^{\rho(\cdot)}\Rightarrow 1]\leq \Pr[\overline{\mathrm{Col}}]. \quad (33)$$

Moreover, using the union bound:

$$\Pr[Col]\leq \Pr[\mathrm{PadCol}]+\Pr[\mathrm{UnpadCol}], \quad (34)$$

where UnpadCol is the event of a collision in the outputs of ρ_3 for unpadded messages, and PadCol is the event of a collision in the outputs of ρ_2 for padded messages.

Now, we can break the probability of PadCol as follows:

$$\Pr[\mathrm{PadCol}]\leq \sum_{i=1}^{q_{\mathrm{pad}}}\Pr[\mathrm{PadCol}_i], \quad (35)$$

where $\Pr[\mathrm{PadCol}_i]$ is the probability that the collision occurs for the first time after the i-th padded query, and q_{pad} is the total number of padded queries. Note that PadCol_i's are disjoint. Because there is no collision in the first $i-1$ queries, the responses of these queries, sent to the adversary, are uniformly random and independent. Thus, any adaptive strategy of $\mathcal{A}$ can be changed to a nonadaptive strategy without loss of advantage.

Using the result of Lemma 4, we have:

$$\Pr[\mathrm{PadCol}_i]\leq (i-1)\mathrm{Col}_{n,\ell}(m,m), \quad (36)$$

and the final bound on the probability of PadCol is:

$$\Pr[\mathrm{PadCol}]\leq \sum_{i=1}^{q_{\mathrm{pad}}}(i-1)\mathrm{Col}_{n,\ell}(m,m)$$

$$\leq \frac{(4m^2+1)q_{\mathrm{pad}}^2}{2^{\ell+1}}. \quad (37)$$

The same bound is also applied for UnpadCol. Finally, the theorem result is obtained:

$$\Pr[\mathrm{Col}]\leq \frac{(4m^2+1)q_{\mathrm{pad}}^2}{2^{\ell+1}}+\frac{(4m^2+1)q_{\mathrm{unpad}}^2}{2^{\ell+1}}$$

$$\leq \frac{(4m^2+1)q^2}{2^{\ell+1}}.\square \quad (38)$$

3.2. Using a Carter-Wegman MAC

LAE1 calls SPRING twice per message block. Moreover, the performance results presented in Section 5 show that SPRING computation is the most time-consuming part of the execution. The idea of designing a more efficient scheme is to reduce the number of SPRING invocations in the authentication part. We have used the technique of Wegman and Carter [45] to build a MAC using a universal class of hash functions and the lattice-based PRF SPRING. In order to apply this technique, the ciphertext is given to a secret function from an XOR-universal class of hash functions [46], and the output is masked with the PRF output on a fresh and unique input.

Scheme 2 and Figure 2 show the second proposed lattice-based AE, referred to as LAE2, using the Carter-Wegman method [45]. Multiplications in this scheme are performed in $\mathrm{GF}(2^n)$. The encryption part is the same as before in LAE1. The universal class of hash functions utilized in LAE2 is defined as follows:

$$H_a(C)=C[1]a^{m+1}+C[2]a^m+\ldots+C[m]a^2+\mathsf{len}(C)a.$$

Here, $m=|C|_\ell$ and the polynomial $a\in\mathrm{GF}(2^n)$ is the function index. All the operations are performed in $\mathrm{GF}(2^n)$.

Using an XOR-universal hash function built by iterative multiplications of a secret polynomial $a\in\mathrm{GF}(2^n)$ is similar to the NIST-recommended GCM mode of operation [15]. However, GCM encrypts a constant block (zero block) to derive the second key for authentication, while LAE2 uses another key K_2 to save one SPRING invocation (see the more detailed discussion in Section 3.1 on a similar case). Moreover, there are many pad and msb functions involved because of asymmetry between the input and output length of SPRING, which should be handled carefully in the security proof.

procedure AUTHENCRYPT(N, M)
  $(M[1],\ldots,M[m])\xleftarrow{\ell} M$
  // LAE2-Enc
  for $i\leftarrow 1$ **to** $m-1$ **do**
    $Z[i]\leftarrow \mathrm{SPR}_{K_1}(N\|\mathsf{gray}(i))$
    $C[i]\leftarrow M[i]\oplus Z[i]$
  $Z[m]\leftarrow \mathsf{msb}_{|M[m]|}\left[\mathrm{SPR}_{K_1}(N\|\mathsf{gray}(m))\right]$
  $C[m]\leftarrow M[m]\oplus Z[m]$
  $C\leftarrow C[1]\|\ldots\|C[m]$
  // LAE2-MAC
  $Y[0]\leftarrow 0^n$
  for $i\leftarrow 1$ **to** $m-1$ **do**
    $Y[i]\leftarrow\left[Y[i-1]\oplus\left(C[i]\|0^{n-\ell}\right)\right]\otimes K_2$
  $Y[m]\leftarrow\left[Y[m-1]\oplus\left(C[m]\|0^{n-|C[m]|}\right)\right]\otimes K_2$
  $Y[m+1]\leftarrow\left(Y[m]\oplus\mathsf{len}(C)\right)\otimes K_2$
  $Y[m+2]\leftarrow\mathsf{msb}_\ell\left(Y[m+1]\right)\oplus\mathrm{SPR}_{K_1}(N\|\mathsf{gray}(0))$
  $T\leftarrow\mathsf{msb}_\tau\left(Y[m+2]\right)$
  return (C,T)

Scheme 2. LAE2.

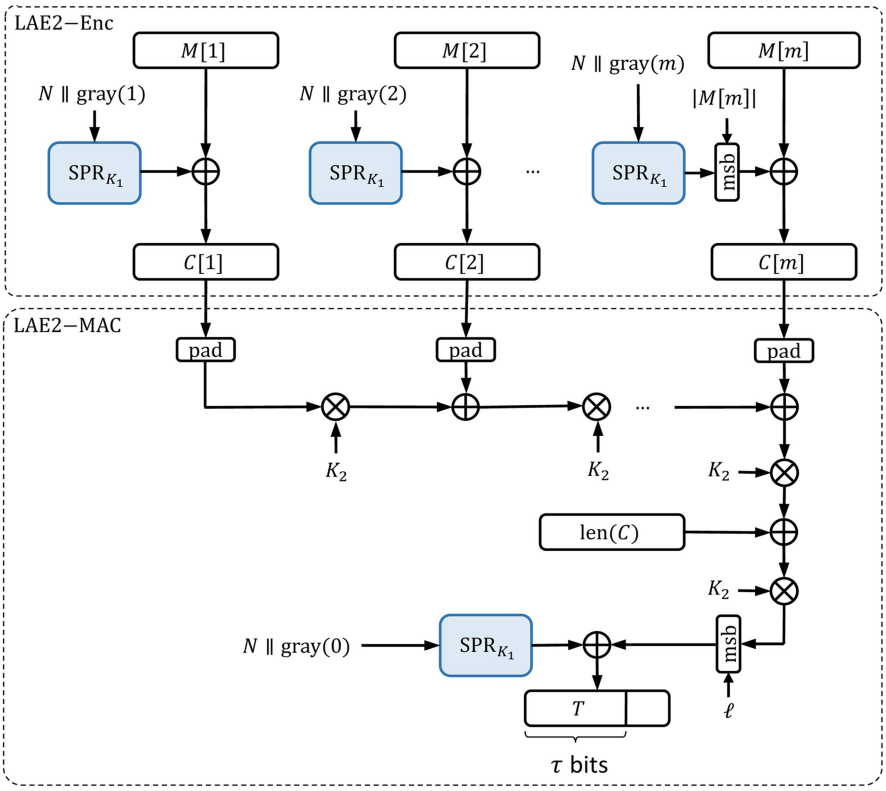

Figure 2. The flow diagram of LAE2 authenticated encryption procedure. Top is the encryption part (LAE2-Enc), and bottom is the authentication part (LAE2-MAC).

