Title: CryptoQNRG: A new Framework for Evaluation of Cryptographic Strength in Quantum and Pseudorandom Number Generation for Key-Scheduling Algorithms

Author(s): Saini, A - Tsokanos, A - Kirner, R – University of Hertfordshire

Address:
Department of computer science
School of Physics, Engineering and Computer Science
University of Hertfordshire, Hatfield, United Kingdom

Abstract

In a cryptosystem, a cipher's security is directly dependent on a key-schedule or *Key-Scheduling Algorithm* (KSA) or that is used for both encryption and decryption. The random-number-based KSA adds another layer of security and prevents hackers from performing cryptanalysis. Several previous studies have investigated the strength of a cipher's encryption process. The strength evaluation of the key scheduling process has received less attention, that can lead to weaknesses in the overall encryption process. This paper proposes a new framework consisting of cryptographic strength evaluation criteria for Random Number Generators (RNG) based KSAs. Our framework (CryptoQNRG) evaluates different key-schedules based on pseudorandom and quantum random number generators with a set of tests. There are test suites that compare the strength of KSAs for different block ciphers. To the best of our knowledge this is the first time that a framework is built to compare the strength of KSAs incorporating RNGs and various block ciphers. CryptoQNRG comprises of four tests: Frequency, Bit_Correlation, Bit_Interfold, and Bit_Entropy. The tests are used to explore cryptographic properties such as unpredictability, balance of bits, correlation, confusion, and diffusion in the subkeys generated by the RNG based KSA. We have evaluated the most common KSAs with different block ciphers and a significant outcome of the proposed framework is the distinction between strong and weak RNG-based KSAs.

Keywords: Pseudo Random Number Generator, Quantum Random Number Generator, Block Cipher, Key Schedule Algorithms

1. Introduction

In cryptography, encryption methods[1] and ciphers are used to ensure data security and prevent unauthorized access. An encryption algorithm combines plaintext with key information to generate a ciphertext, making the plaintext difficult for hackers to decipher. This process of encryption with a secret key can lead to a secure cipher by combining nonlinearity, propagation criteria, correlation, algebraic immunity and randomness [2][3].

As part of an encryption process, the strength of the *Key-Schedule Algorithm* (KSA) directly impacts the security of an encryption algorithm. The KSA expands the secret key and generates subkeys according to the number of rounds of encryption required for a particular cryptographic algorithm. This generation and expansion of the secret key into multiple subkeys increases the robustness and complexity of the KSA. One of the potential flaws in this approach is the vulnerability in the key schedule expansion, which is characterised by the algorithm's resistance to cryptoanalysis attacks [4][5][6].

Researchers have demonstrated that logical gates such as AND, NAND, XOR, and OR, as well as Boolean functions with symmetrical properties, can be used to achieve security in the key expansion [7]. In addition to that, quantum data generation by generating quantum random number bits can increase security.

The best RNGs generate more random and unpredictable data to make keys resilient against a cryptanalysis attack. Randomness is generated using an entropy source, which can be either pseudo or quantum-based [8]. Pseudorandom number sequences are associated with an initial input seed, whereas the Quantum Random Number Generator (QRNG) [9] ensures high entropy from a quantum origin such as light or thermal noise for example.

There are various studies related to QRNGs focusing on different features. Lunghi et al. [10] proposed a protocol for self-testing quantum random number generation, in which the user can monitor the entropy in real time. Hesong Xu et al.[11] proposed a QRNG using single-photon avalanche diodes (SPADs) that produces a quantum random number without any post-processing. Biasing is one of the critical features that researchers consider when generating a quantum number. R.C.Pooser et al. [12] used a tapered amplifier that consists of optical semiconductor devices and an array of random number registration techniques to create quantum-based random numbers. A photon arrival time selectively based high-quality bias-free QRNG was introduced by Jian-min Wang et al.[13].

Another aspect of quantum processes is the speed at which random numbers can be generated. Yu-Huai Li et al. [14] proposed quantum random number generation with an uncharacterized laser and sunlight that generates random numbers at 1 Mbps. Abellan et al. [15] proposed an ultra-fast quantum random number generation accelerated phase diffusion in a pulsed laser diode observing 43 Gbps.

Quantis [16], a QRNG developed by IDQ, generates random numbers using light consists of elementary "particles" called photons [17][18]. The device allows live verification of its operation and provides a high level of entropy without requiring a post-processing function to increase its entropy rate. QRNG [19] is considered superior to traditional random number generators, as their source of randomness is invulnerable to environmental perturbations such as temperature, voltage, or current and considered highly secure random number generators [20] [21].

An improvement in terms of RNG, from pseudo randomness to quantum randomness, will benefit a KSA. The focus of this article is to create a framework to evaluate the strength of cryptography KSAs generated by RNGs. In particular the concrete contributions of this work are:

- Formulation of a test suite to evaluate the strength of cryptography KSAs generated by RNGs. This test suite consists of four parts: a bit-frequency test, a bit-correlation test, a bit-interfold test, and a bit-entropy test.
- Applying the proposed criteria in the framework's test suite to a number of different block ciphers' key-scheduling algorithms. P-value is a measure of the statistical significance of a test result calculated by statistical computation. The test is based on statistics and provides a P-value and the probability of achieving a pass or fail result. A pass does not

indicate that the cipher is secure against all attacks, however, a failure suggests that the algorithm is highly susceptible to attacks.

In contrast to existing randomness tests such as CryptRndTest, NIST Random Number Generator test, Dieharder battery test, our proposed test suite evaluates the strength of the KSA's subkeys [35], [36], [37], [38]. The tests of our framework are designed to evaluate the main properties of a Key-schedule such as unpredictability, balance, confusion, diffusion and correlation. Moreover, our proposed work compares two different RNG-based key schedules of the same block cipher simultaneously to distinguish their strength.

The organisation of this paper is as follows: section 2 provides information about related work; section 3 introduces the structure and evaluation criteria of our framework; section 4 presents the different RNG-based key-scheduling algorithms we use in our tests. The test results and analysis are presented in section 5, while in section 6 the conclusion can be found.

2. Related Work

The key generation is a process that creates a key and expands it based on logical operations to encrypt plain text. The strength of the KSA [7] can be evaluated on the different properties of the key. In 2019, Hakim and Nusrom [22] proposed a new algorithm for scheduling subkeys in the L-block cipher, as the Niaz correlation test concluded that the LBlock's KSA generates keys with a high correlation. Kareem and Rahma [23] proposed a novel method for modifying the Twofish algorithm by implementing multi-level keys in the KSA to control the dynamic block bit sizes and multi-state tables. Using these keys allows for a greater complexity in the algorithm while incurring a relatively small amount of additional computational time. Sulaiman et al. [24] proposed an enhancement of Rijndael's KSA [35]. The analysis conducted by Sulaiman addresses the algorithm's shortcomings and optimises the KSA in terms of frequency and Strict Avalanche Criteria. Huang et. al. [25] modified AES's KSA by transposing its subkey matrix. According to the authors, the new KSA is immune to SQUARE, meet-in-the-middle, and related-key differential type attacks. Shahzadi et al. [26] proposed that 2D Chaotic maps enhance the strength of the generated keys in the KSA of RC5 algorithm, making it difficult for hackers to decrypt the data. Their KSA work targets resource-constrained environments and analyses the security mechanism for specific applications of critical clinical images.

The security of the key generation process starts with the generation of bits based on a random number. Sahmoud et al. [27] proposed generating distinct subkeys from the AES real key using a Pseudo-Random-Number-Generator (PRNG) and encrypting the block with each subkey of KSA. Their research focused on the two techniques: first, preventing predictions of obtaining the sub-key from an available one, and second, presenting an initialisation method to speed up sub-key generation. Maram et al. [28] also used a PRNG and proposed a dynamic key-dependent S-box in KSA that achieved a better Avalanche effect with a cryptography algorithm.

