Randomness Testing of the NIST Light Weight Cipher Finalist Candidates

Emanuele Bellini[1] and Yun Ju Huang[1]

Cryptography Research Centre, Technology Innovation Institute, Abu Dhabi, UAE.
name.lastname@tii.ae

Abstract. In this report, we show the results of the NIST statistical tests performed on different datasets generated from the output of all possible reduced-round versions of the finalists of the NIST Lightweight standardization process. The objective of the experiment is to provide another metric to compare how conservative or aggressive the choice of number of rounds is for each candidate. Note that a similar analysis was also performed during the Advanced Encryption Standard selection in 1999 and 2000 and later in 2011 to the SHA-3 candidates.

Keywords: NIST Light Weight Cipher · Randomness Test · NIST Statistical Tools

Contents

1 Introduction

In August 2018, NIST has initiated a process to solicit, evaluate, and standardize lightweight cryptographic algorithms that are suitable for use in constrained environments where the performance of current NIST cryptographic standards is not acceptable. The cryptographic algorithms were requested to provide authenticated encryption with associated data (AEAD) functionality, and optionally, the hashing functionality.

Since then, the cryptographic community has contributed to the cryptanalysis and benchmarks on different platforms, both software and hardware, of the initial 57 submissions. The ten finalists were selected on March 29, 2021, namely: ASCON, Elephant, GIFT-COFB, Grain128-AEAD, ISAP, Photon-Beetle, Romulus, Sparkle, TinyJambu, and Xoodyak.

In this report, we show the results of the NIST statistical tests performed on different datasets generated from the output of all possible reduced-round versions of the finalists of the NIST Lightweight standardization process. The objective of the experiment is to provide another metric to compare how conservative or aggressive the choice of number of rounds is for each candidate. Note that a similar analysis was also performed during the Advanced Encryption Standard selection in 1999 and 2000 and later in 2011 to the SHA-3 candidates.

1.1 Related Work

While Knuth's empirical statistical tests [Knu69] were already defined in the late sixties, it was only in the late nineties, that statistical test suites started to become more and more popular to systematically test the cryptographic properties of random number generators and stream ciphers. As an example, see Marsaglia's DIEHARD tests [MT+02], Brown's DIEHARDER tests [Bro21], or NIST Statistical Test Suite [BRS+10]. In this work, rather than for testing random number generators, we consider a different number of tests to identify statistical biases in the output bits of a full or reduced round block cipher or cryptographic permutation, without any knowledge of the internal structure of the cipher (*black box scenario*).

One of the first works we are aware of tackling this problem is by Gustafson et al., in 1997 [GDG97]. In this work, the authors present essentially three ways of generating a dataset related to the block cipher input and output of every round, and for each dataset they measure the deviation from an expected distribution using different metrics.

Between 1999 and 2000, NIST released the analysis of the Advanced Encryption Standard candidate algorithms with respect to some statistical properties (including the ones in [GDG97]) that could be measured from different type of output generated by each candidate [Sot99, BS00]. The statistical properties were defined in the so called NIST Statistical Test Suite. At the time, this test suite was on preparation. The test suite was finalized in 2001 [RSN+01] and then finally superseeded by [BRS+10] in 2010. In the context of AES standardization, the purpose of this tests was to demonstrate the suitability of candidate algorithms as random number generators. The 1999 analysis included 15 encryption algorithms, and required to generate more than 135 data sets (9 data sets for each algorithm), for a total of almost 29 billion bits (about 3.6 Gigabytes), only for testing the 128-bit key version of each encryption algorithm. In the 2000 analysis, only the 192 and 256-bit key versions of the 5 finalists (Mars, RC6, Rijndael, Serpent and Twofish) were analyzed, with respect to basically the same set of tests[1]. Note that all statistical tests were performed both for full and partial rounds (the test required several months on several SUN Ultra workstations), but, due to resource constraints, partial round testing was limited to only one of the datasets (the low-density plaintext dataset).

A similar analysis considering different datasets was performed in [TDT05] and extended in [Sul11] (2011), where the tests were applied to both the AES and SHA-3 candidates. Additionally, in [Sul11], a set of additional tests was performed, namely: the Strict Avalanche Criterion Test, the Linear Span Test, the Collision Test, and the Coverage Test.