Although LAE2 is considered as a two-pass authenticated encryption scheme, the multiplications in GF(2^n) can be computed very efficiently. Thus, the authentication part is much more efficient than LAE1. Typical high-end processors have instruction set extensions to perform this operation in a few clock cycles (see Section 5 for more details on the ones used in our benchmarks). Moreover, when an old or constrained processor is required, or in the case of hardware implementations, this multiplication can also be performed efficiently using a moderate-sized precomputed lookup table. Nevertheless, both LAE1 and LAE2 are not fully parallel, and their authentication part should be computed sequentially.

It is worth mentioning that replacing the majority of SPRING functions with polynomial multiplications in GF(2^n) does not introduce any new security assumption in LAE2. The hardness of lattice problems is still the only supporting assumption. Thus, many SPRING invocations in the authentication part of LAE1 are reduced to only one in LAE2, without any loss in the security level.

3.2.1. Security of LAE2

Theorems 4 and 5 show the security of LAE2.

Theorem 4 (privacy of LAE2). Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \rightarrow \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{A}$ to attack the privacy of $LAE2[\mathcal{F}]$, who runs in time t and asks q oracle queries, each of which has at most $m < 2^{n-w}$ blocks; there exists an adversary $\mathcal{P}$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{ind-cpa}}_{LAE2[\mathcal{F}]}(\mathcal{A}) \leq \mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}). \tag{39}$$

Moreover, adversary P asks $q' = \sigma + q$ oracle queries and runs in time $t' = t + (\sigma + q)t_{\otimes} + \alpha n(\sigma + q)$, where $t_{\otimes}$ is the time to compute a GF(2^n) multiplication, and α is a constant depending on the model of the computation.

Theorem 5 (authenticity of LAE2). Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \rightarrow \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{A}$ to attack the authenticity of $LAE2[\mathcal{F}]$, who runs in time t and asks q_e encryption and q_d decryption queries, each of which with a total length of σ_e and σ_d blocks, there exists an adversary $\mathcal{P}$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{int-ctxt}}_{LAE2[\mathcal{F}]}(\mathcal{A}) \leq \mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{q}{2^\ell - q2^{\ell-\tau}} + \frac{q_d}{2^\tau}, \tag{40}$$

where $q = q_e + q_d$. Moreover, adversary $\mathcal{P}$ asks $q' = \sigma_e q_e + q_d$ oracle queries and runs in time $t' = t + (\sigma +$

$q)t_{\otimes} + \alpha n(\sigma + q)$, where $\sigma = \sigma_e + \sigma_d$, $t_{\otimes}$ is the time to compute a GF(2^n) multiplication, and α is a constant depending on the model of the computation.

The proofs of Theorems 4 and 5 are presented in Section 3.2.2.

3.2.2. Security proofs

The proof sketch of Theorem 4 is as follows. The problem in the information-theoretic setting, i.e., when $F \xleftarrow{\$} \text{Func}(n, \ell)$, is simple. Both ciphertext C and tag T are XORed with the output of F. In addition, a nonce-respecting adversary $\mathcal{A}$ always queries with a fresh nonce N. Thus, the distribution of (C, T) is perfectly uniform. Transition from information-theoretic to complexity-theoretic setting is also standard.□

Theorem 5 can be proven in a standard way from Lemma 5.

Lemma 5 (authenticity of ideal LAE2). Fix $n \geq \ell \geq 1$. Let $\mathcal{R} = \text{Func}(n, \ell)$. For any adversary $\mathcal{A}$ asking q_e encryption and q_d decryption queries, we have:

$$\text{Adv}^{\text{int-ctxt}}_{LAE2[\mathcal{R}]}(\mathcal{A}) \leq \frac{q}{2^{\ell} - q2^{\ell-\tau}} + \frac{q_d}{2^{\tau}}, \tag{41}$$

where $q = q_e + q_d$.

Proof of Lemma 5

The technique of sequences of games [47] is used to prove this result. Game 2 (G_0) simulates the environment of Lemma 5 for adversary $\mathcal{A}$. There is an exception whose AuthDecrypt procedure in G_0 returns `true` instead of the message in the case that the forgery is successful. However, this difference can be ignored because the following equation always holds.

$$\text{Adv}^{\text{int-ctxt}}_{LAE2[\mathcal{R}]}(\mathcal{A}) = \Pr[G_0^{\mathcal{A}} \Rightarrow \text{wins}]. \tag{42}$$

Note that, without loss of generality, we can assume that adversary $\mathcal{A}$ is deterministic. Therefore, the probability is taken only over the random function sampled from $\mathcal{R}$.

G_0 models random function ρ using the lazy sampling technique. Games G_1 and G_2 are specified in Game 3. The underlined statements exist only in game G_1 and are removed in G_2. From $\mathcal{A}$'s point of view, G_0 and G_1 are identical. Thus, the winning probability of $\mathcal{A}$ remains unchanged from G_0 to G_1. On the other hand, G_1 and G_2 are identical until bad. That is, as long as no bad flags (bad_1 or bad_2) are set to true, these two games are identical for $\mathcal{A}$. Thus, according to the fundamental lemma of game playing [48], we have:

$$\Pr[G_1^{\mathcal{A}} \Rightarrow \text{wins}] - \Pr[G_2^{\mathcal{A}} \Rightarrow \text{wins}] \leq \Pr[G_2^{\mathcal{A}} \text{sets bad}_1 \text{or bad}_2]. \tag{43}$$

However, as the AuthDecrypt procedure of G_2 always returns $\perp$, we have $\Pr[G_2^{\mathcal{A}} \Rightarrow \text{wins}] = 0$. Thus, we can conclude:

$$\begin{aligned}\text{Adv}^{\text{int-ctxt}}_{LAE2[\mathcal{R}]}(\mathcal{A}) &\leq \Pr[G_2^{\mathcal{A}} \text{ sets bad}_1 \text{ or bad}_2] \\ &\leq \Pr[G_2^{\mathcal{A}} \text{ sets bad}_1] \\ &\quad + \Pr[G_2^{\mathcal{A}} \text{ sets bad}_2].\end{aligned} \tag{44}$$