Rahul Saha et al. [29] took a different approach, proposing a Symmetric Random Function Generator (SRFG) as a cryptographic function generating randomness in the KSA of AES. The results indicate that their proposed work has a threefold improvement in terms of confusion property and avalanche effect over the original AES. In another study [30], the FORTIS algorithm developed by Vuppala et al. [22] for generating sub-keys of KSA was implemented on an FPGA and the authors analysed the algorithm's resistance to a side power channel attack. Gaetan Leurent et. al. [31] presented a new representation for the AES's KSA that efficiently combined the information from the first sub-key and the last sub-key to reconstruct the master key. Lauren et. al. [32] discussed the security properties of AES's KSA and its vulnerability to published attacks. Also, based on the information principle introduced by Claude Shannon, they proposed a faster and more secure KSA for generating subkeys in an AES cipher.

Afzal et al. [33] have recently published work that is closely related to our research. They proposed a Key-Schedule Evaluation Criterion (KSEC) to evaluate the cryptographic strength of subkeys of different KSA and establish a distinction between weak and strong keys using four statistical [34] tests.

They proposed a test suite consisting of a frequency test to analyse the balance of 0 and 1 bits, Bit Independence tests for confusion and diffusion property, Bitwise Uncorrelation tests for correlation among subkeys, and High/low-density key tests for testing randomness of the subkeys. Their test suite compares the strength of KSA of different block ciphers. To the best of our knowledge this is the first time that a framework is built to compare the strength of KSA based on RNGs and various block ciphers.

3. The CryptoQNRG framework

A KSA generates subkeys based on the input of the user-defined key. A RNG-based KSA adds another layer of security to the subkeys. CryptoQNRG has a series of test criteria in order to evaluate the strength of an RNG-based KSA.

Figure 1 illustrates an example of a basic scenario that we considered in our research. In this scenario personal data are stored on an internet server (cloud or a data center) so that they can be easily accessible from anywhere.

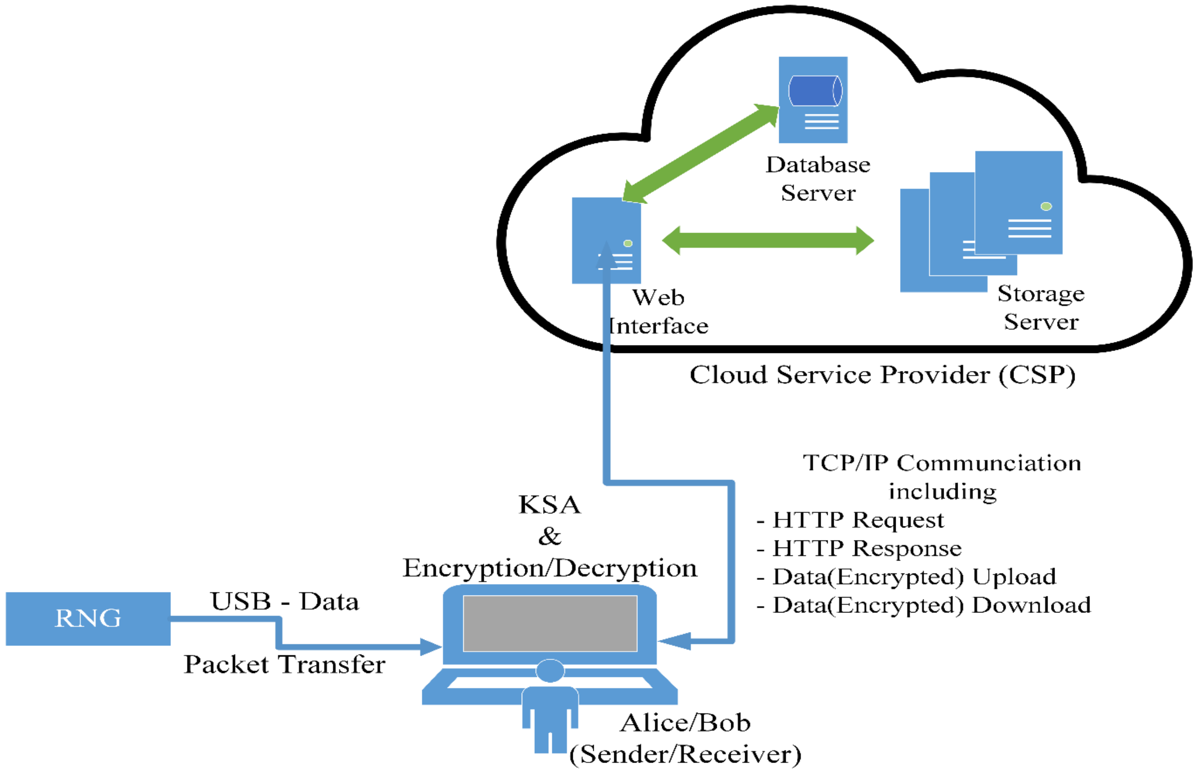

Figure 1 Basic scenario of a secure communication

The requirement is that the files must be encrypted before being sent over the network to make them secure, and when the user downloads them, they can be decrypted locally. Random Number Generator (RNG) plays a significant role as the KSA key comprises the bits of random numbers generated by an RNG to encrypt the contents.

Figure 2 shows the block diagram of CryptoQNRG for evaluating the strength of RNG-based KSA. The two keys, K_1 and K_2, are subkeys of individual KSAs generated using two different RNGs. The K_1 and K_2, are then passed to the proposed Key Schedule Evaluation Criterion – $KSEC_{RNG}(K_1, K_2)$ and tested for the cryptographic properties. The criterion includes four statistical tests. The first test, Frequency test $FT(K_1, K_2)$, calculates the frequency of 0's and 1's and checks how balanced their distribution is. The second test, Bit_Correlation $BCT(K_1, K_2)$ test, measures correlation, whereas, Bit_Interfold $BIT(K_1, K_2)$, the third test, checks for the confusion and diffusion properties. The last test is called Bit_Entropy $BET(K_1, K_2)$ test and examines the unpredictability of the bit stream generation. Based on the evaluation of each test, the strength of KSA is considered. The strength of KSA of K_1 and K_2 is strong if K_1 and K_2 pass all the tests of the $KSEC_{RNG}(K_1, K_2)$; otherwise, the KSA is weak. The proposed criterion, CryptoQNRG, which also compares K_1 and K_2 with the Bit_Entropy test.

The strength of KSA ($\mathrm{ST_{KSA}}$) and difference of KSA ($\mathrm{DF_{KSA}}$) are defined as follows:

$$\mathrm{ST_{KSA}} = \begin{cases} \text{Strong,} & if\ (K_1, K_2)\ pass\ each\ test\ in\ EC_{RNG}(K_1, K_2) \\ \text{Weak,} & otherwise \end{cases}$$

$$\mathrm{DF_{KSA}} = \begin{cases} K_1\ is\ better\ then\ K_2, & if\ \ K_1 > K_2\ in\ BET\ (K_1, K_2)\ tests \\ K_2, & otherwise \end{cases}$$

