Statistical Analysis of the MK-3 Customizable Authenticated Encryption

Peter Bajorski
School of Mathematical Sciences
Rochester Institute of Technology
pxbeqa@rit.edu

Alan Kaminsky
Parallel Crypto LLC
gagetch123@gmail.com

Michael Kurdziel
L3Harris Technologies
Mike.Kurdziel@L3Harris.com

Marcin Łukowiak
Department of Computer Engineering
Rochester Institute of Technology
mxleec@rit.edu

Stanisław Radziszowski
Department of Computer Science
Rochester Institute of Technology
spr@cs.rit.edu

***Abstract*—To provide security autonomy capability, such that different users can have independent variants of the encryption algorithm, MK-3 is designed to be customizable. Two levels of customization are supported, Factory Customization and Field Customization. Customization is done by modifying functions and function parameters in the algorithm to yield differing cipher functions while preserving the algorithm's security.**

The main goal of this work is to present the results from the statistical analysis of the customizable MK-3 encryption scheme, focusing on field customized mixers. We recall the main components of the MK-3 algorithm and overview a subset of available factory and field customizations for MK-3. We test the main instances of the field customized versions and give a general argument for their desired statistical properties expected from an encryption scheme.

***Index Terms*—Array-Based Statistical Analysis, Authenticated Encryption, Customizable**

I. Introduction

The MK-3 is an authenticated symmetric-key encryption scheme based on the duplex sponge construction, suitable for both hardware and software, but whose design features are targeted specially for hardware implementations [1], [2], [3], [4], [5]. The MK-3 scheme is a proprietary algorithm of L3Harris Technologies. MK-3 provides broad customization features, in the form of both factory customization and field customization capacity. This makes it well suited for government and military applications. The main goal of this paper is to present the results from the statistical analysis of the field customized version of the MK-3 encryption scheme. This is similar to, but expands on the analysis of the basic MK-3 scheme presented in [6]. Section 2 describes the design of the MK-3 cipher algorithm at a high level and provides an overview of the customization modes, their rationale, and their security autonomy. The same security claims are valid for all factory customizations, and for further algorithm customizations which can be easily adopted for the unique users. Section 3 provides an overview of the statistical analysis methodology used to test the MK-3 cipher (originally described in [6]) and the results of such tests are presented in Section 4. More specifically, we study randomness of the mapping between the key, plaintext, and ciphertext as defined by the MK-3 scheme using a previously developed array-based statistical method, which is more efficient than checking all outputs as one stream of bits. Within the array-based approach, we use frequency tests and serial tests. In conclusion: Each of the MK-3 cipher's output bits behaves as a uniformly distributed binary random variable, and these random variables are mutually independent. This holds true even when various customization modifications are made to the algorithm. These results ensure that the field customized MK-3 encryption algorithm effectively conceals its plaintext input. We conclude that the MK-3 block cipher's output ciphertext traffic streams look like random data regardless of the statistical behavior of any input.

II. Background

A. MK-3 Authenticated Encryption Algorithm

The MK-3 algorithm is a single pass authenticated encryption algorithm designed with hardware implementation in mind [1], [2], [3], [4], [5]. The core of MK-3 is a permutation function f comprised of 4 transformations, which are computed on a 512-bit state split into 16-bit words. This permutation function f is used in the duplex sponge construction [7] shown in Figure 1. The sponge is a relatively new primitive made popular by Keccak, the winner of the SHA-3 competition [8]. The sponge is special in that it can be configured to allow for a variety of different cryptographic uses. It has an internal state S comprised of b bits split into a rate r and capacity c, such that $b = r + c$. The basic sponge construction consists of two stages. Data is "absorbed" into the sponge by passing it through the underlying function f in r-bit length blocks. It is then "squeezed" out, generating output of arbitrary length specified at run time.