Note that bad_1 never occurs because $\mathcal{A}$ is nonce-respecting and never provides repeated nonces to the AuthEncrypt oracle. This is true even if the nonce value N, given to AuthEncrypt, has been repeatedly used in previous AuthDecrypt queries. We define $\Pr[G_2^{\mathcal{A}} \text{ sets bad}_2 \text{ on } q_i]$ as the probability that bad_2 is set to true on the i-th query for the first time. As these probabilities are disjoint, we have:

$$\Pr[G_2^{\mathcal{A}} \text{ sets bad}_2] = \sum_{i=1}^{q} \Pr[G_2^{\mathcal{A}} \text{ sets bad}_2 \text{ on } q_i]. \tag{45}$$

Now, we bound $\Pr[G_2^{\mathcal{A}} \text{ sets bad}_2 \text{ on } q_i]$.

procedure AuthEncrypt(N, M)
- $(M[1], ..., M[m]) \xleftarrow{\ell} M$
- **for** $i \leftarrow 1$ **to** $m - 1$ **do**
  - **if** $N \| \text{gray}(i) \notin Dom(F)$ **then**
    - $F(N \| \text{gray}(i)) \xleftarrow{\$} \{0,1\}^{\ell}$
  - $Z[i] \leftarrow F(N \| \text{gray}(i))$
  - $C[i] \leftarrow M[i] \oplus Z[i]$
- **if** $N \| \text{gray}(m) \notin Dom(F)$ **then**
  - $F(N \| \text{gray}(m)) \xleftarrow{\$} \{0,1\}^{\ell}$
- $Z[m] \leftarrow \text{msb}_{|M[m]|}\left[F(N \| \text{gray}(m))\right]$
- $C[m] \leftarrow M[m] \oplus Z[m]$
- $C \leftarrow C[1] \| ... \| C[m]$
- $Y[0] \leftarrow 0^n$
- **for** $i \leftarrow 1$ **to** $m - 1$ **do**
  - $Y[i] \leftarrow \left[Y[i-1] \oplus \left(C[i] \| 0^{n-\ell}\right)\right] \otimes K_2$
- $Y[m] \leftarrow \left[Y[m-1] \oplus \left(C[m] \| 0^{n-|C[m]|}\right)\right] \otimes K_2$
- $Y[m+1] \leftarrow \left(Y[m] \oplus \text{len}(C)\right) \otimes K_2$
- **if** $N \| \text{gray}(0) \notin Dom(F)$ **then**
  - $F(N \| \text{gray}(0)) \xleftarrow{\$} \{0,1\}^{\ell}$
- $Y[m+2] \leftarrow \text{msb}_{\ell}(Y[m+1]) \oplus F\left(N \| \text{gray}(0)\right)$
- $T \leftarrow \text{msb}_{\tau}(Y[m+2])$
- **return** (C, T)

procedure AuthDecrypt(N, C, T)
- $(C_1, ..., C_m) \xleftarrow{\ell} C$
- $Y[0] \leftarrow 0^n$
- **for** $i \leftarrow 1$ **to** $m - 1$ **do**
  - $Y[i] \leftarrow \left[Y[i-1] \oplus \left(C[i] \| 0^{n-\ell}\right)\right] \otimes K_2$
- $Y[m] \leftarrow \left[Y[m-1] \oplus \left(C[m] \| 0^{n-|C[m]|}\right)\right] \otimes K_2$
- $Y[m+1] \leftarrow \left(Y[m] \oplus \text{len}(C)\right) \otimes K_2$
- **if** $N \| \text{gray}(0) \notin Dom(F)$ **then**
  - $F(N \| \text{gray}(0)) \xleftarrow{\$} \{0,1\}^{\ell}$
- $Y[m+2] \leftarrow \text{msb}_{\ell}(Y[m+1]) \oplus F\left(N \| \text{gray}(0)\right)$
- $T^* \leftarrow \text{msb}_{\tau}(Y[m+2])$
- **if** $T = T^*$ **then**
  - wins $\leftarrow$ `true`; **return** `true`
- **return** $\perp$

Game 2. G_0 for LAE2.

procedure AUTHENCRYPT(N, M)
  $(M[1], ..., M[m]) \xleftarrow{\ell} M$
  for $i \leftarrow 1$ **to** $m-1$ **do**
    $Z[i] \xleftarrow{\$} \{0,1\}^{\ell}$
    if $N \| \mathsf{gray}(i) \in Dom(F)$ **then**
      $\mathrm{bad}_1 \leftarrow \mathtt{true}$; $\underline{Z[i] \leftarrow F\big(N \| \mathsf{gray}(i)\big)}$
    $F\big(N \| \mathsf{gray}(i)\big) \leftarrow Z[i]$
    $C[i] \leftarrow M[i] \oplus Z[i]$
  $Z[m] \xleftarrow{\$} \{0,1\}^{\ell}$
  if $N \| \mathsf{gray}(m) \in Dom(F)$ **then**
    $\mathrm{bad}_1 \leftarrow \mathtt{true}$; $\underline{Z[m] \leftarrow F\big(N \| \mathsf{gray}(m)\big)}$
  $F\big(N \| \mathsf{gray}(m)\big) \leftarrow Z[m]$
  $Z[m] \leftarrow \mathsf{msb}_{|M[m]|}\big[Z[m]\big]$
  $C[m] \leftarrow M[m] \oplus Z[m]$
  $C \leftarrow C[1] \| ... \| C[m]$
  $Y[0] \leftarrow 0^n$
  for $i \leftarrow 1$ **to** $m-1$ **do**
    $Y[i] \leftarrow \Big[Y[i-1] \oplus \big(C[i] \,\|\, 0^{n-\ell}\big)\Big] \otimes K_2$
  $Y[m] \leftarrow \Big[Y[m-1] \oplus \big(C[m] \,\|\, 0^{n-|C[m]|}\big)\Big] \otimes K_2$
  $Y[m+1] \leftarrow \big(Y[m] \oplus \mathsf{len}(C)\big) \otimes K_2$
  if $N \| \mathsf{gray}(0) \notin Dom(F)$ **then**
    $F\big(N \| \mathsf{gray}(0)\big) \xleftarrow{\$} \{0,1\}^{\ell}$
  $Y[m+2] \leftarrow \mathsf{msb}_{\ell}\big(Y[m+1]\big) \oplus F\left(N \| \mathsf{gray}(0)\right)$
  $T \leftarrow \mathsf{msb}_{\tau}\big(Y[m+2]\big)$
  return (C, T)

procedure AUTHDECRYPT(N, C, T)
  $(C_1, ..., C_m) \xleftarrow{\ell} C$
  $Y[0] \leftarrow 0^n$
  for $i \leftarrow 1$ **to** $m-1$ **do**
    $Y[i] \leftarrow \Big[Y[i-1] \oplus \big(C[i] \,\|\, 0^{n-\ell}\big)\Big] \otimes K_2$
  $Y[m] \leftarrow \Big[Y[m-1] \oplus \big(C[m] \,\|\, 0^{n-|C[m]|}\big)\Big] \otimes K_2$
  $Y[m+1] \leftarrow \big(Y[m] \oplus \mathsf{len}(C)\big) \otimes K_2$
  if $N \| \mathsf{gray}(0) \notin Dom(F)$ **then**
    $F\big(N \| \mathsf{gray}(0)\big) \xleftarrow{\$} \{0,1\}^{\ell}$
  $Y[m+2] \leftarrow \mathsf{msb}_{\ell}\big(Y[m+1]\big) \oplus F\left(N \| \mathsf{gray}(0)\right)$
  $T^* \leftarrow \mathsf{msb}_{\tau}\big(Y[m+2]\big)$
  if $T = T^*$ **then**
    $\mathrm{bad}_2 \leftarrow \mathtt{true}$; $\underline{\mathrm{wins} \leftarrow \mathtt{true}}$; **return** $\mathtt{true}$
  return $\perp$

Game 3. $\underline{G_1}, G_2$ for LAE2.