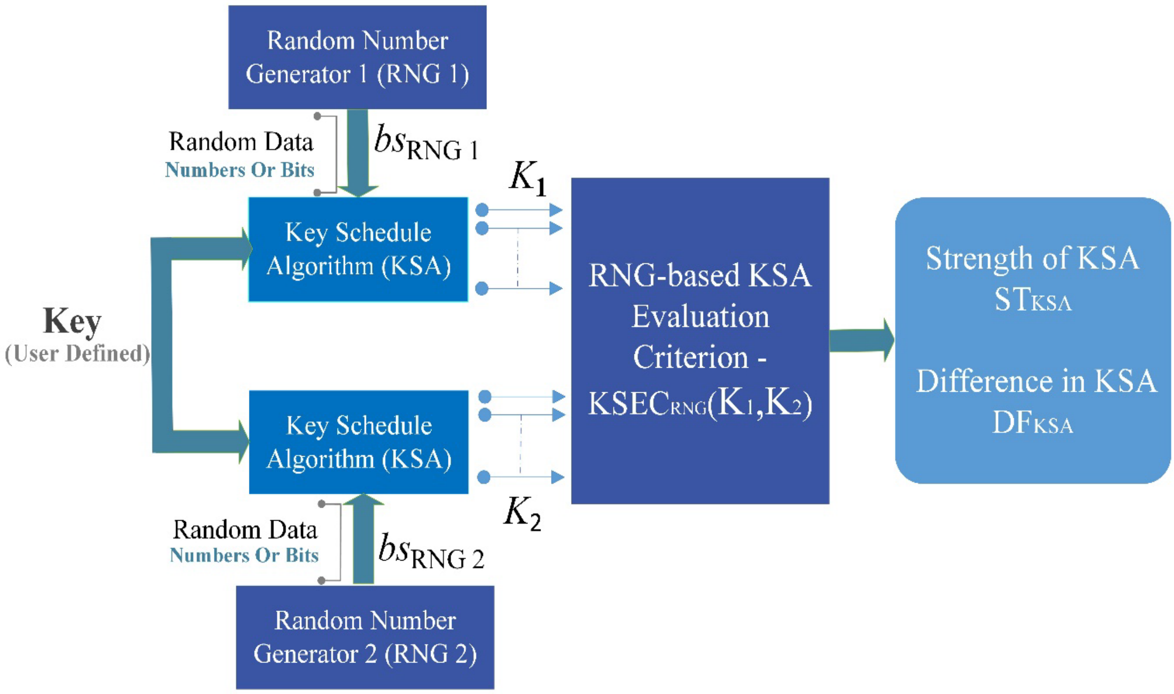

Figure 2 Block diagram of CryptoQNRG

We use the following notation in the equations below:

Key ... user-defined key

bs_{RNGi} ... the bitstring generated by RNG i, with $i \in \{1,2\}$

K_i ... the subkey of length L, obtained with the KSA based on RNG i using multiple iterations ($rounds$), with $i \in \{1,2\}$:

$$K_i = KSA(Key, bs_{RNGi}, rounds, L)$$

$K_{i,j}$ ... the j-th bit of subkey K_i: $1 \leq j \leq len(K_i)$ for $i \in \{1,2\}$

$len(K_i)$ ... number of bits in K_i , with $i \in \{1,2\}$

(note that for all keys K_i the length $len(K_i)$ $is\ even$)

a. *Frequency Test – $FT(K_1, K_2)$*

The first test to be performed is the frequency test which checks that the number of ones and zeroes in a sequence are the same in order to avoid biasing in the K_i. The test measures the distribution of bits in the RNG-based key-schedule algorithms. The sequence is a set of secret subkeys of different block ciphers. A subkey that fails the frequency test is considered weak as it fails the fundamental requirement of randomness, and there is no need to investigate other weaknesses based on the remaining tests. The K_1 and K_2 are balanced if it satisfies the key bits as;

$$FT(K_1, K_2) = \forall K_i \in \{K_1, K_2\}. \left| \bigcup_{K_{i,j} \in K_i \wedge K_{i,j} = 0} K_{i,j} \right| == \left| \bigcup_{K_{i,j} \in K_i \wedge K_{i,j} = 1} K_{i,j} \right| == \frac{len(K_i)}{2} \quad 1$$

where, K_i is the key obtained with the KSA based on RNG ($KSA_{RNG\ i}$) with $i \in \{1,2\}$

b. *Bit_Correlation Test – $BCT(K_1, K_2)$*

The second test is the Bit- Correlation test which measures the correlation of bits in the K_B. This evaluation is divided into two parts, the *Rogers-Tanimoto* distance measure $R(K_1, K_2)$ and the *Pearson's Correlation* r_{K_1,K_2}.

The first part computes the dissimilarity index between the two different RNG key-schedules with the *Rogers-Tanimoto* distance measure $R(K_1, K_2)$ [35]:

$$R(K_1, K_2) = \frac{T}{C_{11} + C_{00} + T} \quad 2$$

where, C_{pq} is the number of corresponding pairs of elements in K_1 and K_2 respectively equal to p and q.
and $T = 2(C_{10} + C_{01})$

The second part calculates the *Pearson's Correlation* r_{K_1,K_2} [36]. Based on that we test whether the key schedules K_1, K_2 are independent (null hypothesis H_0) or dependent (alternative hypothesis H_A).

The *Pearson's Correlation* r_{K_1,K_2} will be calculated as follows:

$$r_{K_1,K_2} = \frac{\sum_{j=1}^{len(K_i)} (K_{1,j} - \overline{K_1})(K_{2,j} - \overline{K_2})}{\sqrt{\sum_{j=1}^{len(K_i)} (K_{1,j} - \overline{K_1})} \sqrt{\sum_{j=1}^{len(K_i)} (K_{2,j} - \overline{K_2})}} \quad 3$$

where $\overline{K_i}$ is the mean value of $K_{i,j}$ with j = 1 ... $len(K_i)$.

Performing the Hypothesis Test based on the P-value generated by r_{K_1,K_2}:

- Null hypothesis H_0: K_1 and K_2 are dependent on each other

$$H_0 = (r_{K_1,K_2} \leq 0.1) \quad 4$$

- Alternative hypothesis H_A: K_1 and K_2 are independent of each other

$$H_A = (r_{K_1,K_2} > 0.1) \quad 5$$

The Bit_Correlation Test $BCT(K_1, K_2)$ combines the two parts and is calculated as follows:

$$BCT(K_1, K_2) = \begin{cases} Pass, & H_0 \wedge (R(K_1, K_2) \leq 0.5) \\ Fail, & otherwise \end{cases} \quad 6$$

The advantage of using $R(K_1, K_2)$ to compute dissimilarity is that it calculates the value of symmetric binary attributes. Symmetric binary attributes mean when both attributes have the same significance. It means the input to this test; both the bits 0 and 1 are equally significant. If K_1 and K_2 passes the frequency test, then only $R(K_1, K_2)$ is computed. The next part r_{K_1,K_2} determines the exact extent of the linear corelation between K_1 and K_2. Linear relationships occur when one variable change proportionally with another.

Therefore, $BCT(K_1, K_2)$ combines the linearity and dissimilarity test for K_1 and K_2. The K_1 and K_2 are considered to be acceptable if the dissimilarity index of the $R(K_1, K_2)$ is greater than or equal to threshold value of 0.5 and it is not linearly dependent on any each other subkey.

c. *Bit-Interfold Test* – $BIT(K_1, K_2)$

The third test is the Bit-Interfold test $BIT(K_1, K_2)$, which measures the confusion and diffusion in the K_1 and K_2, which is an important cryptographic property. This test is divided into two parts.

The first part is the calculation of the Hamming Distance $H(K_1, K_2)$ between the two subkeys $K_1$and K_2. The Hamming Distance is a dissimilarity distance [37].
$H(K_1, K_2)$ is calculated as the number of bit positions $K_{1,j}$ in K_1 that are different to those $K_{2,j}$ in K_2, divided by the subkey length:

$$H(K_1, K_2) = \frac{|\{K_{1,j} \mid K_{1,j} \neq K_{2,j} \wedge (0 \leq j < len(K_1))\}|}{len(K_1)} \quad 7$$

The Inverse Hamming Distance $\overline{H(K_1,K_2)}$, which is the inverse of $H(K_1,K_2)$, i.e., counting the number of equal bit positions, is calculated as follows:

$$\overline{H(K_1,K_2)} = len(K_1) - H(K_1,K_2) \tag{8}$$

$\overline{H(K_1,K_2)}$ refers to whether the bits' position will produce the same proportion of confusion and diffusion in K_1 and K_2. Confusion refers to the process of combining subkey bits with plain text make a cipher. Diffusion refers to the change in a plaintext resulting in changing the bit order in the subkeys K_1 and K_2. To analyse this proportion of similar bits leading to complex subkeys in K_1 and K_2 that are the basis of confusion and diffusion, the second part Z-Proportion [38] statistics hypothesis, is introduced.