1.2 Contribution

In this work, we re-propose a similar analysis as the one performed by NIST for the AES standardization process [Sot99, BS00]. Even if the primitives of the NIST Lightweight standardization process are not meant to be used as random number generators, we still believe this analysis to be of interest, especially as our results can be considered as a metric to compare how conservative is the choice of the number of rounds in each candidate.

1.3 Organization

The remainder of this paper is structured as follows. In Section 2, we describe the type of datasets and the statistical tests used to analyze the datasets. In Section 3, we report the experimental results of the statistical tests. In Section 4, we draw the conclusion according to the experimental results.

2 Randomness Testing Experimental Setup

The statistical test activity can be divided into two phases: the *dataset generation* and the *statistical tests*. For both phases, we followed the approach taken by NIST during the analysis of the AES candidates [Sot99, BS00]. The target of the analysis are the underlying primitives of the candidates, not the Authenticated Encryption nor the Hash constructions as a whole. This means we considered the underlying permutations or block ciphers.

2.1 Dataset generation

During this phase, we generated the following datasets:

1. Avalanche Plaintext
2. Avalanche Key
3. Plaintext-Ciphertext correlation
4. Cipher Block Chaining Mode
5. Random
6. Low-Density with Plaintext
7. Low-Density with Key
8. High-Density with Plaintext
9. High-Density with Key

[1] The data sets were reduced from 9 to 8 (removing the Random Plaintext/Random 128-Bit Keys dataset). The statistical tests contained one extra test, the Serial Test, with respect to 1999. Precisely, 16 core statistical tests that, under different parameter inputs, could be viewed as 189 statistical tests.

Table 1: Breakdown of the 188 statistical tests applied during experimentation.

Statistical Test	No. of P-values	Test ID	Statistical Test	No. of P-values	Test ID
Monobit	1	1	Periodic Template	1	157
Block Frequency	1	2	Universal Statistical	1	158
Cusum	2	3-4	Approximate Entropy	1	159
Runs	1	5	Random Excursions	8	160-167
Long Runs of Ones	1	6	Random Excursions Variant	18	168-185
Rank	1	7	Serial	2	186-187
Spectral DFT	1	8	Linear Complexity	1	188
Aperiodic Templates	148	9-156			

Please refer to [Sot99, BS00] for the detailed description of the datasets.

The datasets containing the keyword "key" could only be generated for the block ciphers. We generated each of the above dataset for every round of the primitive.

We remark that the analysis could have been stopped at a certain round as soon as randomness was determined at this specific round.

2.2 Statistical tests

The NIST Statistical Test Suite [BRS+10] was used to perform the statistical tests. This suite consists of 15 core statistical tests that, under different parameter inputs, can be viewed as 188 statistical tests. Lempel-Ziv Compression test, stated in [BRS+10] is not implemented here. In Table 1, we list each of the core statistical tests, followed by the number of P-values reported by each core test and the test identifier.

Each P-value corresponds to the application of an individual statistical test on a single binary sequence. For a brief description of each of these tests, see Appendix A of [Sot99]. Full details documenting the derivation and description of the tests may be found in the NIST publication [BRS+10].

3 Experimental Results

For the plaintext/key avalanche dataset, the total data that has been generated is 175,865,217,024 bits (i.e. 175 Giga Bytes). The approximate time to generate this dataset is 3 hours. For the plaintext/ciphertext correlation dataset, the total data that has been generated is 57,532,395,520 bits (i.e. 57 Giga Bytes). The approximate time to generate all this dataset is 1 hour. For the CBC mode dataset, due to its chaining behavior, cannot be parallelized and hence the data generation is still in progress. Currently, the total data that has been generated, excluded Spongent-π[160], Spongent-π[176] and skinny-128-384+, is 69,291,600,000 bits (i.e. 69 Giga Bytes). The approximate time to generate all this dataset is 2 weeks. For the random dataset, has the same data size with the plaintext/ciphertext correlation one, and it takes 1.5 hours to finish the task. For the plaintext/key low density dataset, the total data that has been generated is 50,866,290,688 bits (i.e. 50 Giga Bytes). The approximate time to generate all this dataset is 5 minutes. The plaintext/key high density dataset has the same size and similar time spent. Currently, we generated total around 462 Giga Bytes datasets for the test.