Let N_i, C_i, T_i, and T_i^* be the values appeared in the AUTHENCRYPT or AUTHDECRYPT procedure of the i-th query in G_2 (not all of them may be defined for each i). The probability of setting bad_2 is zero for the encryption queries because $\mathcal{A}$ is nonce-respecting. For the decryption queries, we have:

$$\Pr[G_2^{\mathcal{A}} \text{ sets } \mathrm{bad}_2 \text{ on } q_i] = \Pr[T_i = T_i^*]. \tag{46}$$

We claim that the event $T_i = T_i^*$ is independent of all the previous queries q_j $(j < i)$ with $N_j \neq N_i$. If q_j is an encryption query, the returned C_j is uniformly random and independent, and T_j is deterministically calculated from C_j. If q_j is a decryption query, the returned $\perp$ only reveals that $T_j \neq T_j^*$, where T_j^* is independent of the event $T_i = T_i^*$.

Now, we assume that adversary $\mathcal{A}$ uses the same nonce N for all the queries prior and include q_i, to maximize the probability of setting bad_2 on q_i. There are two cases in this setting:

- **Case 1:** All the previous queries are AUTHDECRYPT, and are responded by a $\perp$. In this case, for $j < i$, it is revealed to $\mathcal{A}$ that $T_j \neq T_j^* = \mathrm{msb}_{\tau}\big(H_{K_2}(C_j)\big) \oplus N^*$, where $N^* = \mathrm{msb}_{\tau}\left(\rho\left(N \| \mathrm{gray}(0)\right)\right)$ and is common between all the queries. Thus, at most $2^{\ell-\tau}$ choices of K_2 become invalid after each query. On the other hand, only one of the choices makes $T_i = T_i^*$. Therefore, the following equation holds:

$$\Pr[T_i = T_i^*] = \frac{1}{2^{\ell} - (i-1)2^{\ell-\tau}}. \tag{47}$$

- **Case 2:** Exactly one of the previous queries is AUTHENCRYPT. Assume that q_k $(1 \leq k < i)$ is the encryption query. The other queries are all AUTHDECRYPT and are responded by a $\perp$. Thus, $\mathcal{A}$ has a tuple (M_k, C_k, T_k) for which $T_k = \mathrm{msb}_{\tau}\big(H_{K_2}(C_k)\big) \oplus N^*$ holds. Due to the fact that N^* is an independent and uniformly-random value (random function F evaluated on fresh nonce N), T_k has also this property. Hence, the pair (C_k, T_k) causing at most $2^{\ell-\tau}$ choices of K_2 becomes invalid; similar to Case 1, Eq. (47) holds.

Moreover, another strategy is to perform exhaustive search on the correct tag with an advantage of $q_d/2^{\tau}$. Finally, we can conclude as follows:

$$\mathrm{Adv}^{\text{int-ctxt}}_{LAE2[\mathcal{R}]}(\mathcal{A}) \leq \sum_{i=1}^{q} \frac{1}{2^{\ell} - (i-1)2^{\ell-\tau}} + \frac{q_d}{2^{\tau}}$$

$$\leq \frac{q}{2^{\ell} - q2^{\ell-\tau}} + \frac{q_d}{2^{\tau}}.\square \tag{48}$$

4. Single-pass lattice-based authenticated encryption

LAE1 and LAE2 perform the encryption and authentication processes separately, running the underlying primitive twice per message block. A single-pass AE performs these tasks in a single processing. To be more specific, it calls the primitive function or permutation once per input block, in addition to a few constant number of calls. Firstly, Jutla [21] introduced the first secure single-pass authenticated encryption IAPM. Rogaway et al. [22] extended this idea to build OCB, which is a fast and well-featured AE [23,24].

Scheme 3 is the third proposed AE scheme, referred to as LAE3. It is derived from a recent AE from Minematsu [25] called OTR. OTR is an extension to OCB, which uses a Feistel structure to replace each pair of block ciphers in OCB with a pair of PRFs. Figure 3 shows the structure and data flow of LAE3. In this scheme, similar to the last block of LAE1, a

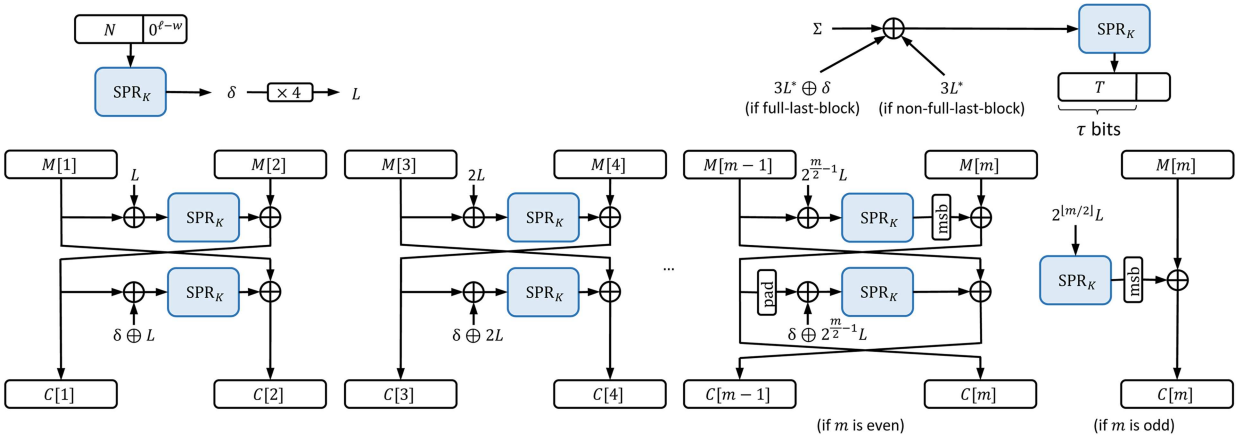

Figure 3. The flow diagram of LAE3 authenticated encryption procedure. For the description of L^* and $\sum$, refer to Scheme 3. Note that, in LAE3, the input of SPRING is reduced to ℓ bits.