In the second part, we use the calculated *Inverse Hamming Distance* $\overline{H(K_1,K_2)}$ to evaluate the Z-Proportion [38] statistics hypothesis. The analysis checks whether the confusion and diffusion will be similar or not by K_1 and K_2.

The Z-proportion test is calculated as follows:

$$Z(K_1,K_2) = \frac{\overline{H(K_1,K_2)} - k_0}{\sqrt{\frac{k_0(1-k_0)}{len(K_1)}}} \tag{9}$$

where k_0 is the hypothesized value of population proportion in the null hypothesis, i.e., it is the acceptance threshold of the Hamming Distance. n is the sample size.
The value of k_0 is 0.7 as at least 70 percent of bits differ in K_1 and K_2 to justify the bits responsible for confusion and diffusion, and KSA is considered strong.

The *Null Hypothesis* (H_0) and *Alternative Hypothesis* (H_A) based on a P-value computed by $Z(K_1,K_2)$ are as follows:

H_0 = Confusion and Diffusion is similar in K_1 and K_2.

$$H_0 = Z(K_1,K_2) \leq 0.7 \tag{10}$$

H_A = Confusion and Diffusion is not similar in K_1 and K_2.

$$H_A = Z(K_1,K_2) > 0.7 \tag{11}$$

The Bit_Interfold Test $BIT(K_1,K_2)$ is calculated as:

$$BIT(K_1,K_2) = \begin{cases} Pass, & H_A \\ Fail, & H_0 \end{cases} \tag{12}$$

The advantage of using $H(K_1,K_2)$ to compute dissimilarity is that it calculates the value of the exact number of different bits in K_1 and K_2. The inversion of $H(K_1,K_2)$, i.e., $\overline{H(K_1,K_2)}$ is given to the $Z(K_1,K_2)$ to analyse the proportion of bits responsible for generating similar confusion and diffusion by K_1 and K_2. K_1 and K_2 pass the Bit-Interfold Test if the confusion and diffusion with threshold value is not similar in both the KSAs, i.e., the $Z(K_1, K_2)$ results in *Alternative Hypothesis (HA).*

d. Bit-Entropy Test – $BET(K_1, K_2)$

The entropy is a measure of a random variable's uncertainty, and it plays a critical role in information theory. The higher the entropy, the greater is the uncertainty in predicting the value of an observation. There are various definitions available for entropy. In this work we use the Shannon entropy [39] (or *entr* for short) and the BiEntropy [40] (or *BiEn* for short).

The *entr* calculates the entropy as the amount of information conveyed when identifying a random outcome. The *BiEn* is a weighted average of the Shannon entropies of the string and the first n − 2 binary derivatives of the string.

The *entr* is advantageous for the larger value of binary strings, whereas the *BiEn* calculation is helpful for the smaller length of binary strings.

The *entr* $E(K_i)$ of K_1 and K_2, that takes values from the set A={$K_{i,j}$, $K_{i,j+1}$,..., $K_{i,n}$} with probability Pr (X=K_i) = $K_{i,j}$ for $i \in \{1,2\}$ (keys K_1, K_2) and $j \in \{1,2, \dots, len(K_1)\}$ is defined as:

$$E(K_i) = -\sum_{j=1}^{len(K_i)} p(K_{i,j})\, log_2(K_{i,j}) \qquad 13$$

The *BiEn* $BiEn(K_i)$ is calculated as follows for K_1 and K_2 distinctly. The *BiEn* value ranges from 0 to 1. When the disorder is more significant in a binary string, the *BiEn* value will be higher.

$$BiEn(K_i) = (1/(2^{n-1} - 1)) \left[\sum_{b=0}^{n-2} \left(-p(b).\, log_2\, p(b) - (1 - p(b)).\, log_2(1 - p(b)) \right).\, 2^b \right] \qquad 14$$

where, $p(b)$ is the proportion of 1's in K_1 and K_2.

The Bit-Entropy Test $BET(K_1, K_2)$ is calculated as follows:

$$BET(K_1, K_2) = \begin{cases} Pass, & E_{K_i} \geq 1.0 \land (BiEn(K_i) \geq 0.1) \\ Fail, & otherwise \end{cases} \qquad 15$$

The, $BET(K_1, K_2)$ combines E_{K_i} and $BiEn(K_i)$ to test entropy along with relative and disorder bits of any length in K_1 and K_2. The K_1 and K_2 are considered to be pass if the $E(K_i)$ is greater than threshold value of 0.1 and $BiEn(K_i)$is greater than 1.0.

e. RNG-based Key Schedule Evaluation Criterion - $KSEC_{RNG}(K_1, K_2)$

The four tests: Frequency-$FT(K_1, K_2)$, Bit_Correlation-$BCT(K_1, K_2)$, Bit_Interfold-$BIT(K_1, K_2)$ and Bit_Entropy-$BET(K_1, K_2)$ together form a test suite – $KSEC_{RNG}(K_1, K_2)$ - to evaluate the strength of the KSA based on RNG.

An RNG-based key schedule evaluation criterion $KSEC_{RNG}(K_1, K_2)$ is illustrated in Table 1. The table shows the required data generation for each test and the corresponding value for the threshold. The table also summarizes the cryptographic properties and the random keys associated with each test. During the test, a key size column specifies the length of the key in bits. In the next session, we have described the RNG-based KSA and data generation to evaluate their strength.

Table 1 Performed Tests with the RNG-based Key Schedule Evaluation Criterion – $KSEC_{RNG}(K_1, K_2)$ with Key Size $L = 2^N$ with $N \in \{6,7,8\}$

Test Type	Number of Keys	Cryptographic Property	Threshold level
Frequency			
Frequency	500	Balance of 0 and 1	$\frac{len(K_i)}{2}$
Bit_Correlation			
Rogers-Tanimoto	500	Correlation	0.5
Pearson Correlation			0.1
Bit_Interfold			
Hamming Distance	400	Confusion & Diffusion	0.7
Z Proportion			
Bit_Entropy			
BiEntropy: $BiEn(K_i)$	50	Unpredictability	0.1
entr: $E(K_i)$	500		1.0

In the next session, we have described the RNG-based KSA and data generation to evaluate their strength.

4. The RNG-based KSAs tested with CryptoQNRG

We analysed the KSA of the following five Block ciphers from a symmetric cryptosystem for an experimental purpose: AES, DES, CAST, Camellia, and GOST. The block cipher encrypts data in blocks of specified key size. The KSA key size taken is 128 bits for AES, Camellia, and CAST, 64 Bits for DES, and 256 bits for GOST. The sub keys are extended using the key expansion function of the KSA. These ciphers are proposed by different authors and with different KSA key size.

Rijndael [41] proposed an Advanced Encryption Standard (AES) with three variants AES-128, 192 and 256 based on KSA key length sizes of 128, 192, and 256 respectively, all of which were approved by National Institute of Standards and Technology(NIST). Endre Bangerter et al. [42] were able to recover AES-128 encryption keys in 2010. The second block cipher, DES [43] with KSA key size of 64 bits, is based on the Balanced Feistel structure and was proposed by IBM. Biham and Shamir [6] proposed a differential cryptanalysis attack on complete rounds of DES.

CAST [44], is based on the Feistel Network (FN) with a KSA key size of 40 to 128 bits. The fourth cipher, Camellia [45], was developed by Mitsubishi Electric and NTT of Japan, based on the FN algorithm with a KSA key size of 128, 192 or 256 bits. The final cipher, GOST [46], supports KSA key sizes of 256 bits. Nicolas Courtois et al. [47] proposed a Contradiction Immunity to attack the complete 32 - rounds of GOST cipher in 2011.

The RNG-based key-schedule also depends on different entropy based on its generator. In our set up we have taken CSPRNG and QRNG to evaluate the cryptographic strength. The entropy within the system is used to provide pseudo-random bits for the key-schedule (KSA$_{PK}$) that is required to create the keys. In QRNG, photons are used to generate quantum random bits for the key schedule (KSA$_{QK}$). The bits length of the generated random key depends on the key size of the KSA. The subkeys are generated using the cryptol [48] language. The results will demonstrate the strength of KSA in terms of cryptographic parameters for each block ciphers.

