2021 Fifth Underwater Communications and Networking Conference (UComms) | 978-1-7281-9315-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/UCOMMS50339.2021.9598159

Physical Layer Security against an Informed Eavesdropper in Underwater Acoustic Channels: Reconciliation and Privacy Amplification

George Sklivanitis[1], Konstantinos Pelekanakis[2], Seçkin Anıl Yıldırım[2]
Roberto Petroccia[2], João Alves[2], and Dimitris A. Pados[1]

[1] Center for Connected Autonomy and AI, Florida Atlantic University, FL 33431, USA
[2] NATO STO Centre for Maritime Research and Experimentation, La Spezia 19126, Italy

***Abstract*—We propose a strategy for generating the same crypto-key between two trusted underwater acoustic nodes (Alice and Bob) without revealing it to an eavesdropper (Eve). Our work builds upon the results of [1] where a methodology for generating a string of bits for Alice, Bob and Eve based on channel feature extraction and quantization is discussed. In this paper, we aim to reconcile the respective bits of Alice and Bob while minimizing the information leaked to Eve. To this end, we examine various Reed Solomon (RS) codes and measure the reconciliation rate of Alice, Bob and Eve. Additionally, we propose the Secure Hash Algorithm-3 (SHA-3) as means to eliminate any information that Eve acquires during reconciliation. We evaluate our reconciliation and privacy amplification strategies with bits generated from real underwater acoustic channel probe exchanges between Alice and Bob and Bellhop-simulated channels for Eve. Our analysis confirms that appropriate combinations of channel features and RS codes lead to a computationally secure generation of a 256-bit crypto-key according to the principles of the National Institute of Standards and Technology (NIST), even if Eve is informed about the RS encoder and the SHA-3 function.**

I. Introduction

Underwater acoustic networks typically have an ad-hoc infrastructure and therefore symmetric cryptography, such as the Advanced Encryption Standard (AES) [2], is preferred for establishing confidential links among the nodes. In symmetric cryptography, a common key (i.e., a fixed-length string of bits) is pre-loaded among the communicating nodes that want to encrypt/decrypt their messages. This is a serious limitation when a new, key-less node must join the network. Solutions from the terrestrial domain [3], [4] are either based on some pre-shared secret or suffer from communication overhead. A novel approach that does not involve any pre-shared secrets is to independently generate a key over a two-way acoustic link based on physical layer channel characteristics [5].

We consider the scenario where Alice and Bob are two legitimate underwater acoustic nodes that wish to establish a confidential link, and Eve, is a passive eavesdropper. In [1], we discuss a methodology for extracting channel-based features in Alice, Bob and Eve and generating their respective quantized bits. Due to the lack of channel reciprocity, the extracted bits of Alice and Bob are not identical and this brings up the challenge of how to reconcile them without communicating much information over the channel that can be exploited by Eve. Existing information reconciliation techniques use either error correcting codes or some interactive information reconciliation protocol. The Cascade protocol uses a lossy quantization method and exchanges the indices relating to the parity bits between Alice and Bob as the reconciliation method [6]. In [7], Alice and Bob reconcile the effects of noise and channel variations in the quantized versions of their channel frequency response by mapping them to Bose–Chaudhuri–Hocquenghem (BCH) codewords. Other key reconciliation protocols include Low Density Parity Check (LDPC) [8], Cyclic Redundancy Check (CRC) [9] and Turbo coding [9].

In this paper, reconciliation is proposed via Reed Solomon (RS) codes due to their simplicity and effectiveness in correcting bit errors. The considered approach is to have Alice transmitting her parity bits to Bob and then Bob try to match his bit string with that of Alice. We test different combinations of channel features and RS code parameters and measure the successful reconciliation rate for Alice-Bob. We also show that Eve cannot reconcile to Alice and Bob despite the assumption that she can intercept Alice's parity bits and use Bob's decoder.

The aim of privacy amplification is to practically eliminate the amount of information Eve may have after the reconciliation process. To this end, we propose Alice and Bob to use the SHA-3 (Secure Hash Algorithm 3) [10] and map the reconciled bit string to a random sequence of 256 bits, which is the declared crypto-key. The benefit of using SHA-3 is that, even if Eve has the knowledge of the hash function, and possess a bit sequence with only one bit that is distinct from the reconciled bits of Alice and Bob, she will generate a completely different key. Finally, we show that the generated 256-bit crypto-key passes the randomness tests of the National Institute of Standards and Technology (NIST) test suite [11], hence we confirm its suitability for cryptographic applications such as the AES.