procedure AUTHENCRYPT(N, M)
  $\Sigma \leftarrow 0^{\ell}$
  $\delta \leftarrow \text{SPR}_K(N \,\|\, 0^{\ell-w})$
  $L \leftarrow 4\delta$
  $(M[1], ..., M[m]) \xleftarrow{\ell} M$
  for $i \leftarrow 1$ **to** $\lceil m/2 \rceil$ **do**
    $C[2i-1] \leftarrow \text{SPR}_K\Big(L \oplus M[2i-1]\Big) \oplus M[2i]$
    $C[2i] \leftarrow \text{SPR}_K\Big(L \oplus \delta \oplus C[2i-1]\Big) \oplus M[2i-1]$
    $\Sigma \leftarrow \Sigma \oplus M[2i]$
    $L \leftarrow 2L$
  if m is even **then**
    $L^* \leftarrow L \oplus \delta$
    $Z \leftarrow \text{SPR}_K\Big(L \oplus M[m-1]\Big)$
    $C[m] \leftarrow \text{msb}_{|M[m]|}(Z) \oplus M[m]$
    $C[m-1] \leftarrow \text{SPR}_K\Big(L^* \oplus \text{pad}_{\ell}\big(C[m]\big)\Big) \oplus M[m-1]$
    $\Sigma \leftarrow \Sigma \oplus Z \oplus \text{pad}_{\ell}\big(C[m]\big)$
  if m is odd **then**
    $L^* \leftarrow L$
    $C[m] \leftarrow \text{msb}_{|M[m]|}\big(\text{SPR}_K(L^*)\big) \oplus M[m]$
    $\Sigma \leftarrow \Sigma \oplus M[m]$
  if $|M[m]| \neq n$ **then**
    $T \leftarrow \text{msb}_{\tau}\Big(\text{SPR}_K\big(3L^* \oplus \Sigma\big)\Big)$
  else
    $T \leftarrow \text{msb}_{\tau}\Big(\text{SPR}_K\big(3L^* \oplus \delta \oplus \Sigma\big)\Big)$
  $C \leftarrow C[1] \,\|\, ... \,\|\, C[m]$
  return (C, T)

Scheme 3. LAE3.

whitening value is XORed with the input of SPRING to obtain a tweakable PRF. The polynomial operations in this scheme are performed in GF(2^{ℓ}). Note that the multiplication of small constants, for instance in 4δ and $2L$, are easily computed by some shifts and XORs. For the sake of simplicity, the input of SPRING is reduced to ℓ bits in LAE3. Thus, both input and output of SPRING are ℓ bits in this scheme. The remaining least significant $(n - \ell)$ bits are padded with zero. Note that, in Scheme 3 and Figure 3, this padding is considered inside the SPRING function and is not applied explicitly.

There are two differences between LAE3 and OTR. Firstly, the block cipher is replaced with a PRF. Although [25] introduced a PRF-capable version of OTR; however, there were technical issues about the integration of SPRING in this scheme. The proposed scheme does not use the OTR technique to obtain a tweakable PRF from SPRING. Instead, some whitening value is XORed with the input of SPRING. The other difference is that the input and output lengths of SPRING are not equal. We use only ℓ bits of SPRING input (padding it with zeroes) to make the input and output lengths the same.

There are few proposals for single-pass non-lattice-based authenticated encryption in the literature. The methods of IAPM [21] and OCB [22-24] cannot be followed in the case of SPRING. In these schemes, the plaintext is directly fed into the underlying primitive function, and the output becomes a part of the ciphertext. PRFs are not useful in this setting because a PRF is not necessarily a bijection; hence, it has data loss. Only a pseudorandom permutation (e.g., a block cipher) can be used in such designs. Minematsu [25] introduced OTR as a PRF-capable single-pass AE. OTR utilizes the tweak-prepending technique to construct a tweakable PRF from a normal one. Unfortunately, this technique cannot be applied to SPRING. That is because SPRING has a fixed and short input length n, while tweak-prepending requires a variable-input-length PRF or one with a large-enough input length. Alternatively, the input whitening technique is used in LAE3. In this case, the outputs of some functions of an XOR-universal class are XORed with the input of SPRINGs to make a tweakable PRF. As a final note, LAE3 requires a fresh δ for each nonce N. Thus, we cannot save a SPRING invocation, similar to LAE1, by introducing the second key.

4.1. Security of LAE3

Theorems 6 and 7 show the security of LAE3.

Theorem 6 (privacy of LAE3)

Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \rightarrow \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{A}$ to attack the privacy of $LAE3[\mathcal{F}]$, who runs in time t and asks q encryption queries, with a maximum total length of σ blocks, there exists an adversary $\mathcal{P}$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{ind-cpa}}_{LAE3[\mathcal{F}]}(\mathcal{A}) \leq \mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{6(q+\sigma)^2}{2^\ell}. \tag{49}$$

Moreover, adversary $\mathcal{P}$ asks $q' = \sigma + 2q$ oracle queries, and runs in time $t' = t + \alpha\ell(\sigma + q)$, where α is a constant depending on the model of the computation.

Theorem 7 (authenticity of LAE3)

Fix $n \geq \ell \geq 1$. Let $\mathcal{F} = \{F : \{0,1\}^n \rightarrow \{0,1\}^\ell\}$ be a family of functions. For any adversary $\mathcal{A}$ to attack the authenticity of $LAE3[\mathcal{F}]$, who runs in time t and asks q_e encryption and q_d decryption queries, with a maximum total length of σ_e and σ_d blocks, respectively, there exists an adversary $\mathcal{P}$ against the pseudorandomness of $\mathcal{F}$, and we have:

$$\mathrm{Adv}^{\text{int-ctxt}}_{LAE3[\mathcal{F}]}(\mathcal{A}) \leq \mathrm{Adv}^{\text{prf}}_{\mathcal{F}}(\mathcal{P}) + \frac{6(q_e + q_d + \sigma_e + \sigma_d)^2}{2^\ell} + \frac{q_d}{2^\tau}. \tag{50}$$

Moreover, adversary $\mathcal{P}$ asks $q' = q_e + q_d + \sigma_e + \sigma_d$ number of oracle queries, and runs in time $t' = t + \alpha\ell(\sigma_e + \sigma_d + 2q_e + 2q_d)$, where α is a constant depending on the model of the computation.

Proofs of Theorems 6 and 7

We skip the proofs of Theorems 6 and 7 as they are similar to the privacy and authenticity proofs of OTR [25, Theorems 1 and 2]. The main difference is that the associated data are not involved in the proof because LAE3 does not support it.

5. Comparison and performance results

In this section, the efficiency and performance results of the proposed schemes are reported. These schemes are implemented with moderate optimizations. The target of the implementations are high-end processors because single-instruction multiple-data (SIMD) is utilized to speed up polynomial operations of the lattice-based AEs. The source codes of the implementations are integrated into the OpenSSL framework to be benchmarked along with the optimized implementation of AES-128-GCM and AES-256-GCM in OpenSSL. GCM [15] is an efficient and widely-used authenticated encryption mode of operation. AES-128-GCM and AES-256-GCM are two instantiations of GCM using 128-bit and 256-bit AES, respectively. The implemented lattice-based schemes are parameterized with $n = 128$, $\ell = 127$, $p = 514$, and $w = 96$ to claim 128-bit security in the pre- and post-quantum settings. Consequently, AES-256-GCM is also included in the comparison, which is 128-bit secure in the post-quantum setting. The three proposed schemes LAE1, LAE2, and LAE3 are implemented based on the SPRING implementation provided by its designers [27]. The implementation of the SPRING function in LAE3 and in the authentication part of LAE1 cannot make use of the optimizations introduced in [27, Section 3] regarding the Gray-code input.