We use the same notation for result analyses:

K_1 ... the subkey obtained with the KSA based on QRNG

K_2 ... the subkey obtained with the KSA based on PRNG

To generate subkey K_1 of size L we take the bitstream of the QRNG and combine it with the user-defined key (Key) with multiple KSA iterations (rounds):

$$K_1 = KSA(Key, bs_{QRNG}, rounds, L)$$

Analogously, to generate subkey K_2 of size L we take the bitstream of the PRNG and combine it with the user-defined key (Key) with multiple KSA iterations (rounds):

$$K_2 = KSA(Key, bs_{PRNG}, rounds, L)$$

In our experiment we use KSA with 11 iterations ($rounds$) and subkey-size $L = 2^N$ with $N \in \{6,7,8\}$:

$$K_1 = KSA(Key, bs_{QRNG}, 11, 2^N) \quad | N \in \{6,7,8\}$$

$$K_2 = KSA(Key, bs_{PRNG}, 11, 2^N) \quad | N \in \{6,7,8\}$$

where the subkey-size $L = 2^N$ depends on the concrete block cipher length, so for DES we have $N = 6$ ($L = 2^6 = 64$), for AES, Camellia, and CAST we have $N = 7$ ($L = 2^7 = 128$), and for GOST we have $N = 8$ ($L = 2^8$ =256).

In this study, we use two sets of data, one for the Frequency and Bit_Entropy test, where we used random subkeys, and another set for Bit_Correlation and Bit_Interforld, where we used the samples of subkeys to test the hypotheses.

Finally, the key schedule evaluation criterion $KSEC_{RNG}(K_1, K_2)$ is used to evaluate the cryptographic strength of K_1 and K_2.

5. Results and Analysis

The results and data are covered in this section, where the result is drawn on the strength of the KSA$_{RNG}$. We have used the following hardware and software to implement the proposed framework.

Hardware: Quantis [49]– A USB-based Quantum random number generators developed by IDQ Its general specifications include - Random bit rate 1 : 4 Mbit/s ± 10% (Quantis-USB-4M), Thermal noise contribution: < 1% (Fraction of random bits arising from thermal noise), Storage temperature : - 25 to + 85°C, USB specification 2.0 and Power Via USB port

Computer - Processor: Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz, RAM: 8.00 GB, System type: 64-bit operating system, x64-based processor

Software: Java – Netbeans with JDK 1.8.0

a. *Frequency Test*

Table 2 displays the results of the frequency test. The number of bits is balanced in both K_1 and K_2. The numbers of bits in the Quantum K_1 and Pseudo-based K_2 key-schedule integrate the equal number of 0's and 1's. The table shows that the $FT(K_1, K_2)$ of each block cipher passes the frequency test. However, the frequency test alone cannot predict the strength of the RNG-based key-schedule.

Table 2 Frequency analysis $FT(K_1, K_2)$ of Bits in $\boldsymbol{K_1}$ and $\boldsymbol{K_2}$ Schedule of five block ciphers.

	AES	**Camellia**	**CAST**	**DES**	**GOST**
Ratio of Percentage of 0:1 in K_1	50:50	50:50	50:50	50:50	50:50
Ratio of Percentage of 0:1 in K_2	50:50	50:50	50:50	50:50	50:50

The Number of 0's and 1's is compared in K_1 and K_2 schedules of five different block ciphers. The statistical analysis shows the bits are balanced in K_1 and K_2 for all the ciphers.

b. *Bit – Correlation Test*

Bit correlation tests the strength in terms of the correlation while taking the Pearson's correlation hypothesis test in conjunction with the Rogers-Tanimoto distance measure. The graph in Figure 3 shows the changes in the Rogers-Tanimoto distance measure in K_1 and K_2. The results show that the dissimilarity index of CAST is the lowest of all, whereas the GOST key-schedule shows the highest. The index of DES is slightly less then Camellia and AES with 0.66589.

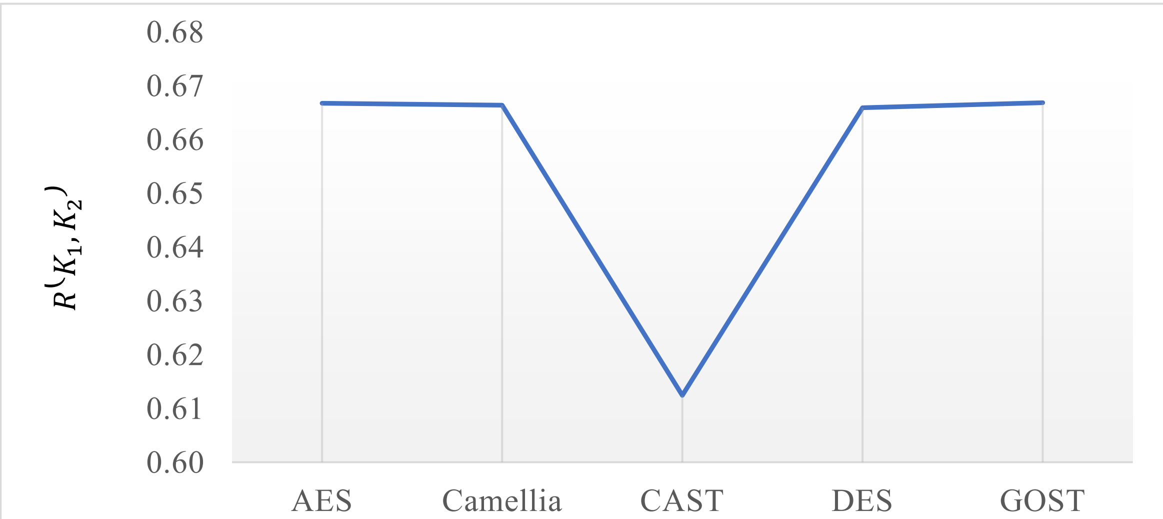

Figure 3 Rogers-Tanimoto distance measure $R(K_1, K_2)$ of five block ciphers.
The dissimilarity index of K_1 and K_2 is compared for five different block ciphers. Statistical analysis shows the bits are nearly thirty percent similar in both the key-schedule for four ciphers and forty percent in CAST; among all the ciphers.

:

Table 3 shows the Pearson's Correlation hypothesis testing result of each block cipher. The values of the r_{K_1,K_2} correspond to P-value statistics. The P-value of AES, Camellia, DES and GOST subkeys passes the threshold value of 0.1; therefore, we reject the null hypothesis that the K_1 and K_2 key-schedules are dependent on each other for these ciphers. On the other hand, CAST subkeys are failed to pass the threshold value. This means the K_1 and K_2 are dependent of each other and do not pass the Bit Correlation – $BCT(K_1, K_2)$ test. This shows that CAST keys are weak and susceptible for key-dependent [5] and correlation [50] attacks with both, Quantum and Pseudo random number-based key-schedules.

Table 3 Correlation of bits of K_1 and K_2 of five block ciphers based on Pearson's Correlation Hypothesis Test.

	AES	Camellia	CAST	DES	GOST
r_{K_1,K_2}	0.979	0.889	0.076	0.745	0.808
H_0 *or* H_A	Independent	Independent	Dependent	Independent	Independent

c. *Bit – Interfold Test*

Table 4 shows the results of the Bit-Interfold test. The test first calculates the Hamming Distance of the K_1 and K_2 based on a 64, 128 and 256-bit key size with a sample of 400 keys. The result of the Hamming Distance measures dissimilarity between the K_1 and K_2, and the inverse of which is then passed to one Z-Proportions hypothesis testing and the corresponding P-values are calculated.

Camellia and CAST result in the alternative hypothesis-H_A (taken from below result Table 3), which means they pass the Bit-interfold test. Any KSA that fails this test will create a weak cipher which is vulnerable to an easy cryptoanalysis. For example, AES, DES and GOST failed the $BIT(K_1, K_2)$ test and showed that they are weak and vulnerable to attacks such as related-key and side-channel [51] attacks.