II. Reconciliation and Privacy Amplification

Let us denote $\mathbf{y}_A$ and $\mathbf{y}_B$ the L-bit strings of Alice and Bob, respectively, that are used for the crypto-key generation. In [1], we discuss a method that produces these random bits from quantizing four channel features. The objective here is to reconcile the differences between $\mathbf{y}_A$ and $\mathbf{y}_B$ while minimizing the amount of information leaked to Eve. To this end, we use an encoder taken from the family of RS(N,K) codes. Such an encoder takes as input $K = L/M$ symbols (M bits/symbol) and transforms it into a codeword of $N = 2^M - 1$ symbols. The first K symbols (or L bits) of the codeword are identical to the input while the remaining $S = N - K$ symbols correspond to the parity symbols. By adding S parity symbols to the data, an RS(N,K) code can detect (but not correct) any combination of up to and including S erroneous symbols, or locate and correct up to and including $T = \lfloor S/2 \rfloor$ erroneous symbols at unknown locations. Since $\mathbf{y}_A$ and $\mathbf{y}_B$ are expected to be correlated, our solution involves Alice encoding $\mathbf{y}_A$ and sending her parity symbols to Bob. Then, Bob combines $\mathbf{y}_B$ with the received parity symbols to reconcile with Alice. In addition, we assume that Eve can perfectly eavesdrop Alice's parity symbols and use the same RS decoder for reconciliation. Note that the parity symbols alone, i.e., the information leaked to Eve is not sufficient to recover $\mathbf{y}_A$.

Algorithm 1 describes the reconciliation protocol. First, the RS(N,K) encoder is selected based on the practical security constraint:

$$M \cdot (K - T) \geq 80. \quad (1)$$

This is because $M \cdot (K - T)$ bits is the portion of the input that cannot be corrected. Hence, the only way for Eve to predict $\mathbf{y}_A$ is to fire a brute force attack on $M \cdot (K - T)$ bits, which is computationally hard. Alice computes her parity symbols by encoding $\mathbf{y}_A$ via the RS(N,K) encoder and sends her parity symbols $\mathbf{s}_A$ to Bob. Then, Bob decodes $[\mathbf{y}_B, \mathbf{s}_A]$ via the RS decoder to derive $\widehat{\mathbf{y}}_A$. Bob's decoder finds error locations and magnitudes on the received N symbols and corrects up to T symbol errors to recover the original input $\mathbf{y}_A$. Note that the location of symbol errors does not affect the error correcting capability of the RS code, however the occurrence of burst errors in each block of M bits (i.e., one symbol) could affect the number of bit errors that can be corrected. The RS decoder uses the Berlekamp-Massey algorithm to compute error locator polynomial [12], [13], Chien search to find its roots [14], and the Forney algorithm to find exact error values [15].

After reconciliation, the last step is to amplify the privacy of $\mathbf{y}_A$. Both Alice and Bob use the SHA-3 to practically eliminate the partial information that the eavesdropper may have about the legitimate key and to minimize the correlation among the bits of the reconciled message. The SHA-3 is based on an instance of a family of sponge functions called KECCAK [10], which in turn is based on the sponge construction and is expressed as KECCAK[R, C], where R is the message block size in bits, and C the capacity. The default values are $C = 1024$ bits and $R = 576$ bits. The larger the value of R, the greater the rate at which message bits are processed by the sponge construction, while the capacity is a measure of the achievable complexity of the sponge construction and therefore the achievable level of security. A sponge function allows both variable length input and output.

For SHA-3, the n-bit hash values are 224, 256, 384 and 512 bits and have the following security properties:

Algorithm 1: RS-based reconciliation protocol

```
Select $K$, $M$, such that $K = L/M$ and
    $M \times (K - T) \geq 80$
    where $L$ =# of bits of $\mathbf{y}_A$, $N = 2^M - 1$ and
    $T = \lfloor (N - K)/2 \rfloor$
Alice: Encode $\mathbf{y}_A \rightarrow RS(N,K) \rightarrow [\mathbf{y}_A, \mathbf{s}_A]$
    Send parity symbols $\mathbf{S}_A$ to Bob
Bob: Append parity symbols $\mathbf{s}_A$ to $\mathbf{y}_B$
    Decode $[\mathbf{y}_B, \mathbf{s}_A] \rightarrow RS(N,K) \rightarrow \widehat{\mathbf{y}}_A$
Return: Bob reconciles to Alice if $\widehat{\mathbf{y}}_A = \mathbf{y}_A$
```