The MK-3 scheme allows keys K of length 128 and 256 bits. In both cases the rate r is kept at 128 bits and $c = 384$, which permits simple transitioning between the two key lengths. The operational difference in MK-3 between key sizes is the number of rounds N_r which are iterated while

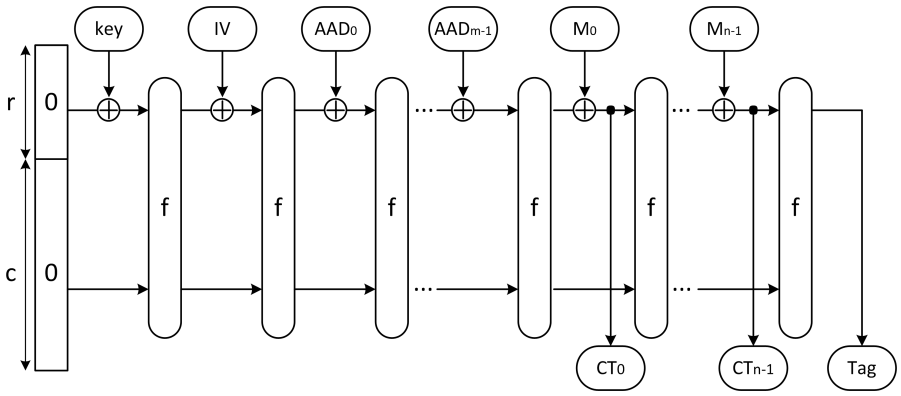

Fig. 1: The duplex construction.

computing f. Using a 128-bit key requires 10 rounds while 256-bit key requires 16 rounds.

The duplex sponge construction is a slight modification of the sponge which maintains state between calls while absorbing and squeezing the data during each iteration [7]. Figure 1 illustrates this approach in the context of the MK-3 authenticated encryption scheme, where Authenticated Encryption (AE) functionality can be achieved through successive duplexing calls with the key, initialization vector (IV), additional authenticated data (AAD_i) and blocks of the plaintext (M_i). Each absorbed r-bit plaintext block M_i is used to produce one ciphertext block CT_i, and the authentication tag T is outputted at the end. The details of permutation f, each round operation and overall security analysis are described in [1], [2], [9].

B. Security Autonomy

Sovereign cryptography refers to the capability that allows cryptography users to install their own cryptographic algorithm into a product after delivery and without U.S. Government or radio vendor involvement. This provides the customer with the capability of "Security Autonomy." Security Autonomy is defined as the ability to manage the security posture of an information system in a way that is independent of any third party. This includes not only independent operational management of the system, but also independent design and maintenance of the security elements of the system. The most significant challenge to providing sovereign cryptographic capability is policy related. The U.S. Department of State export policy regulates the international distribution of military-grade cryptographic technologies and specifically limits sovereign cryptography approaches in military communications equipment. Development and economic feasibility, deployment logistics, system security verification, and system reliability testing present additional challenges. For military applications, a Security Autonomy capability can also be provided by supplying users with their own customized versions of a proprietary cryptographic algorithm.

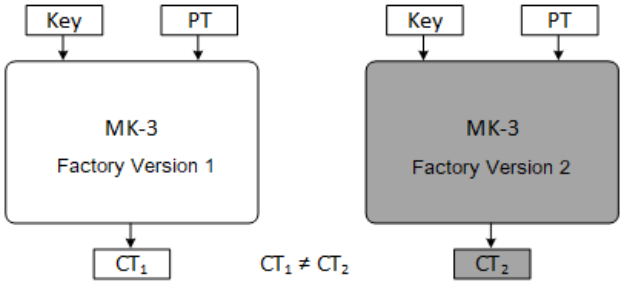

Fig. 2: Different factory customized versions yield noninteroperable ciphers; factory ensures the security of each customized version.