Table 4 Confusion and Diffusion of K_1 and K_2 of five block ciphers based on Hamming Distance and Z - Proportion Hypothesis Test

	AES	Camellia	CAST	DES	GOST
$H(K_1, K_2)$	27661	28502	31676	25623	27461
$Z(K_1, K_2)$	H_0	H_A	H_A	H_0	H_0

d. *Bit – Entropy Test*

The most critical parameter for a key in order to be secure is unpredictability. The K_1 and K_2 are tested with two different entropy tests. The K_1 shows better entropy than the K_2, as shown in Figure 4 and Figure 5. The variations in entropy resulted in different values with the *entr* and Bi_Entropy tests, one for each K_1 and K_2, and same were analysed against the threshold value. The distinct entropy values of each block cipher exceed the threshold value of 1.0 for *entr* and 0.1 for Bi_Entropy; hence, all the K_1 and K_2 pass the entropy test – $BEnT(K_1, K_2)$.

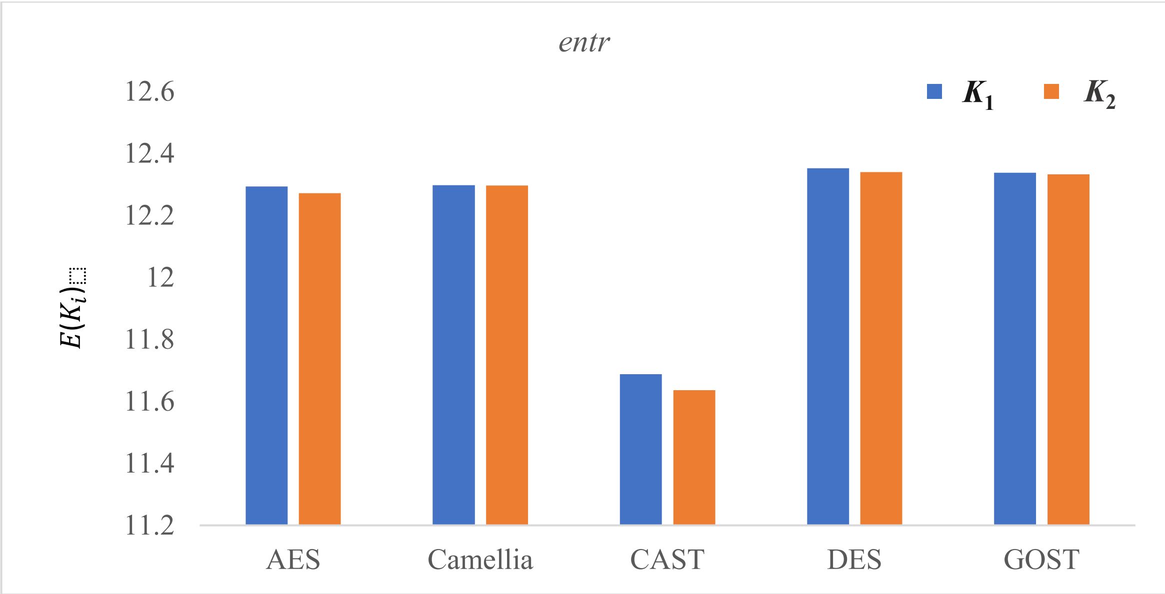

Figure 4 Entropy analysis entr $E(K_i)$ for K_1 and K_2 using five block ciphers.

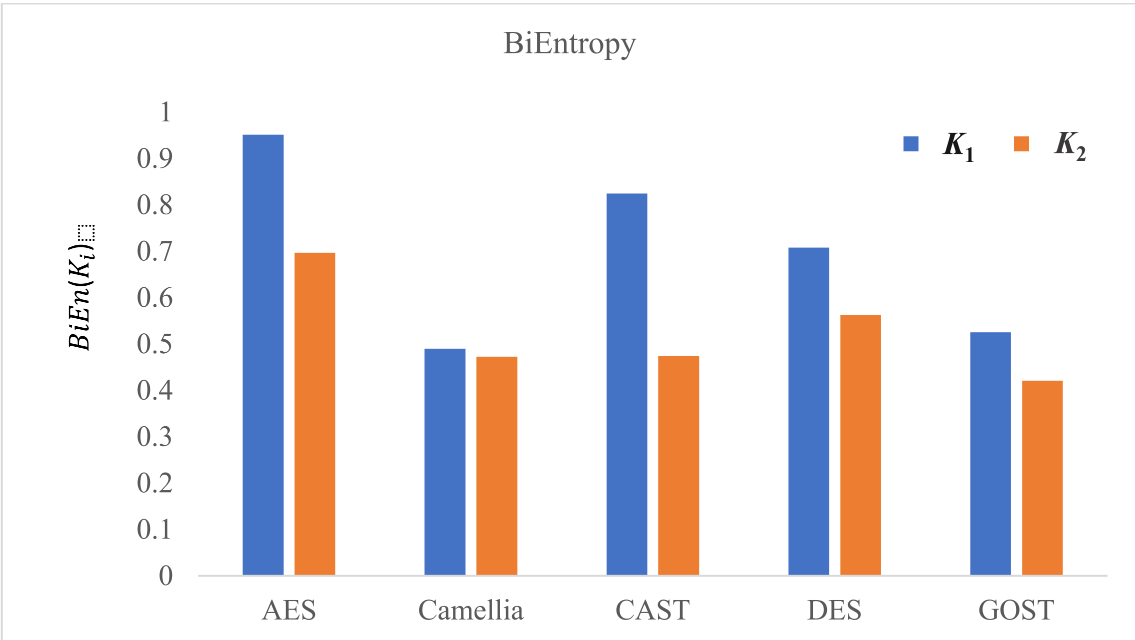

Figure 5 Bi_ Entropy analysis $BiEn(K_i)$ for K_1 and K_2 using five block ciphers.

The K_1 and K_2 of five different block ciphers are compared with the 500 and 50 different subkeys using two entropy tests: entr and Bi_Entropy. Statistical analysis shows that the key-schedule generated by quantum random bits are more unpredictable than pseudo random for all the block ciphers.

The analysis also proves that the quantum random number-based(K_1) key-schedule is more unpredictable than the pseudo-random(K_2) one. Unpredictability increases the K_1 schedule's strength, making it strong and hard to do cryptanalysis to partially access the key with Related-Key and Fault-Injection Attacks[52].

e. Encryption Time

The encryption time for all of the block ciphers was calculated to evaluate the impact of the K_1 and K_2. We used two different file sizes, 8 MB and 16 MB, to illustrate the time required to convert plain text to ciphertext. The encryption time for each file size can be seen in Figure 6. DES takes the longest time to encrypt, while GOST takes the second-longest time. The minimum time computation is by AES in both the K_1 and K_2 schedules. The analysis also shows that quantum and pseudo-based key-schedules are taking nearly the same time for encryption. All the ciphers showed nearly the same transformation time with K_1 and K_2, with AES taking the least time among all the ciphers for both the schedules.