- Resistance against collision of approximately $n/2$ bits (i.e., an attacker should not be able to find a pair of messages $x \neq x'$ such that $H(x) = H(x')$ with less than about $2^{n/2}$ work);
- Preimage resistance of approximately n bits (i.e., an attacker given an output value y in the range of hash should not be able to find an input x from its domain so that $y = H(x)$ with less than about 2^n work);
- Second preimage resistance of approximately $n - L$ bits, where the length of the first preimage is at most $2L$ blocks (i.e., an attacker given one message x should not be able to find a second message, x' to satisfy $H(x) = H(x')$ with less than about 2^{n-L} work).

Other than security, SHA-3 is cheap to implement in specialized hardware, e.g., FPGAs [16], [17], [18], and costs fewer Joules-per-byte than SHA-2 when implemented in hardware.

The last stage of privacy amplification checks whether the generated crypto-key is random according to the NIST test suite [11]. The NIST suite program is composed of 15 distinct tests, however only nine of them are typically applied [7]. The program's input is the crypto-key and its output is the p-value calculated for each test. If the p-value is greater than 0.01, the generated key is considered to be random with a 99% of confidence. We declare a successful key if all nine tests yield p-value greater than 0.01. In such an event, Alice and Bob have independently generated a crypto-key that can be used in cryptographic application of their choice (e.g., AES). In case the key fails to pass all nine tests, Alice and Bob have the option to use a different combination of channel features and RS code and repeat the steps of reconciliation and privacy amplification.

The current key-generation protocol assumes that Alice and Bob are reconciled after Alice sends her parity symbols. Alice and Bob understand that reconciliation has failed only after the first exchange of data, which is encrypted with different keys. In such an event, the current solution is to restart the process from the first step of channel feature estimation.

III. Results and Discussion

We evaluate the proposed reconciliation and privacy amplification protocols with channel-based features extracted from 897 acoustic probes exchanged over different ranges and environmental conditions during the Rapid Environmental

TABLE I
REED SOLOMON CODE PARAMETERS.

Bits per Symbol (M)	Input Size Symbols (K)	RS Codeword Symbols (N)	Parity Symbols (S)	Error Corr. (T)	Input Size in Bits (L)
6	28	63	35	17	168
7	48	127	79	39	336
7	52	127	75	37	364

Picture 2018 (REP18) sea trial [1]. Three assets were used to establish underwater acoustic links: the NRP Alm. Gago Coutinho hydrographic vessel (Bob), a Wave Glider (Bob) and a sea-surface floating buoy (Alice). The role and deployment details of each asset are discussed in [1]. Simulated channel impulse responses are generated for Eve based on the Bellhop ray tracer. The amplitudes of experimental and Bellhop-simulated baseband channel impulse responses are included in [1]. The entire dataset of 897 acoustic probes results in 43 batches, namely, one batch includes 20 probe exchanges for deriving the data-based quantization intervals and 2 probe exchanges to generate the quantization bit vectors. As discussed in [1], different channel features show different correlations between Alice, Bob and Eve. These correlations were quantified in terms of the average Bit Disagreement Ratio (BDR), which depends on the channel variability and the probe exchange duration. With the aim towards developing an adaptive way to select channel features for a real-life implementation, our analysis here studies the reconciliation of different combinations of channel features with different RS codes. This is also instructive as different combinations will generate a different number of keys. Clearly, the larger the number of keys, the more efficient the key generation algorithm. Hence, based on the RS selection criterion in (1), we test the following combinations:

1) delay spread, channel sparseness, L2-norm, and L0-norm channel features with RS(63,28);
2) delay spread, channel sparseness and L2-norm channel features with RS(127,48);
3) L2-norm and channel sparseness channel features with RS(127,52).

Table I summarizes the error correcting capability T of the selected RS codes based on the symbol size M bits/symbol, the input length K symbols and the RS codeword length N. Note that combination 1 involves four channel features yielding 28 bits (7 bits/feature [1]) per probe exchange or 56 bits per batch while the RS(63,28) encoder takes as $\mathbf{y}_A$ input 168 bits. Clearly, three batches need to be concatenated in order to fill the encoder input. Similarly, combination 2 requires the concatenation of eight batches since it involves three channel features (42 bits per batch) and 336 bits as encoder input. Finally, combination 3 requires the concatenation of 13 batches since it involves two channel features (28 bits per batch) and 364 bits as encoder input.