C. MK-3 Customization Implementation

The MK-3 cryptographic algorithm can be customized in various ways. This capability is provided with two types of customization, namely Factory Customization and Field Customization. Factory Customization is substantial in that the structure and major components of the cryptographic algorithm are modified. Proper Factory Customization and verification allows a custom algorithm variant to be provided with maximum security. Field Customization allows users to make changes to the cryptographic algorithm via a tool after the device is provided to them. All possible parameters that can be input into the system via the tool to provide the Field Customization are equally valid, and none can degrade the cryptographic strength of the algorithm. In addition, parameters for this mode of customization are known only to the user.

Customized versions of the MK-3 algorithm are implemented by altering the round function in specific ways designed to preserve the algorithm's security while ensuring that different customized versions of the algorithm are not interoperable. Factory customizations affect any or all steps of the round function. Field customizations affect the Mixing Step. This idea is illustrated in Figures 2 and 3. The customizations' full description was presented in [4]. Below we overview only parts discussed further in statistical analysis.

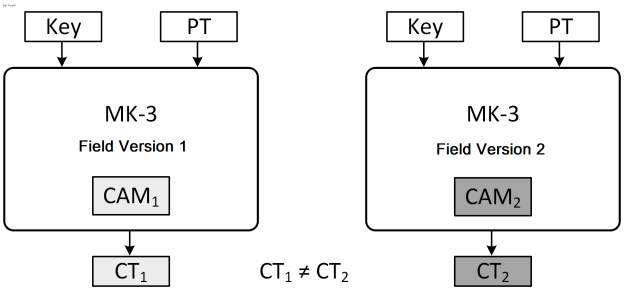

Fig. 3: Custom Algorithm Modification register (CAM) settings yield noninteroperable ciphers; no additional security analysis is needed.

1) Factory Customization: The MK-3 round function is designed to be customizable. The user cannot change the customized features once they are installed at the factory. The parameters and functions are designed to preserve the algorithm's security; consequently, each potential group of parameters and functions must be analyzed to ensure they meet security requirements.

2) Field Customization/Custom Algorithm Modification: MK-3 includes a 128-bit Custom Algorithm Modification register (CAM) that performs additional customization beyond the factory installed function and parameter settings. The CAM register's contents may be changed at any time during operation. All of the possible CAM values yield fully secure, yet noninteroperable customized versions of the MK-3 algorithm. The user can therefore pick any CAM value without needing to analyze the security of the resulting version. Each mixer in the Mixing Step treats its inputs as field elements in $GF(2^{16})$, that is, polynomials. The mixer's outputs A' and B' are computed from its inputs A and B as:

$$\begin{bmatrix} A' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & x \\ x & x+1 \end{bmatrix} \times \begin{bmatrix} A \\ B \end{bmatrix}. \tag{1}$$

The mixer's output computations involve field multiplications. A $GF(2^{16})$ field multiplication is the polynomial product of the field elements modulo an irreducible degree-16 polynomial $p(x)$. Kelly [1] specified the mixer's irreducible polynomial as $p(x) = x^{16} + x^5 + x^3 + x^2 + 1$.

However, the security analysis does not depend on the particular choice of the mixers' irreducible polynomial. Choosing a different irreducible polynomial will alter the output values computed by the above formulas, thereby altering the MK-3 round function's mapping, without affecting its security. The 128-bit CAM is partitioned into sixteen 8-bit sections, one section for each of the sixteen mixers in the round function. Each mixer's circuitry uses the value of the corresponding CAM section as an index into a precomputed 256-element irreducible polynomial table to obtain the irreducible polynomial coefficients for that mixer. Note that different mixers can be made to use different irreducible polynomials. Thus, changing the CAM setting in the field changes the mapping calculated by each mixer, yielding a customized version of the MK-3 algorithm. Because the CAM setting picks each mixer's polynomial out of a factory-specified table of irreducible polynomials, every possible CAM setting is guaranteed to yield a valid mixer mapping, see Figure 4.