The proposed schemes are implemented in C++ and integrated with the benchmark of OpenSSL version 1.0.1i. The OpenSSL benchmark is modified to feed more variable sizes of input, and use the "rdtsc" instruction to count the processor clocks. The instruction set of the Intel Carry-Less Multiplication (CLMUL) is used to multiply polynomials in GF(2^{128}). The benchmarking process is executed on one CPU core. The source code is compiled using GCC version 4.8.2. Two compiler options "O3" and "march=native" are specified in order to configure the compiler to generate optimized codes according to the targeted processors. Thus, the source code is recompiled on each targeted machine. The Intel SSE2 instruction set is utilized indirectly by using vector processing features of GCC. OpenSSL, and the integrated new schemes were built into a 64-bit binary executable. The benchmark is run on a 64-bit Linux machine with kernel version 3.13.30. Two Intel CPUs are used in the experiments. The first one is Intel Core i3-2120 running at 3.3GHz with Sandy Bridge microarchitecture, and the second one is Intel Core i7-4770 running at 3.4GHz with Haswell microarchitecture.

Table 1 presents the processor clock cycles on Intel Sandy Bridge and Haswell microarchitectures, running the encryption procedures of LAE1, LAE2, and LAE3. Figure 4 also shows these results in a form of graph. The lengths chosen for the plaintexts are powers of 2 and 10 bytes in order to cover both complete and incomplete last blocks, as well as common IP packet sizes 40, 576, 1300, and 1500 bytes. Note that OpenSSL is configured not to use AES-NI instruction set of Intel CPUs for AES-128-GCM and AES-256-GCM.

Although LAE3 is of single-pass, it is not more efficient than the two-pass LAE2 due to the optimization techniques applied to the lattice-based PRF SPRING. SPRING is very efficient if it runs multiple times given the values of a Gray-code counter. In this environment, the aggregated computation of each SPRING is reduced significantly. In the case of LAE1, only the encryption part enjoys the optimization for the Gray-code input. As an improvement, the authentication part of LAE2 utilizes GF(2^{128}) multiplications instead

Table 1. Number of clock cycles per byte to run the encryption procedure of the proposed schemes LAE1, LAE2, and LAE3, compared to AES-128-GCM and AES-256-GCM. The top represents the results on one core of an Intel Core i3-2120 CPU with Sandy bridge microarchitecture, and the bottom represents the results on one core of an Intel Core i7-4770 CPU with Haswell microarchitecture. Green highlights are faster than AES-256-GCM, and pink highlights are twice slower.

	Sandy Bridge				
Message length (byte)	**LAE1**	**LAE2**	**LAE3**	**AES-128-GCM**	**AES-256-GCM**
10	319	123	294	40	52
16	283	89	241	39	52
32	185	60	149	37	49
40	148	54	119	38	48
64	135	45	103	35	47
100	113	40	84	34	47
128	110	38	80	34	47
256	98	34	70	35	46
512	92	32	64	33	46
576	90	31	63	33	46
1000	87	31	60	33	46
1024	87	31	60	33	46
1300	86	30	59	33	46
1500	86	30	59	33	46
	Haswell				
Message length (byte)	**LAE1**	**LAE2**	**LAE3**	**AES-128-GCM**	**AES-256-GCM**
10	212	81	196	26	34
16	192	60	159	26	34
32	126	43	98	24	32
40	101	39	79	23	31
64	93	33	68	23	30
100	79	30	55	22	30
128	76	29	52	21	30
256	68	27	45	21	29
512	64	25	41	21	29
576	64	25	41	21	29
1000	62	25	39	21	29
1024	62	25	39	21	29
1300	63	25	39	21	29
1500	62	24	39	21	29

of a chain of SPRINGs. There is only one SPRING invocation in the authentication part, which is also in the sequence of the encryption Gray-code counter. Therefore, SPRING computation is highly optimized in LAE2. The authentication pass is removed in LAE3, though the SPRING's inputs are no longer from a Gray-code counter, resulting in a less efficient scheme than LAE2. All the three proposed schemes are computationally heavy for very short plaintexts (e.g., less than 40 bytes). Only LAE2 becomes good for larger inputs. This phenomenon is common in symmetric encryption schemes; however, it is a more considerable issue in the case of the proposed lattice-based schemes because the saving in the aggregated computation of SPRINGs is substantial.

Table 2. Key size of the proposed schemes LAE1, LAE2, and LAE3 in comparison with AES-128-GCM and AES-256-GCM. Note that the issue of large keys for the proposed AEs can be managed in practice using a PRNG. More details are provided in Section 5.

Scheme	Key size (bits)
LAE1	98431 ($2 \times 49152 + 127$)
LAE2	49280 ($49152 + 128$)
LAE3	49152
AES-128-GCM	128
AES-256-GCM	256

If a small IP packet to be 40 bytes and a large IP packet to be 1500 bytes are considered, LAE1 is the least efficient scheme of all the other proposed schemes to be used for IP packet encryption. It is around twice slower than AES-256-GCM for both small and large packet sizes on Haswell and Sandy Bridge microarchitectures. LAE3 has a better performance; as for small IP packets, it is 146% and 151% slower than AES-256-GCM on Sandy Bridge and Haswell, respectively. However, for large IP packets, this scheme is 28% and 36% slower than AES-256-GCM, respectively. LAE2 has the best performance among the proposed AE schemes. It is only 12% and 24% slower than AES-256-GCM for small IP packets on Sandy Bridge and Haswell, respectively. Finally, it is 34% and 15% faster than AES-256-GCM for large IP packets, respectively, on the targeted microarchitectures.

Table 2 also shows the comparison of the key sizes between the proposed schemes and the two references AES-128-GCM and AES-256-GCM. Similar to most lattice-based cryptographic schemes, the key sizes of the proposed AEs are very large. These sizes are for the key before applying the Fast Fourier Transform (FFT). The issue of large keys can be managed in practice by utilizing a pseudorandom number generator (PRNG). The PRNG is seeded with a short key to generate the larger key of each lattice-based AE. Note that the PRNG is executed only once in the key setup phase. It is similar to the key expansion process in block cipher-based AEs, which is usually not a performance critical task.