Figures 1-3 evaluate the proposed reconciliation strategy in terms of the Successful Reconciliation (SR) vs. batch number. The boolean parameter SR $\in \{0, 1\}$ is used to flag successful reconciliation for the considered three combinations. Additionally, Figs. 1-3 show the computed BDR per batch (defined as the number of disagreeing bits between a pair of nodes divided by the length of the message in bits L) and the average BDR over the total number of batches. The notation Alice-Eve means that Eve eavesdrops Bob's channel probes, while the notation Bob-Eve denotes that Eve eavesdrops Alice's channel probes. In both cases, the parity symbols are transmitted by Alice.

Figure 1 shows that Bob successfully reconciles to Alice two out of the fourteen reconciliation attempts, i.e., two keys will be checked during privacy amplification. In Fig. 2, we

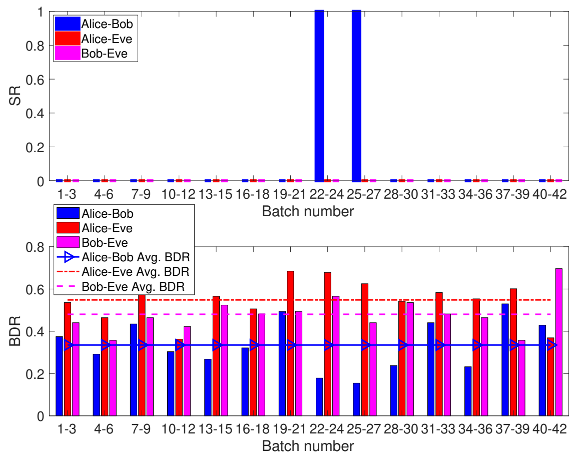

Fig. 1. SR vs. batch no. (top) and BDR vs. batch no. (bottom) for Alice, Bob and Eve with RS(63,28). SR:0 denotes failure to reconcile. Reconciled message (SR:1) was generated based on four features: *delay spread*, *channel sparseness*, *L2-norm*, *L0-norm*.

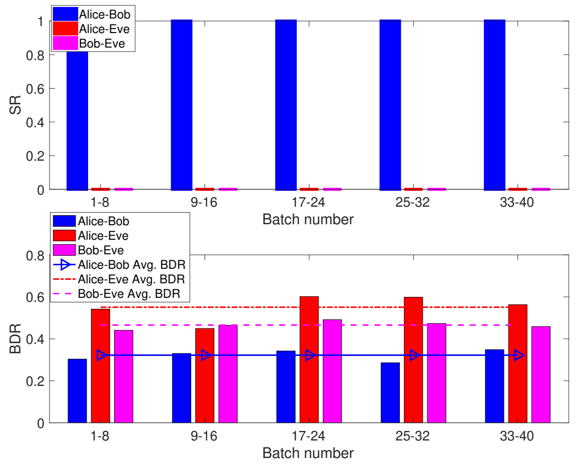

Fig. 2. SR vs. batch no. (top) and BDR vs. batch no. (bottom) for Alice, Bob and Eve with RS(127,48). SR:0 denotes failure to reconcile. Reconciled message (SR:1) was generated based on three features *delay spread*, *channel sparseness*, *L2-norm*.

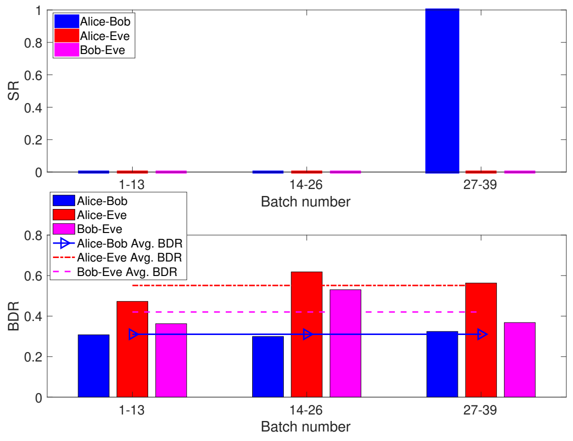

Fig. 3. SR vs. batch no. (top) and BDR vs. batch no. (bottom) for Alice, Bob and Eve with RS(127,52). SR:0 denotes failure to reconcile. Reconciled message (SR:1) was generated based on two features: *L2-norm*, *channel sparseness*.