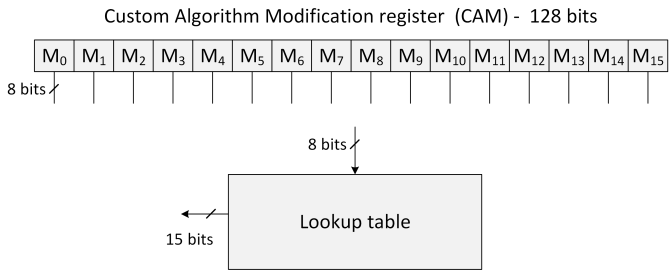

Fig. 4: CAM register stores 16 8-bit values. Lookup table stores information about 256 irreducible polynomials. Only 15 bits are needed to define each polynomial since bits 0 and 16 are set to 1.

III. Statistical Analysis

A. The Array-based Approach

In this section, we study randomness of the mapping between the key, plaintext, and ciphertext as defined by the customizable MK-3 scheme using a previously developed array-based statistical method described in [6]. We focus on the field customization of the Mixing Step.

In the process of checking randomness of cryptographic primitive functions, various inputs are selected, and then randomness of the resulting outputs is checked. This task is computationally intensive, since we want to check a large number of possible inputs. Hence, it is important to use efficient techniques, which, in turn, increases chances of detecting non-random behavior (that is, a pattern that is highly unlikely to occur by chance). The main purpose of the array-based approach is to improve the efficiency of this process. This approach was developed in [6] as a follow-up to some ideas presented in [10], [11]. We start with a simple scenario, where a cryptographic primitive function can be represented as a function of two inputs, the key and the plaintext, with the ciphertext being the output of that function. The two inputs will be denoted as $\mathbf{A}$ and $\mathbf{B}$, and the cryptographic function output will be written as $\mathbf{Y} = f(\mathbf{A}, \mathbf{B})$. Since $\mathbf{Y}$ consists of N bits, we will treat it as an N-dimensional vector of bits. The inputs $\mathbf{A}$ and $\mathbf{B}$ consist of K bits each and will be treated as K-dimensional vectors. In the array-based approach, we use sequences of vector inputs $\mathbf{A}_i, i = 1, \ldots, I$ and $\mathbf{B}_j, j = 1, \ldots, J$, and produce the outputs $Y_{n,i,j}$, for all possible pairs (i, j) and $n = 1, \ldots, N$, where $Y_{n,i,j}$ represents the n-th bit of the output for $\mathbf{A}_i$ and $\mathbf{B}_j$. The values of $Y_{n,i,j}$ can be arranged into a 3-dimensional array, and the statistical testing is done along various dimensions of that array. For example, we can fix the value of j and test randomness of the bit-array consisting of $N \cdot I$ bit positions

$$[Y_{n,i,j_0}]_{n=1,\ldots,N;i=1,\ldots,I}. \tag{2}$$

This testing would be repeated J times for all possible values of j_0. In a similar fashion, other dimensions can have a fixed value, and the corresponding bit-array is tested. Two dimensions can also be fixed, so that the set of bits for testing is selected based on changing the third dimension. For example, we could fix i at i_0 and j at j_0, and the bit positions for testing randomness would become

$$[Y_{n,i_0,j_0}]_{n=1,\ldots,N}. \tag{3}$$

Due to performing many statistical tests here, the critical values for randomness detection need to be adjusted for the multiple inference effect, which we do here using the Bonferroni approach. When D instances of hypothesis testing are performed, the probability of type I error of each individual test needs to be set to α/D in order to obtain the joint probability of one or more type I errors (in all D instances of hypothesis testing) that is not larger than α. Note that this means that any points up to the limit are entirely acceptable. On the other hand, individual points outside of the limits are expected in about 5% of plots. An issue with randomness would be detected if there were much more plots with points outside of the limits or there were many points outside of the limits in one single plot. See [12] for more details.