6. Conclusion

In this paper, three Authenticated Encryption (AE) schemes LAE1, LAE2, and LAE3, which enjoy a

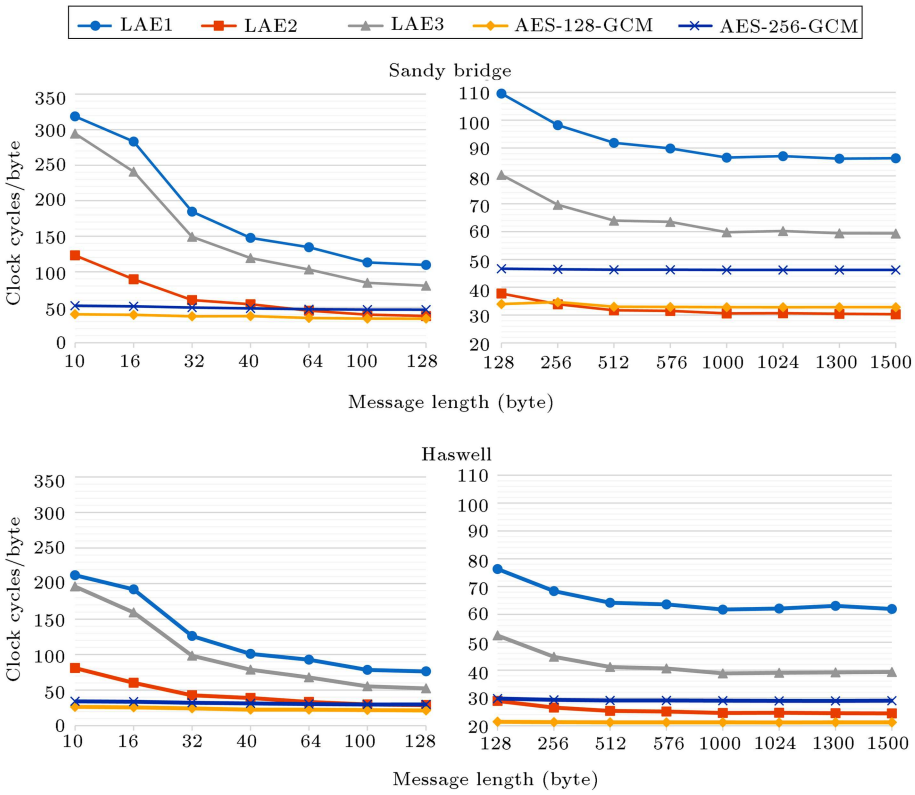

Figure 4. Performance results of the proposed schemes LAE1, LAE2, and LAE3 (encryption procedure), compared with AES-128-GCM and AES-256-GCM. The implementations are executed on one core of Intel Core i3-2120 with Sandy Bridge microarchitecture, and on one core of Intel Core i7-4770 with Haswell microarchitecture. The vertical axis is re-scaled by of the message length of 128 bytes.

security proof based on hard lattice problems, were presented. In addition, the exact security of these schemes in the paradigm of practice-oriented provable security was analyzed and proved. The proposed schemes have the following motivations and advantages over the previous AE constructions: (1) These schemes are alternatives to conventional AEs based on a block cipher, which may help if a quantum algorithm is discovered to break, e.g., AES; (2) If they are combined with a lattice-based asymmetric scheme for key distribution; then, the resulting hybrid encryption depends only on one (type of) security assumption. Otherwise, if a block-cipher-based AE (e.g., AES-GCM) is combined with an entirely different asymmetric key distribution (e.g., an RSA-based one), then a flaw or a new attack to the assumption of each part (AES-GCM, or RSA) breaks the whole security; (3) Computational resources can be saved in the case of a hybrid encryption which uses the proposed AEs as well as a lattice-based asymmetric scheme, due to the fact that some primitive computations are in common; (4) Our implementation results of the Intel Sandy Bridge and Haswell microarchitectures show that LAE2 is efficient enough to be used in practice and to compete with widely-used AEs.

Acknowledgment

The authors would like to thank Banerjee et al. [27] for providing the source code of their SPRING implementation. They also thank Mohammad Razeghi for his assistance in the implementation of the proposed schemes.

References

1. Bernstein, D.J. "Introduction to post-quantum cryptography", *Post-Quantum Cryptography*, Springer Berlin Heidelberg, pp. 1-14 (2009).
2. Peikert, C. "A decade of lattice cryptography", Technical Report 939, ePrint IACR (2015).
3. Shor, P.W. "Algorithms for quantum computation: Discrete logarithms and factoring", *Foundations of Computer Science, 35th Annual Symposium on*, IEEE, pp. 124-134 (1994).
4. Proos, J. and Zalka, C. "Shor's discrete logarithm quantum algorithm for elliptic curves", *Quantum Info. Comput.*, **3**(4), pp. 317-344 (2003).
5. Lindner, R. and Peikert, C. "Better Key Sizes (and Attacks) for LWE-Based Encryption", *Topics in Cryptology - CT-RSA 2011*, LNCS, **6558**, Springer, pp. 319-339 (2011).

6. Micciancio, D. and Peikert, C. "Trapdoors for lattices: Simpler, tighter, faster, smaller", *EUROCRYPT 2012*, LNCS, **7237**, Springer, pp. 700-718 (2012).
7. Ducas, L., Durmus, A., Lepoint, T., and Lyubashevsky, V. "Lattice signatures and bimodal gaussians", *CRYPTO 2013*, LNCS, **8042**, Springer, pp. 40-56 (2013).
8. Göttert, N., Feller, T., Schneider, M., Buchmann, J., and Huss, S. "On the design of hardware building blocks for modern lattice-based encryption schemes", *Cryptographic Hardware and Embedded Systems - CHES*, 2012, LNCS, **7428**, Springer, pp. 512-529 (2012).
9. Oder, T., Pöppelmann, T., and Güneysu, T. "Beyond ECDSA and RSA: lattice-based digital signatures on constrained devices", *51st Annual Design Automation Conference*, USA DAC'14, ACM, pp. 110:1-110:6 (2014).
10. Boorghany, A. and Jalili, R. "Implementation and comparison of lattice-based identification protocols on smart cards and microcontrollers", Technical Report 078, ePrint IACR (2014).
11. Boorghany, A., Bayat Sarmadi, S., and Jalili, R. "On constrained implementation of lattice-based cryptographic primitives and schemes on smart cards", *ACM Transactions on Embedded Computing Systems (TECS)*, **14**(3), ACM, pp. 42:1-42:25 (2014).
12. Azarderakhsh, R., Liu, Z., Seo, H., and Kim, H. "NEON PQCryto: Fast and Parallel Ring-LWE Encryption on ARM NEON Architecture", Technical Report 1081, ePrint IACR (2015).
13. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness. http://competitions.cr.yp.to/caesar-call.html (2013).
14. McGrew, D. and Hoffman, P., *Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)*, RFC 7321, IETF (2014).
15. McGrew, D.A. and Viega, J. "The security and Performance of the galois/counter mode (GCM) of operation", *INDOCRYPT 2004*, LNCS, **3348**, Springer, pp. 343-355 (2005).
16. Whiting, D., Ferguson, N., and Housley, R. "Counter with CBC-MAC (CCM)", RFC 3610, IETF (2003).
17. Namprempre, C., Rogaway, P., and Shrimpton, T. "Reconsidering generic composition", *EUROCRYPT 2014*, LNCS, **8441**, Springer Berlin Heidelberg, pp. 257-274 (2014).
18. Kurosawa, K. and Iwata, T. "TMAC: two-key CBC MAC", *Topics in Cryptology – CT-RSA 2003*, LNCS, **2612**, Springer Berlin Heidelberg, pp. 33-49 (2003).
19. Iwata, T. and Kurosawa, K. "OMAC: One-Key CBC MAC", *Fast Software Encryption*, LNCS, **2887**, Springer Berlin Heidelberg, pp. 129-153 (2003).
20. Black, J. and Rogaway, P. "A block-cipher mode of operation for parallelizable message authentication", *EUROCRYPT 2002, LNCS*, **2332**, Springer Berlin Heidelberg, pp. 384-397 (2002).
21. Jutla, C.S. "Encryption modes with almost free message integrity", *EUROCRYPT 2001*, LNCS, **2045**, Springer Berlin Heidelberg, pp. 529-544 (2001).
22. Rogaway, P., Bellare, M., and Black, J. "OCB: A blockcipher mode of operation for efficient authenticated encryption", *ACM Trans. Inf. Syst. Secur.*, **6**(3), ACM, pp. 365-403 (2003).
23. Rogaway, P. "Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC", *ASIACRYPT 2004*, LNCS, **3329**, Springer, pp. 16-31 (2004).
24. Krovetz, T. and Rogaway, P. "The software performance of authenticated-encryption modes", *Fast Software Encryption*, LNCS, **6733**, Springer, pp. 306-327 (2011).
25. Minematsu, K. "Parallelizable rate-1 authenticated encryption from pseudorandom functions", *EUROCRYPT 2014*, LNCS, **8441**, Springer Berlin Heidelberg, pp. 275-292 (2014).
26. Banerjee, A., Peikert, C., and Rosen, A. "Pseudorandom functions and lattices", *EUROCRYPT 2012*, LNCS, **7237**, Springer, pp. 719-737 (2012).
27. Banerjee, A., Brenner, H., Leurent, G., Peikert, C., and Rosen, A. "SPRING: fast pseudorandom functions from rounded ring products", *Fast Software Encryption*, LNCS, **8540**, Springer Berlin Heidelberg, pp. 38-57 (2014).
28. Bellare, M. and Rogaway, P. "Entity authentication and key distribution", *CRYPTO 1993*, pp. 232-249. Springer (1993).
29. Bellare, M., Kilian, J., and Rogaway, P. "The security of cipher block chaining", *CRYPTO 1994*, Springer-Verlag, pp. 341-358 (1994).
30. Bellare, M., Guérin, R., and Rogaway, P. "XOR MACs: New methods for message authentication using finite pseudorandom functions", *CRYPTO 1995*, Springer Berlin Heidelberg, pp. 15-28 (1995).
31. Grover, L.K. "A fast quantum mechanical algorithm for database search", *Twenty-Eighth Annual ACM Symposium on Theory of Computing*, USA STOC '96, New York, NY, ACM, pp. 212-219 (1996).
32. Kaplan, M., Leurent, G., Leverrier, A., and Naya-Plasencia, M. "Breaking symmetric cryptosystems using quantum period finding", *CRYPTO 2016*, LNCS, **9815**, Springer Berlin Heidelberg, pp. 207-237 (2016).
33. Kuwakado, H. and Morii, M. "Security on the quantum-type even-mansour cipher", *International Symposium on Information Theory and Its Applications (ISITA)*, pp. 312-316 (2012).
34. Peikert, C. "Lattice cryptography for the internet", *Post-Quantum Cryptography*, LNCS, **8772**, Springer International Publishing, pp. 197-219 (2014).
35. Lyubashevsky, V., Peikert, C., and Regev, O. "On ideal lattices and learning with errors over rings", *J. ACM*, **60**(6), pp. 43:1-43:35 (2013).