All the parameters follow the parameters used in [Sot99] and the adapted change will be stated in the next sub-sections.

The dataset generation has been performed using the NumPy library and an independent non-optimized python implementation of each cipher.

The total time to execute all tests was approximately two weeks (excluded the CBC Mode datasets).

All experiments were executed in the following machine:

- Server 1 and 2: 16 Intel(R) Xeon(R) Gold 5222 CPUs, 4-cores, 3.80GHz, 252G RAM
- Server 3: 112 Intel(R) Xeon(R) Platinum 8280 CPUs, 28-cores, 2.70GHz, 1152G RAM

We report a summary of the results of the tests in Table 2. In the table, when we say that an underlying primitive comes to random at round r we means the tests it failed (of 188 tests) at round r are no more than 4.

Currently, the datasets of Sparkle-family and Grain-128 are missing. Also, the CBC datasets of Spongent-π[160], Spongent-π[176] and skinny-128-384+ are still in progress. We will add the results in the final version of this report.

3.1 Plaintext and Key Avalanche

Unlike the AES candidates, the underlying primitives are different in their block size. In order to make the test results to be comparable, we fixed the length of the input sequences for the NIST statistical test in the level of 10*6 bits. In the processing of the plaintext/key avalanche datasets, in the case with block size 128, we follow the settings in [Sot99]. For those primitives which has larger block size, we use a suitable number of avalanche samples in one sequence which made the sequence length of each the underlying primitives are closer. Here, one avalanche sample means the total derived blocks with one fixed input. For example, one avalanche sample of ASCON contains 300 blocks, and the size of the sample is 300*300 bits, that is 90,000 bits. Table 3 shows the parameters of each cipher for the plaintext/key avalanche datasets generation. Total 384 sequences are generated for both plaintext and key avalanche dataset.

As shown in Table 2, we see that most of the schemes becomes random after a few rounds compared to the total round suggested, however TinyJambu P uses more rounds to be random. In TinyJambu-192 and TinyJambu-256, the suggestion rounds of P_{640} are not random at the final round.

For more details of passing tests in certain rounds nearby the round comes to random, please refer to Appendix A.

3.2 Plaintext/Ciphertext Correlation

Following the design of the plaintext/key avalanche dataset, for the underlying primitives with block size 128 bits, we use the parameters shown in [Sot99]. For other primitives with larger block size, we uses less blocks in one sequence. Please refer to Table 4 for detailed parameters. Total 128 sequences are generated for the randomness test.

As shown in Table 2, most of the underlying primitives with SPN-based structure show good randomness at this test. However, for others types, especially TinyJambu P and Gift-128, the inputs and outputs are highly correlated at the first few rounds.

Since the primitives with SPN-based structure are random at the first beginning, we are not going to show the detail of the behavior of randomness here. For more details of other primitives, the passing tests in certain rounds nearby the round comes to random are shown in Appendix A.

3.3 CBC Mode

Due to the nature of The CBC Mode, the CBC Mode datasets cannot be parallelized and require more computing time for data generation. At this moment, we currently miss the results of Spongent-π[160], Spongent-π[176] and skinny-128-384+. The CBC results as shown in Table 2 is quite similar to the plaintext/ciphertext correlation dataset. That is, they all came to the randomness at the same round. The dataset parameters are also similar to the parameters of plaintext/ciphertext correlation dataset, except that total 300 sequences are tested.