B. Data Generation Process

For the purpose of checking randomness of MK-3, we used the approach depicted in Figure 5 as the basic test setup of the sponge construction used in MK-3.

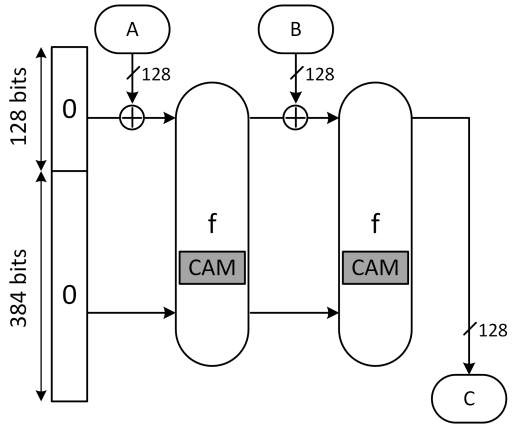

Fig. 5: Test setup for 128-bit key with CAM.

The inputs $\mathbf{A}$ and $\mathbf{B}$ are as described in the background section; the $\mathbf{C}$ output from the algorithm, consisting of 128 bits, is used to create each n-bit (i, j) element of the mathematical $\mathbf{Y}$ output. In order to generate a sequence of inputs $\mathbf{A}_i, i = 1, \ldots, I$, we started with a sequence of 128 zero bits as the $\mathbf{A}_1$ vector. Then we defined the $\mathbf{A}_2$ vector by flipping the first bit of $\mathbf{A}_1$ to 1, defined the $\mathbf{A}_3$ vector by flipping the second bit of $\mathbf{A}_1$ to 1, and so on, for all 128 bits of $\mathbf{A}_1$. This resulted in 129 vectors $\mathbf{A}_i$, where $i = 1, \ldots, 129$. Note that each $\mathbf{A}_i$ for $i > 1$ has exactly one bit equal to 1. The inputs $\mathbf{B}$ were defined in the same way, so $\mathbf{B}_i = \mathbf{A}_i$ for $i = 1, \ldots, 129$.

To summarize, our data can be written as an array of bits $Y_{n,i,j} = f_n(\mathbf{A}_i, \mathbf{B}_j)$, where $n = 1, \ldots, 128$; $i, j = 1, \ldots, 129$.

The MK-3 algorithm is designed to allow customization by its user; this is done by providing Custom Algorithm Modification (CAM) values as additional parameters. As described in Section II-C2, the CAM register controls the selection of degree-16 irreducible polynomials used as the moduli in each of the 16 mixers in the algorithm's Mixing Step. In total, there exist 4080 such polynomials, each of which may be used in any of the 16 mixers. In order to perform more comprehensive statistical testing, our tests specified each mixer's polynomial directly, instead of using any particular factory-customized subset available through the 128-bit register described earlier. Therefore, in the statistical analysis, a "CAM" refers to a selection of 16 values from 1 to 4080, each of which indicates the index of polynomial to use in its corresponding mixer. All statistical tests were performed using each of 4335 $(= 4080 + 255)$ different CAMs. The first 4080 CAMs used for testing are simply the CAM number repeated 16 times. The additional 255 CAMs are each composed of 16 randomly-selected polynomial indexes.

IV. Results

A. Frequency Tests

The frequency tests check whether 0 bits and 1 bits occur in equal proportions in a given binary sequence. More specifically, we test the frequency of 1's against the expected statistical variability.

In the following graphs, the horizontal dashed lines mark the limits of the acceptable statistical variability, which were calculated as

$$0.5 \pm z(1 - \alpha/(2k)) \tag{4}$$

for $s = \sqrt{0.25/n}$, where n is the number of bits being tested for the frequency of 1's, k is the number of statistical inferences being performed, and $z(\alpha)$ is the α-level percentile from the standard normal distribution. The specific values for n and k vary depending on which test is performed.