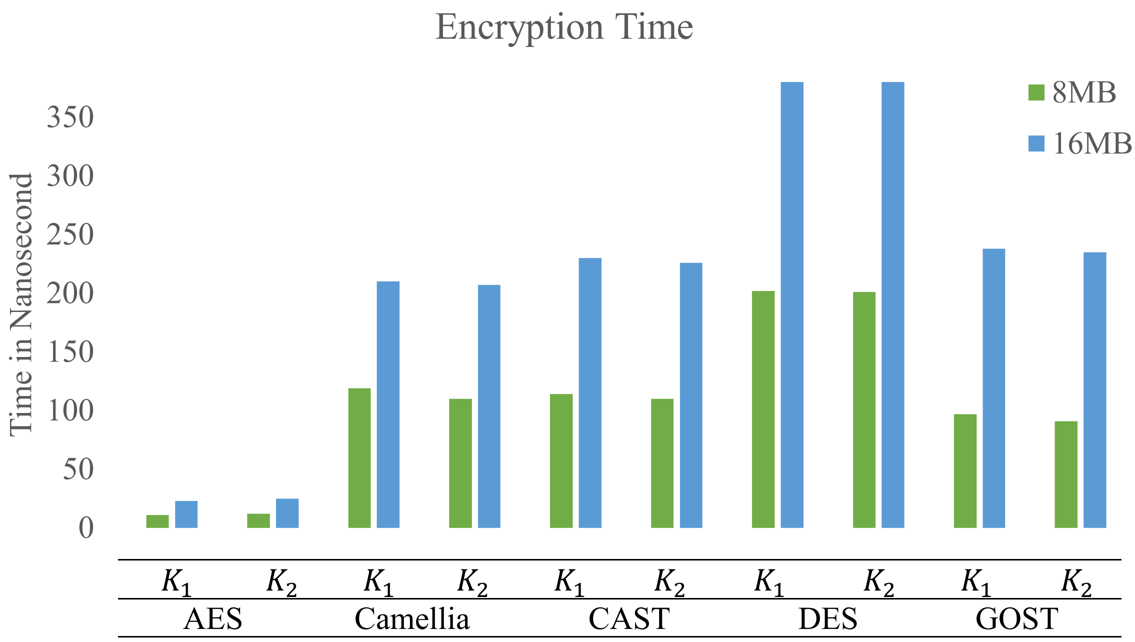

Figure 6 Encryption Time of plain text with K_1 and K_2 with file size of 8 MB and 16 MB. The time of converting plain text to cipher text with the help of Quantum and Pseudo based key-schedule is measured in nanoseconds.

6. Conclusion

We proposed CryptoQNRG, a new framework in order to evaluate the strength of RNG-based Key Schedules using four tests: Frequency, Bit_Correlation, Bit-Interfold, and Entropy. The test suite evaluates the resilience of subkeys of KSA in terms of the balance of 0 and 1's, correlation of bits, confusion and diffusion, and an essential parameter of security, uncertainty.

The proposed CryptoQNRG evaluates and assesses the subkeys of the most common KSA with quantum and pseudo-random numbers. The main focus of the paper is to compare the strength of KSA based on RNGs, as compared to Afzal et al. [33], who evaluated the subkeys without considering them. The results indicate the strength of Quantum- and Pseudo-based key-schedules and their cryptographic properties. The results show that CAST did not pass the Bit_Correlation test, and keys are prone to cipher attacks. The analysis also indicates that the AES, DES, and GOST did not pass the Bit-Interfold test, whereas CAST and Camellia did. However, the computational time required to generate a cipher with a quantum random number-based(K_1) key-schedule and pseudo-random(K_2) of AES is much faster than the

others. The results also revealed that a quantum-based key is less predictable than a pseudo-random number-based key.
The future work of the study includes testing the KSA of lightweight cryptographic algorithms that play a major role in the field of the internet of things (IoT).

Declarations

Ethical Approval: Not Applicable

Competing interests: The authors declare that they have no conflict of interest.

Authors' contributions: A.S. and A.T. devised the idea presented here. A.S. developed the theory, performed the computations, and prepared figures. R.K and A.S. verified the analytical methods. All authors reviewed the manuscript.

Funding: This research work received funding by University of Hertfordshire, United Kingdom.

Data Availability Statements: The data used to support the findings of this study are included in this article.

References

[1] W. Stallings, “Cryptography and Network Security: Principles and Practices,” *Cryptography and Network Security*. Pearson, USA, 2005.
[2] K. Verma and D. K. Sharma, “Calculation of non-linearity and algebraic degree of constructed boolean function,” in *2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, May 2017, pp. 501–505, doi: 10.1109/RTEICT.2017.8256647.
[3] F. L. Shi and H. Bin, “Propagation properties of symmetric Boolean functions,” in *International Conference on Intelligent Computation Technology and Automation*, May 2010, pp. 947–950, doi: 10.1109/ICICTA.2010.614.
[4] A. Biryukov and D. Khovratovich, “Related-key cryptanalysis of the full AES-192 and AES-256,” in *Advances in Cryptology – ASIACRYPT Lecture Notes in Computer Science*, Springer, 2009, pp. 1–18.
[5] K. B. Jithendra and T. K. Shahana, “New Results in Related Key Impossible Differential Cryptanalysis on Reduced Round AES-192,” in *2018 International Conference On Advances in Communication and Computing Technology, ICACCT 2018*, Feb. 2018, pp. 291–295, doi: 10.1109/ICACCT.2018.8529666.
[6] E. Biham and A. Shamir, “Differential cryptanalysis of DES-like cryptosystems,” *Journal of Cryptology*, vol. 4, no. 1, pp. 3–72, Jan. 1991, doi: 10.1007/BF00630563.
[7] N. P. Smart, V. Rijmen, B. Warinschi, and G. Watson, “Algorithms, Key Sizes and Parameters Report,” *Report*. ENISA, Nov. 2014, Accessed: Sep. 09, 2021. [Online]. Available: https://www.enisa.europa.eu/publications/algorithms-key-size-and-parameters-report-2014.
[8] J. Lee, Y. Seo, and J. Heo, “Analysis of random number generated by quantum noise source and software entropy source,” *In Proceedings of the International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, Jeju, Korea (South), pp. 729–732, Oct. 17, 2018, doi: 10.1109/ICTC.2018.8539618.
[9] M. Herrero-Collantes and J. C. Garcia-Escartin, “Quantum random number generators,” *Reviews of Modern Physics*, vol. 89, no. 1, p. 015004, Feb. 2017, doi: 10.1103/RevModPhys.89.015004.

[10] T. Lunghi *et al.*, "Self-Testing Quantum Random Number Generator," *Phys. Rev. Lett.*, vol. 114, no. 15, p. 150501, Apr. 2015, doi: 10.1103/PhysRevLett.114.150501.
[11] H. Xu, D. Perenzoni, A. Tomasi, and N. Massari, "A 16 × 16 Pixel Post-Processing Free Quantum Random Number Generator Based on SPADs," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 5, pp. 627–631, 2018, doi: 10.1109/TCSII.2018.2821904.
[12] R. C. Pooser, P. G. Evans, and T. S. Humble, "Self correcting quantum random number generators using tapered amplifiers," *In Proceedings of the IEEE Photonics Society Summer Topical Meeting Series*. IEEE, Waikoloa, HI, USA, pp. 147–148, Jul. 08, 2013, doi: 10.1109/PHOSST.2013.6614471.
[13] J. M. Wang, T. Y. Xie, H. F. Zhang, D. X. Yang, C. Xie, and J. Wang, "A bias-free quantum random number generation using photon arrival time selectively," *IEEE Photonics Journal*, vol. 7, no. 2, 2015, doi: 10.1109/JPHOT.2015.2402127.
[14] Y.-H. Li *et al.*, "Quantum random number generation with uncharacterized laser and sunlight," *npj Quantum Information*, vol. 5, no. 1. p. 97, Dec. 14, 2019, doi: 10.1038/s41534-019-0208-1.
[15] C. Abellán *et al.*, "Ultra-fast quantum randomness generation by accelerated phase diffusion in a pulsed laser diode," *Optics Express*, vol. 22, no. 2, p. 1645, 2014, doi: 10.1364/oe.22.001645.
[16] ID Quantique, "What is the Q in QRNG ?" 2020, Accessed: Jul. 07, 2020. [Online]. Available: https://www.idquantique.com/random-number-generation/overview/.
[17] G. Shaw, S. R. Sivaram, and A. Prabhakar, "Quantum Random Number Generator with One and Two Entropy Sources," *In Proceedings of the National Conference on Communications (NCC)*. IEEE, Bangalore, India, pp. 1–4, Feb. 20, 2019, doi: 10.1109/NCC.2019.8732222.
[18] G. Mogos, "Quantum Random Number Generator vs. Random Number Generator," in *IEEE International Conference on Communications*, Jun. 2016, pp. 423–426, doi: 10.1109/ICComm.2016.7528306.
[19] ID Quantique, "Understanding Quantum Cryptography." ID Quantique SA, 2020, Accessed: Jul. 07, 2020. [Online]. Available: https://www.idquantique.com/quantum-safe-security/quantum-key-distribution/.
[20] IDQ, "Quantum versus Classical Random Number Generators." Switzerland, May 2020.
[21] ID Quantique, "gaming-and-lotteries." Accessed: Jul. 07, 2020. [Online]. Available: https://www.idquantique.com/random-number-generation/applications/gaming-and-lotteries/.
[22] A. R. Hakim and Z. Z. Nusron, "An improved Lblock-s key schedule algorithm," in *International Conference on Information and Communications Technology*, Jul. 2019, pp. 232–236, doi: 10.1109/ICOIACT46704.2019.8938569.
[23] S. M. Kareem and A. M. S. Rahma, "A novel approach for the development of the Twofish algorithm based on multi-level key space," *Journal of Information Security and Applications*, vol. 50, Feb. 2020, doi: 10.1016/j.jisa.2019.102410.
[24] S. Sulaiman, Z. Muda, J. Juremi, R. Mahmod, and S. M. Yasin, "A New ShiftColumn Transformation : An Enhancement of Rijndael Key Scheduling," *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, vol. 1, no. 3, pp. 160–166, 2013.
[25] J. Huang, H. Yan, and X. Lai, "Transposition of AES key schedule," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10143 LNCS, pp. 84–102, 2017, doi: 10.1007/978-3-319-54705-3_6.
[26] R. Shahzadi, S. M. Anwar, F. Qamar, M. Ali, and J. J. P. C. Rodrigues, "Chaos based enhanced RC5 algorithm for security and integrity of clinical images in remote health monitoring," *IEEE Access*, vol. 7, Apr. 2019, doi: 10.1109/ACCESS.2019.2909554.