Nist LW cipher	Underlying Primitives	Block Size	Key Size	Passed NIST Statistical Tests (Randomness) at Round | Total Rounds								
				Avalanche		Plaintext/Ciphertext Correlation	CBC	Random	Low Density		High Density	
				Plaintext	Key				Plaintext	Key	Plaintext	Key
SPN-based Permutation												
Ascon	Ascon's Permutation	320	-	4|12	-	1|12	1|12	1|12	-	-	-	-
Elephant	Dumbo: Elephant-Spongent-π[160]	160	-	8|80	-	1|80	-	1|80	-	-	-	-
	Jumbo: Elephant-Spongent-π[176]	176	-	8|90	-	1|90	-	1|90	-	-	-	-
	Delirium: Elephant-Keccak-f[200]	200	-	3|18	-	1|18	1|18	1|18	-	-	-	-
ISAP	Ascon's Permutation	320	-	4|12	-	1|12	1|12	1|12	-	-	-	-
	Keccak-p[400,16]	400	-	3|16	-	1|16	1|16	1|16	-	-	-	-
	Keccak-p[400,20]	400	-	3|20	-	1|20	1|20	1|20	-	-	-	-
PHOTON-Beetle	PHOTON$_{256}$	256	-	3|12	-	1|12	1|12	1|12	-	-	-	-
Xoodyak	Xoodoo	384	-	4|12	-	1|12	1|12	1|12	-	-	-	-
Keyed Permutation												
TinyJambu	TinyJambu-128 P_{640}	128	128	17|20	19|20	4|20	4|20	1|20	14|20	17|20	14|20	17|20
	TinyJambu-128 P_{1024}	128	128	17|32	19|32	4|32	4|32	1|32	14|32	17|32	14|32	16|32
	TinyJambu-192 P_{640}	128	192	17|20	–|20	4|20	4|20	1|20	14|20	17|20	14|20	17|20
	TinyJambu-192 P_{1152}	128	192	17|36	21|36	4|36	4|36	1|36	14|36	17|36	14|36	17|36
	TinyJambu-256 P_{640}	128	256	17|20	–|20	4|20	4|20	1|20	15|20	19|20	14|20	20|20
	TinyJambu-256 P_{1280}	128	256	17|40	23|40	4|40	4|40	1|40	15|40	19|40	14|40	20|40
SPN-based Block Cipher												
GIFT-COFB	GIFT-128	128	128	8|40	10|40	2|40	2|40	1|40	7|40	9|40	7|40	8|40
Tweakable Block Cipher												
Romulus	skinny-128-384+	128	384	7|40	8|40	1|40	-	1|40	6|40	8|40	6|40	8|40

Table 2: Comparison of Randomness

Table 3: Parameters for Plaintext/Key Avalanche Dataset

Nist LW cipher	Underlying Primitives	Block Size	Key Size	Sequnces	Plaintext			Key		
					Sample Size	Samples per Seq	Bits per Seq	Sample Size	Samples per Seq	Bits per Seq
SPN-based Permutation										
Ascon	Ascon's Permutation	320	-	384	102400	11	1126400	-	-	-
Elephant	Dumbo: Elephant-Spongent-π[160]	160	-	384	25600	41	1049600	-	-	-
	Jumbo: Elephant-Spongent-π[176]	176	-	384	30976	34	1053184	-	-	-
	Delirium: Elephant-Keccak-f[200]	200	-	384	40000	27	1080000	-	-	-
ISAP	Ascon's Permutation	320	-	384	102400	11	1126400	-	-	-
	Keccak-p[400,16]	400	-	384	160000	7	1120000	-	-	-
	Keccak-p[400,20]	400	-	384	160000	7	1120000	-	-	-
PHOTON-Beetle	PHOTON$_{256}$	256	-	384	65536	16	1048576	-	-	-
Xoodyak	Xoodoo	384	-	384	147456	8	1179648	-	-	-
Keyed Permutation										
TinyJambu	TinyJambu-128 P_{640}	128	128	384	16384	64	1048576	16384	64	1048576
	TinyJambu-128 P_{1024}	128	128	384	16384	64	1048576	16384	64	1048576
	TinyJambu-192 P_{640}	128	192	384	16384	64	1048576	24576	43	1056768
	TinyJambu-192 P_{1152}	128	192	384	16384	64	1048576	24576	43	1056768
	TinyJambu-256 P_{640}	128	256	384	16384	64	1048576	32768	32	1048576
	TinyJambu-256 P_{1280}	128	256	384	16384	64	1048576	32768	32	1048576
SPN-based Block Cipher										
GIFT-COFB	GIFT-128	128	128	384	16384	64	1048576	16384	64	1048576
Tweakable Block Cipher										
Romulus	skinny-128-384+	128	384	384	16384	64	1048576	49152	22	1081344

Similar to the plaintext/ciphertext correlation dataset, the primitives with SPN-based structure are random at the first beginning, therefore, we are not going to show the detail of the behavior of randomness here. For more details of other primitives, the passing tests in certain rounds nearby the round comes to random are shown in Appendix A.