The above limits are calculated based on the normal approximation of the thresholds for rejection of the null hypothesis that the probability of observing the value 1 is equal to 0.5. We use k, the number of statistical inferences being performed, as the so-called Bonferroni adjustment (more details are given in [12]). Observed frequencies of 1's outside of the limits point to nonrandom behavior.

For each CAM, three graphs were created (not all shown here), first for the frequency of 1's for each A input, then over all B inputs, and finally for all bit positions in C; in terms of $\mathbf{Y}$, each point on the graph is calculated as $\frac{1}{NJ}\sum_{n=1}^{N}\sum_{j=1}^{J} Y_{n,i_0,j}$ for its respective i_0.

Figure 6 shows the frequency of 1 over dimension A for CAM 1 when using the 128-bit key.

Figure 7 provides a summary of the frequency information over all CAMs for A inputs. The graph shows the frequency which has the greatest absolute difference from the expected value of 0.5 within each CAM. Note that the acceptable

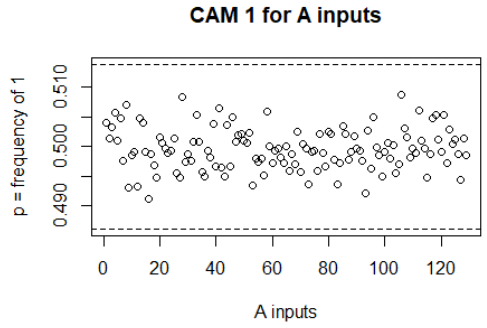

Fig. 6: The frequency of 1 over dimension A for CAM 1 when using the 128-bit key.

bounds have changed due to the Bonferroni adjustment, as described earlier.

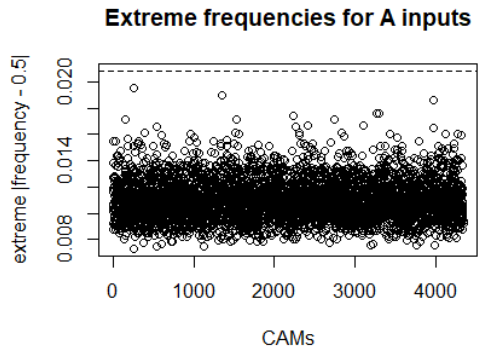

Fig. 7: A summary of the frequency information over all CAMs for A inputs when using the 128-bit key.

In all graphs, we can see results confirming the variability within the acceptable range of values. We conclude that no departures from randomness of the MK-3 scheme were detected based on the frequency tests for the 128-bit key.

B. Serial Tests

The Serial-2 test checks whether all possible values of two-bit blocks 00, 01, 10, and 11 occur in equal proportions in the binary sequence. The blocks do not overlap: the first block consists of the first and second bits of the sequence; the second block consists of the third and fourth bits of the sequence; and so on. This serves the purpose of ensuring independent observations for testing.

We first check each pattern separately in order to see if the frequency follows the expected range around the expected value (of 0.25 for two-bit patterns). The acceptable limits of variability are calculated based on the formula

$$p \pm z(1-\alpha/(2k))\sqrt{(p(1-p)/n)} \tag{5}$$

where p is the expected frequency, $z(x)$ is the percentile (quantile) from the standard normal distribution, n is the sample size, and k is the number of statistical inferences performed.

Figure 8 shows the frequencies of the Serial-2 pattern 00 in tests along the A input dimension for CAM 1 when using the 128-bit key.

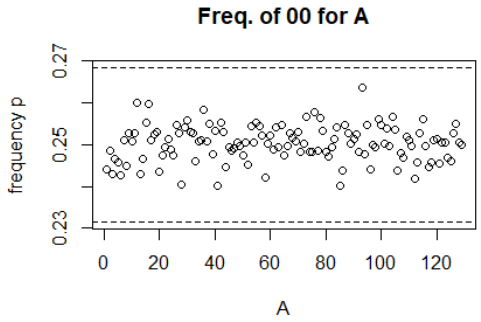

Fig. 8: The frequencies of the Serial-2 pattern 00 in tests along the A input dimension for CAM 1 when using the 128-bit key.