[27] S. Sahmoud, W. Elmasry, and A. Shadi, “Enhancement the Security of AES Against Modern Attacks by Using Variable Key Block Cipher,” *International Arab Journal of e-Technology*, vol. 3, no. 1, pp. 17–26, Jan. 2013.
[28] B. Maram and J. M. Gnanasekar, “A Block Cipher Algorithm to Enhance the Avalanche Effect Using Dynamic Key-Dependent S-Box and Genetic Operations,” *International Journal of Pure and Applied Mathematics*, vol. 119, no. 10, pp. 399–418, 2018.
[29] R. Saha, G. Geetha, G. Kumar, and T. H. Kim, “RK-AES: An Improved Version of AES Using a New Key Generation Process with Random Keys,” *Security and Communication Networks*, vol. 2018, pp. 1–11, Nov. 2018, doi: 10.1155/2018/9802475.
[30] A. Vuppala, R. S. Roshan, S. Nawaz, and J. V. R. Ravindra, “An Efficient Optimization and Secured Triple Data Encryption Standard Using Enhanced Key Scheduling Algorithm,” in *Procedia Computer Science*, Jun. 2020, vol. 171, pp. 1054–1063, doi: 10.1016/j.procs.2020.04.113.
[31] G. Leurent and C. Pernot, “New Representations of the AES Key Schedule,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12696 LNCS. pp. 54–84, 2021, doi: 10.1007/978-3-030-77870-5_3.
[32] L. May, M. Henricksen, W. Millan, G. Carter, and E. Dawson, “Strengthening the key schedule of the AES,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 2384, pp. 226–240, 2002, doi: 10.1007/3-540-45450-0_19.
[33] S. Afzal, M. Yousaf, H. Afzal, N. Alharbe, and M. R. Mufti, “Cryptographic Strength Evaluation of Key Schedule Algorithms,” *Security and Communication Networks*, pp. 1–9, May 2020, doi: 10.1155/2020/3189601.
[34] S. Afzal, U. Waqas, M. A. Mir, and M. Yousaf, “Statistical Analysis of Key Schedule Algorithms of Different Block Ciphers,” *Science International - Report*. Jun. 2015.
[35] M. M. Chatzimichailidou and I. M. Dokas, “RiskSOAP: On the Relationship between Systems Safety and the Risk SA Provision Capability,” *IEEE Systems Journal*, vol. 12, no. 2, pp. 1148–1157, 2018, doi: 10.1109/JSYST.2016.2614953.
[36] P. Socha, V. Miskovsky, H. Kubatova, and M. Novotny, “Optimization of Pearson correlation coefficient calculation for DPA and comparison of different approaches,” in *International Symposium on Design and Diagnostics of Electronic Circuit and Systems*, Apr. 2017, pp. 184–189, doi: 10.1109/DDECS.2017.7934563.
[37] T. S. Community, “hamming.” https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.hamming.html (accessed Jul. 09, 2020).
[38] E. Volchok, “Clear-Sighted Statistics: Module 14: One-Sample Hypothesis Tests (slides).” City University of New York (CUNY)., 2020.
[39] S. Vajapeyam, “Understanding Shannon’s Entropy metric for Information,” no. March, pp. 1–6, Mar. 2014, doi: https://doi.org/10.48550/arXiv.1405.2061.
[40] G. J. Croll, “Bientropy, TriEntropy and primality,” *Entropy*, vol. 22, no. 3, Mar. 2020, doi: 10.3390/e22030311.
[41] J. Daemen and V. Rijmen, *The Design of Rijndael*. Springer Berlin Heidelberg, 2002.
[42] D. Gullasch, E. Bangerter, and S. Krenn, “Cache games - Bringing access-based cache attacks on AES to practice,” in *IEEE Symposium on Security and Privacy*, May 2011, pp. 490–505, doi: 10.1109/SP.2011.22.
[43] A. Biryukov and C. Cannière, “Data encryption standard (DES),” *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, USA, Oct. 1999, doi: 10.1007/0-387-23483-7_94.
[44] C. Adams, “The CAST-128 Encryption Algorithm.” 1997, Accessed: Jun. 12, 2021. [Online]. Available: https://www.rfc-editor.org/info/rfc2144.

[45] "Japan's First 128-bit Block Cipher 'Camellia' Approved as a New Standard Encryption Algorithm in the Internet." NTT News Release, Accessed: Jul. 17, 2021. [Online]. Available: https://www.ntt.co.jp/news/news05e/0507/050720.html.
[46] C. Cannière, "GOST," *Encyclopedia of Cryptography and Security*. Springer US, Boston, MA, 2011, doi: 10.1007/978-1-4419-5906-5_579.
[47] N. T. Courtois, J. A. Gawinecki, and G. Song, "Contradiction Immunity and Guess-Then-Determine Attacks on Gost," *Tatra Mountains Mathematical Publications*, vol. 53, no. 1, pp. 65–79, Dec. 2013, doi: 10.2478/v10127-012-0039-3.
[48] "Cryptol." Galois, Inc., [Online]. Available: https://cryptol.net/.
[49] IDQ, "quantis-random-number-generator," 2020. https://www.idquantique.com/random-number-generation/products/quantis-random-number-generator (accessed Jul. 07, 2020).
[50] N. N. Anandakumar and S. Dillibabu, "Correlation power analysis attack of AES on FPGA using customized communication protocol," in *International Conference on Computational Science, Engineering and Information Technology*, Oct. 2012, pp. 683–688, doi: 10.1145/2393216.2393330.
[51] Y. Niu, J. Zhang, A. Wang, and C. Chen, "An Efficient Collision Power Attack on AES Encryption in Edge Computing," *IEEE Access*, vol. 7, pp. 18734–18748, 2019, doi: 10.1109/ACCESS.2019.2896256.
[52] Y. Li, M. Chen, Z. Liu, and J. Wang, "Reduction in the number of fault injections for blind fault attack on SPN block ciphers," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 2, pp. 1–20, Apr. 2016, doi: 10.1145/3014583.