TABLE II
NIST STATISTICAL TEST SUITE RESULTS.

NIST Test	Four Features RS(63,28)	Three Features RS(127,48)	Two Features RS(127,52)
Frequency	0.580	0.463	0.802
Block Frequency (Block length: 128)	0.795	0.490	0.939
Runs	0.539	0.406	0.379
Longest Run of Ones	0.007	0.550	0.420
DFT	0.449	0.266	0.358
Serial (Block length:5)	0.524, 0.445	0.631, 0.756	0.515, 0.223
Approximate Entropy (Block length: 2)	0.549	0.446	0.669
Cumsum Forward	0.579	0.455	0.573
Cumsum Backward	0.661	0.593	0.803

observe that Bob successfully reconciles to Alice five out of five reconciliation attempts, therefore five keys will be checked at the next step of privacy amplification. This result is achieved due to the adoption of a higher error correcting code RS(127,48) with parity size almost double than that of RS(63,28). Figure 3 shows that successful reconciliation is achieved only once for batches 27-39, therefore one crypto-key will be tested at the privacy amplification step. Note that the rate of SR is low given the number of exchanged probes. This was expected due to the fact that the probe exchange duration was long (10 seconds [1]) and there was considerable mobility. In retrospect, had we exercised a shorter exchange period, we would have seen a higher SR. Despite this issue, we note that for all combinations 1-3, Eve did not manage to reconcile with Alice and/or Bob. We tested three different RS codes with different number of features and different number of batches. We observe that RS(127,48) is the strongest RS code with T=39 symbols. It seems that for this dataset the error correcting capability of T=39 is critical below the limit which reconciliation becomes poor.

Privacy amplification of successfully reconciled messages, i.e., two messages for RS(63,28), five messages for RS(127,48), one message for RS(127,52) is carried out with the SHA-3, where the length of the output is fixed to 256 bits. Hence, combination 1 generates two keys, combination 2 generates five keys and combination 3 generates one key. We evaluate the effectiveness of SHA-3 using the NIST statistical test suite [19] comprising nine tests that generate a probability value (p-value) for each individual test. Table II shows the corresponding p-values of the nine tests. For RS(63,28) and RS(127,48), the p-values are averaged across the two and five keys, respectively. Note that the two keys generated by the four channel features fail to pass the "longest run of ones" test. Hence, the keys from combination 1 are not declared valid. However, the six keys generated by combination 2 and 3 pass all nine tests and are considered valid to be used for data encryption. As a result, the ability to have an on-the-fly adaptation of the RS input leads to time and power savings since there is no need always to start the key generation process from the first step of channel-features estimation. Additionally, since Eve did not manage to reconcile, due to the SHA-3 imposed security, there is no better strategy for her other than firing a brute force attack on the 256-bit crypto-key, which is computationally infeasible with the current state-of-the-art computing power.

IV. CONCLUSIONS

We studied the last two steps of the physical layer crypto-key generation strategy, namely, the reconciliation and the privacy amplification. Our study was based on the fact that Alice, Bob, and Eve had already extracted their quantized bits from their respective channel features. We proposed RS error correction at the reconciliation step and the process involved Alice sending her parity bits to Bob. We tested reconciliation using three different RS codes that were applied on three different combinations of channel features. For all combinations, we showed that Alice and Bob reconciled eight times within the entire data set while Eve did not manage to reconcile despite her knowledge about the RS decoder. The privacy amplification step relied on the SHA-3 whose input was the eight reconciled bit vectors and the output was the corresponding 256-bit crypto-keys. Eve's knowledge about the SHA-3 could not help her to gain a better strategy than a brute force attack on each generated crypto-key. Finally, to confirm the suitability of each key for cryptographic applications, we computed the p-values of the NIST test suite. We found that six out of eight keys were valid to encrypt data.