Figure 9 is analogous to Figure 8 but it shows the tests along the B input dimension.

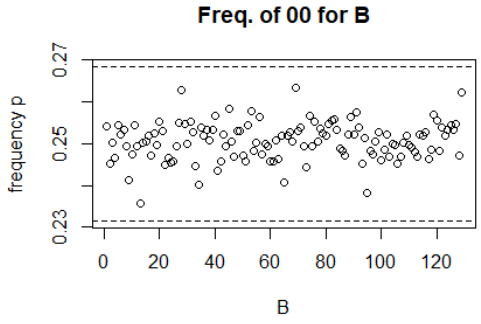

Fig. 9: The frequencies of the Serial-2 pattern 00 in tests along the B input dimension for CAM 1 when using the 128-bit key.

Figure 10 is again analogous to Figure 8 but it shows the tests along the C output dimension.

Figure 11 summarizes the results of the Serial-2 tests with the 128-bit key. This is done by showing the most extreme frequencies for each dimension and bit pattern over all CAMs. We conclude that no departures from randomness of the MK-3 scheme were detected based on the Serial-2 tests for all CAMs when using the 128-bit key.

In all figures, we can see results confirming the variability within the acceptable range of values. We conclude randomness of the field customized MK-3 scheme based on the serial tests when using the 128-bit key.

V. Conclusions

The goal of the security analysis presented in this paper was to establish that the field customized algorithm does not exhibit any statistical anomalies (not expected in a random

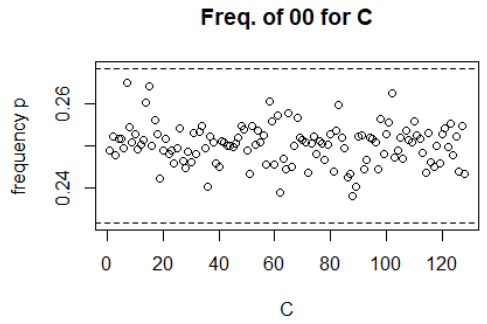

Fig. 10: The frequencies of the Serial-2 pattern 00 in tests along the C output dimension for CAM 1 when using the 128-bit key.

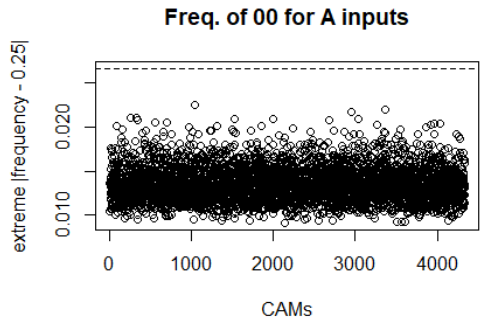

Fig. 11: Serial-2 tests for A inputs of all CAMs when using the 128-bit key.

function); that is, together with previous analysis of the basic and factory customized versions, the algorithm behaves as a random function.

In Section II, we described the design of the MK-3 cipher algorithm at a high level and then its customization modes, their rationale, and their security autonomy. The same security claims are valid for all factory customizations, and for further algorithm field customizations which can be easily adopted for the unique users. The number of possible distinct versions is very large, so that, from a practical point of view, this number can be considered as unbounded.

Overall, three phases of analysis were performed on the MK-3 authenticated encryption algorithm: cryptographic analysis, bit position statistical analysis, and ciphertext statistical analysis. Each phase entailed running an extensive series of tests on the algorithm. The tests were executed in an isolated computing environment. The MK-3 encryption algorithm is immune to a brute force attack; is immune to differential and linear attacks; and is immune to other attacks. These results ensure that an adversary cannot discover the secret key being used to encrypt traffic.