We will put the missing results of other underlying primitives in the final version of this paper.

3.4 Random

The random dataset setup shares the same parameters with the plaintext/ciphertext correlation dataset, refers to Table 4. As shown in Table 2, all the underlying primitives are going to random at the first round.

3.5 Plaintext and Key Low Density

The low density dataset generation requires a key, hence, for the underlying primitives belongs to the class of SPN-based permutation, the low/high density test is not applicable.

For other type of underlying primitives, in the case that its block/key size is 128 bits, we use the parameters of [Sot99], that is, 8257 blocks with all possible low density inputs. For lager key size, we discard some weight 2 low density sequence (two 1s in the sequence) and make the blocks in one sequence still be 8257, to fit the proper sequence size. Total 128 sequences is generated for the test.

As shown in Table 2, the skinny-128-384 scheme goes to random faster (in proportion to total round) than other primitives. And also, from the testing results, we can see that fixed key with different inputs is more secured than fixed inputs with different keys.

For more details of the passing tests in different rounds, please refer to Appendix A.

3.6 Plaintext and Key High Density

The high density datasets shares the same parameter with low density datasets in data generation, that is, 8257 blocks in each sequence and total 128 sequences. For the larger key size, we also discard some weight 2 high density sequence (two 0s in the sequence).

Table 4: Parameters for Correlation and Random Dataset

NIST LW cipher	Underlying Primitives	Block Size	Key Size	Sequence	Blocks per Seq	Bits per Seq
SPN-based Permutation						
Ascon	Ascon's Permutation	320	-	128	3252	1040640
Elephant	Dumbo: Elephant-Spongent-π[160]	160	-	128	6503	1040480
	Jumbo: Elephant-Spongent-π[176]	176	-	128	5912	1040512
	Delirium: Elephant-Keccak-f[200]	200	-	128	5202	1040400
ISAP	Ascon's Permutation	320	-	128	3252	1040640
	Keccak-p[400,16]	400		128	2601	1040400
	Keccak-p[400,20]	400	-	128	2601	1040400
PHOTON-Beetle	PHOTON$_{256}$	256	-	128	4064	1040384
Xoodyak	Xoodoo	384	-	128	2710	1040640
Keyed Permutation						
TinyJambu	TinyJambu-128 P_{640}	128	128	128	8128	1040384
	TinyJambu-128 P_{1024}	128	128	128	8128	1040384
	TinyJambu-192 P_{640}	128	192	128	8128	1040384
	TinyJambu-192 P_{1152}	128	192	128	8128	1040384
	TinyJambu-256 P_{640}	128	256	128	8128	1040384
	TinyJambu-256 P_{1280}	128	256	128	8128	1040384
SPN-based Block Cipher						
GIFT-COFB	GIFT-128	128	128	128	8128	1040384
Tweakable Block Cipher						
Romulus	skinny-128-384+	128	384	128	8128	1040384

For each primitives, the high/low density datasets have the same behavior. There is ±1 difference in the round comes to random. This is because at the margin of randomness, the number of tests failed are quite near.

For more details of the passing tests in different rounds, please refer to Appendix A.

4 Conclusion

According to Table 2, we can see that most of the underlying primitives produce datasets which seem random in the first third of the total number of rounds. For the Spongent-*pi* this proportion is much higher, which seems to indicate a very conservative choice in the number of rounds of this cipher. On the other hand, we can see that the choice of the number of rounds for TinyJambu seems more aggressive.

Still, some experiments are missing to draw a final conclusion. We will finalize all experiments for the final version of this report.

References

[Bro21] Robert G. Brown. Dieharder: A Random Number Test Suite Version 3.31.1, 2021. Available at https://webhome.phy.duke.edu/~rgb/General/dieharder.php.