ACKNOWLEDGMENT

This work is funded by NATO Allied Command Transformation (ACT) Future Solutions Branch under the Autonomous Security Network Programme and the Office of Naval Research Global under grant no N62909-17-1-2093. The REP18-Atlantic sea trial was co-organized by the Portuguese Navy, FEUP, and CMRE. The authors would like to thank the Captain and crew of the NRP Gago Coutinho for the excellent support during the experiments. The work of D. A. Pados and G. Sklivanitis was supported in part by the National Science Foundation under Grant CNS-1704813 and the Schmidt Family Foundation, Boca Raton, FL.

REFERENCES

[1] K. Pelekanakis, S. Yildirim, G. Sklivanitis, R. Petroccia, J. Alves, and D. Pados, "Physical Layer Security against an InformedEavesdropper in Underwater Acoustic Channels: Feature Extraction and Quantization," in *2021 IEEE Fifth Underwater Communications and Networking Conference (UComms)*, (accepted).

[2] F. I. P. S. P. 197, "Announcing the ADVANCED ENCRYPTION STANDARD (AES)," November 26 2001, United States National Institute of Standards and Technology (NIST).

[3] M. A. Simplicio, P. S. Barreto, C. B. Margi, and T. C. Carvalho, "A survey on key management mechanisms for distributed wireless sensor networks," *Computer Networks*, vol. 54, no. 15, pp. 2591–2612, 2010.

[4] S. Seo, J. Won, S. Sultana, and E. Bertino, "Effective key management in dynamic wireless sensor networks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 371–383, 2015.

[5] K. Pelekanakis, C. M. G. Gussen, R. Petroccia, and J. Alves, "Robust Channel Parameters for Crypto Key Generation in Underwater Acoustic Systems," in *OCEANS 2019 MTS/IEEE SEATTLE*, 2019, pp. 1–7.

[6] S. N. Premnath, S. Jana, J. Croft, P. L. Gowda, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy, "Secret Key Extraction from Wireless Signal Strength in Real Environments," *IEEE Transactions on Mobile Computing*, vol. 12, no. 5, pp. 917–930, 2013.

[7] Y. Huang, S. Zhou, Z. Shi, and L. Lai, "Channel Frequency Response-Based Secret Key Generation in Underwater Acoustic Systems," *IEEE Transactions on Wireless Communications*, vol. 15, no. 9, pp. 5875–5888, 2016.

[8] J. Etesami and W. Henkel, "LDPC code construction for wireless physical-layer key reconciliation," in *2012 1st IEEE International Conference on Communications in China (ICCC)*, 2012, pp. 208–213.

[9] A. Ambekar, N. Kuruvatti, and H. D. Schotten, "Improved method of secret key generation based on variations in wireless channel," in *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2012, pp. 60–63.

[10] "The SHA-3 cryptographic hash algorithm competition," http://csrc.nist.gov/groups/ST/hash/sha-3/index.html, Nov. 2007-Oct. 2012.

[11] NIST, "Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm," *Technical report, NIST*, 2007.

[12] E. Berlekamp, "Nonbinary BCH decoding (Abstr.)," *IEEE Transactions on Information Theory*, vol. 14, no. 2, pp. 242–242, 1968.

[13] J. Massey, "Shift-register Synthesis and BCH Decoding," *IEEE Transactions on Information Theory*, vol. 15, no. 1, pp. 122–127, 1969.

[14] R. Chien, "Cyclic Decoding Procedures for Bose-Chaudhuri-Hocquenghem Codes," *IEEE Transactions on Information Theory*, vol. 10, no. 4, pp. 357–363, 1964.

[15] R. E. Blahut, *Algebraic Codes for Data Transmission*. Cambridge University Press, 2003.

[16] M. A. Elmohr, M. A. Saleh, A. S. Eissa, K. E. Ahmed, and M. M. Farag, "Hardware implementation of a SHA-3 application-specific instruction set processor," in *2016 28th International Conference on Microelectronics (ICM)*, 2016, pp. 109–112.

[17] X. Wu and S. Li, "High Throughput Design and Implementation of SHA-3 Hash Algorithm," in *2017 International Conference on Electron Devices and Solid-State Circuits (EDSSC)*, 2017, pp. 1–2.

[18] M. Sundal and R. Chaves, "Efficient FPGA Implementation of the SHA-3 Hash Function," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2017, pp. 86–91.

[19] L. E. Bassham, A. L. Rukhin, J. Soto, J. Nechvatal, M. E. Smid, S. D. Leigh, M. Levenson, M. Vangel, N. A. Heckert, and D. L. Banks, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," *Special Publication (NIST SP)*, 2010.