Section III described the statistical analysis methodology used to test the field customized MK-3 cipher. More specifically, we studied randomness of the mapping between the key, plaintext, and ciphertext as defined by the MK-3 scheme using a previously developed array-based statistical method, which is more efficient than checking all outputs as one stream of bits. Within the array-based approach, we use frequency tests and serial tests.

The results of our tests are presented in Section IV. We show that variability in frequency tests stays within the acceptable limits of statistical variability. We illustrate it with graphs for individual CAMs as well as summary graphs for all CAMs. The acceptable bounds were adjusted for the number of statistical inferences being performed, using the so-called Bonferroni adjustment. A similar strategy was used in our Serial Tests, where both the individual CAMs were represented as well as summary graphs for all CAMs were produced.

In conclusion: The MK-3 cipher behaves as random function. This holds true even when various customization modifications are made to the algorithm. We conclude that the MK-3 block cipher's output ciphertext traffic streams look like random data. These results ensure that the MK-3 encryption algorithm effectively conceals its plaintext input.

REFERENCES

[1] Matthew Kelly. Design and Cryptanalysis of a Customizable Authenticated Encryption Algorithm. Master's thesis, Rochester Institute of Technology, 2014. http://scholarworks.rit.edu/theses/8325/.

[2] Matthew Kelly, Alan Kaminsky, Michael Kurdziel, Marcin Łukowiak, and Stanisław Radziszowski. Customizable sponge-based authenticated encryption using 16-bit S-boxes. In *MILCOM 2015 - IEEE Military Communications Conference*, pages 43–48, 2015.

[3] Gordon Werner, Steven Farris, Alan Kaminsky, Michael Kurdziel, Marcin Lukowiak, and Stanisław Radziszowski. Implementing authenticated encryption algorithm MK-3 on FPGA. In *MILCOM 2016 - IEEE Military Communications Conference*, pages 1225–1230, 2016.

[4] Peter Bajorski, Alan Kaminsky, Michael Kurdziel, Marcin Łukowiak, and Stanisław Radziszowski. Customization modes for the Harris MK-3 authenticated encryption algorithm. In *MILCOM 2018 - IEEE Military Communications Conference*, pages 1–5, 2018.

[5] Michael Kurdziel, Matthew Kelly, Alan Kaminsky, Marcin Lukowiak, and Stanisław Radziszowski. Customizable encryption algorithm based on a sponge construction with authenticated and non-authenticated modes of operation, 2016. US Patent 9,438,416 B2.

[6] Peter Bajorski, Alan Kaminsky, Michael Kurdziel, Marcin Łukowiak, and Stanisław Radziszowski. Array-based statistical analysis of the MK-3 authenticated encryption scheme. In *MILCOM 2018 - IEEE Military Communications Conference*, pages 1–9, 2018.

[7] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the sponge: single-pass authenticated encryption and other applications. In *Selected Areas in Cryptography*, pages 320–337. Springer, 2012.

[8] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The KECCAK reference. *NIST SHA-3 Submission Document*, January 2011. http://keccak.noekeon.org/Keccak-reference-3.0.pdf.

[9] Philipp Jovanovic, Atul Luykx, and Bart Mennink. Beyond $2^{c/2}$ Security in Sponge-Based Authenticated Encryption Modes. Cryptology ePrint, Report 373, 2014. http://eprint.iacr.org.

[10] Alan Kaminsky. The coincidence test: a Bayesian statistical test for block ciphers and MACs, September 2013. https://www.cs.rit.edu/~ark/parallelcrypto/cryptostat/coincidence.pdf.

[11] Kevin Hoyt. Structuring statistical tests for validating encryption: An array-based approach. Master's thesis, Rochester Institute of Technology, 2016. http://scholarworks.rit.edu/theses/9073/.

[12] Jay L. Devore. *Probability and Statistics for Engineering and the Sciences*. Cengage, 2012.