36. Black, J. and Rogaway, P. "CBC MACs for arbitrary-length messages: The three-key constructions", *Journal of Cryptology*, **18**(2), pp. 111-131 (2005).

37. Luby, M. and Rackoff, C. "How to construct pseudorandom permutations from pseudorandom functions", *SIAM J. Comput.*, **17**(2), pp. 373-386 (1988).

38. Gligor, V.D. and Donescu, P. "Fast encryption and authentication: XCBC encryption and XECB authentication modes", *Fast Software Encryption*, LNCS, **2355**, Springer Berlin Heidelberg, pp. 92-108 (2001).

39. Rogaway, P. "Authenticated-encryption with associated-data", *9th ACM Conference on Computer and Communications Security*, USA CCS'02, New York, NY, ACM, pp. 98-107 (2002).

40. Rogaway, P. and Shrimpton, T. "A provable-security treatment of the key-wrap problem", *EUROCRYPT 2006*, LNCS, **4004**, Springer Berlin Heidelberg, pp. 373-390 (2006).

41. Bellare, M., Kilian, J., and Rogaway, P. "The security of the cipher block chaining message authentication code", *Journal of Computer and System Sciences*, **61**(3), pp. 362-399 (2000).

42. Iwata, T. and Kurosawa, K. "Stronger security bounds for OMAC, TMAC, and XCBC", *INDOCRYPT 2003*, LNCS, **2904**, Springer Berlin Heidelberg, pp. 402-415 (2003).

43. Bellare, M. and Namprempre, C. "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm", *Journal of Cryptology*, **21**(4), pp. 469-491 (2008).

44. Bellare, M., Desai, A., Jokipii, E., and Rogaway, P. "A concrete security treatment of symmetric encryption", *38th Annual Symposium on Foundations of Computer Science*, pp. 394-403 (1997).

45. Wegman, M.N. and Carter, J.L. "New hash functions and their use in authentication and set equality", *Journal of Computer and System Sciences*, **22**(3), pp. 265-279 (1981).

46. Damgard, I., Pedersen, T.B., and Salvail, L. "A quantum cipher with near optimal key-recycling", *CRYPTO 2005*, Springer Berlin Heidelberg, pp. 494-510 (2005).

47. Shoup, V. "Sequences of games: A tool for taming complexity in security proofs", Technical Report 332, ePrint IACR (2004).

48. Bellare, M. and Rogaway, P. "The security of triple encryption and a framework for code-based game-playing proofs", *EUROCRYPT 2006*, LNCS, 4004, Springer Berlin Heidelberg, pp. 409-426 (2006).

Biographies

Ahmad Boorghany received the BSc degree in Software Engineering and the MSc degree in Information Technology in 2010 and 2012, respectively, both from Sharif University of Technology, Tehran, Iran. He is now a PhD candidate in Computer Engineering at the Department of Computer Engineering, Sharif University of Technology. His research interests are lattice-based cryptography, post-quantum cryptography, provable security, and building efficient cryptographic schemes and protocols to be used in practice. He is a member of the International Association for Cryptologic Research (IACR).

Siavash Bayat-Sarmadi received the BSc degree from the University of Tehran, Iran in 2000, the MSc degree from Sharif University of Technology, Tehran, Iran in 2002, and the PhD degree from the University of Waterloo in 2007, all in Computer Engineering (hardware). He was with Advanced Micro Devices, Inc. for about 6 years. Since September 2013, he has been a Faculty Member at the Department of Computer Engineering, Sharif University of Technology. He has served on the executive committees of several conferences. His research interests include hardware security and trust, cryptographic computations, and secure, efficient and dependable computing and architectures. He is a member of the IEEE.

Rasool Jalili received his BS degree in Computer Science from Ferdowsi University of Mashhad in 1985, and MS degree in Computer Engineering from Sharif University of Technology in 1989. He received his PhD in Computer Science from The University of Sydney, Australia in 1995. He then joined the Department of Computer Engineering, Sharif University of Technology in 1995. He has published more than 140 papers in international journals and conference proceedings. He is now an Associate Professor, doing research in the areas of computer dependability and security, access control, distributed systems, and database systems in his Data and Network Security Laboratory (DNSL).