[BRS$^+$10] Lawrence Bassham, Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Stefan Leigh, M Levenson, M Vangel, Nathanael Heckert, and D Banks. Special

Publication (NIST SP) - 800-22 Rev 1a: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, September 2010.

[BS00] Lawrence Bassham and Juan Soto. NISTIR 6483: Randomness testing of the advanced encryption standard finalist candidates. *NIST Internal or Interagency Reports*, 2000.

[GDG97] HM Gustafson, EP Dawson, and J Dj Golić. Automated statistical methods for measuring the strength of block ciphers. *Statistics and Computing*, 7(2):125–135, 1997.

[Knu69] Donald Knuth. The art of computer programming, vol. 2: Seminumerical algorithms, 1969.

[MT+02] George Marsaglia, Wai Wan Tsang, et al. Some difficult-to-pass tests of randomness. *Journal of Statistical Software*, 7(3):1–9, 2002.

[RSN+01] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, M Levenson, M Vangel, D Banks, Nathanael Heckert, James Dray, and S Vo. Special Publication (NIST SP) - 800-22: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, May 2001.

[Sot99] Juan Soto. NISTIR 6390: Randomness testing of the advanced encryption standard candidate algorithms. *NIST Internal or Interagency Reports*, 1999.

[Sul11] Fatih Sulak. *Statistical analysis of block ciphers and hash functions*. PhD thesis, Graduate School of Applied Mathematics of Middle East Technical University, February 2011. Available at `https://open.metu.edu.tr/bitstream/handle/11511/20626/index.pdf?sequence=1`.

[TDT05] Deniz Toz, Ali Doğanaksoy, and Meltem Sönmez Turun. Statistical analysis of block ciphers. *Ulusal Kriptologi Sempozyumu, Ankara, Turkey*, pages 56–66, 2005.

A NIST Statistical Test Results of Underlying Primitives

A.1 ASCON permutation

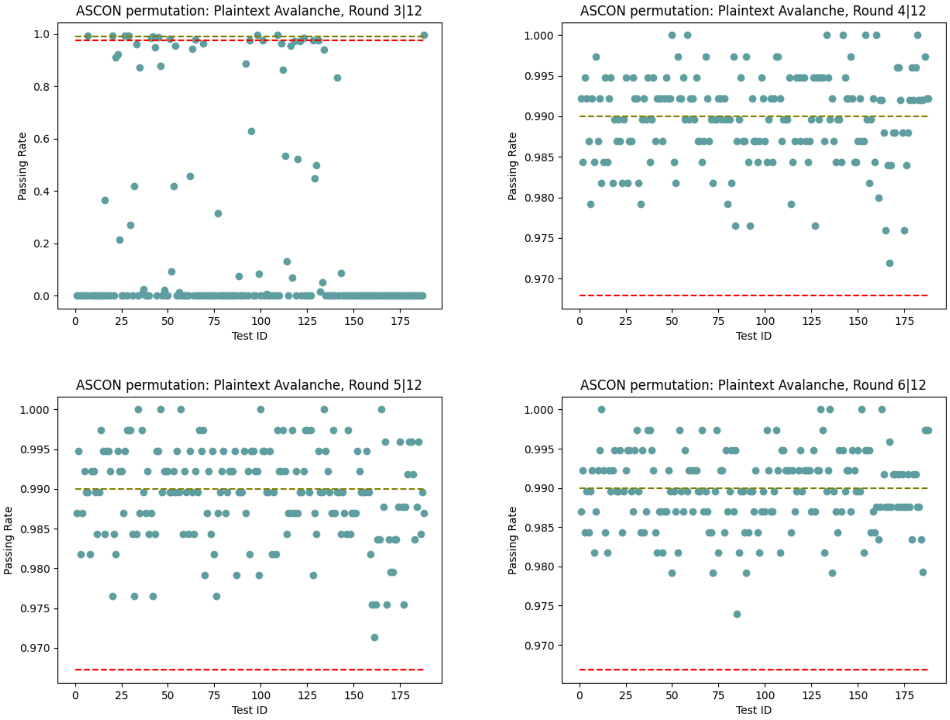


Figure 1: Test for ASCON permutation with avalanche datasets from round 3 t0 6.

A.2 Spongent-π[160]

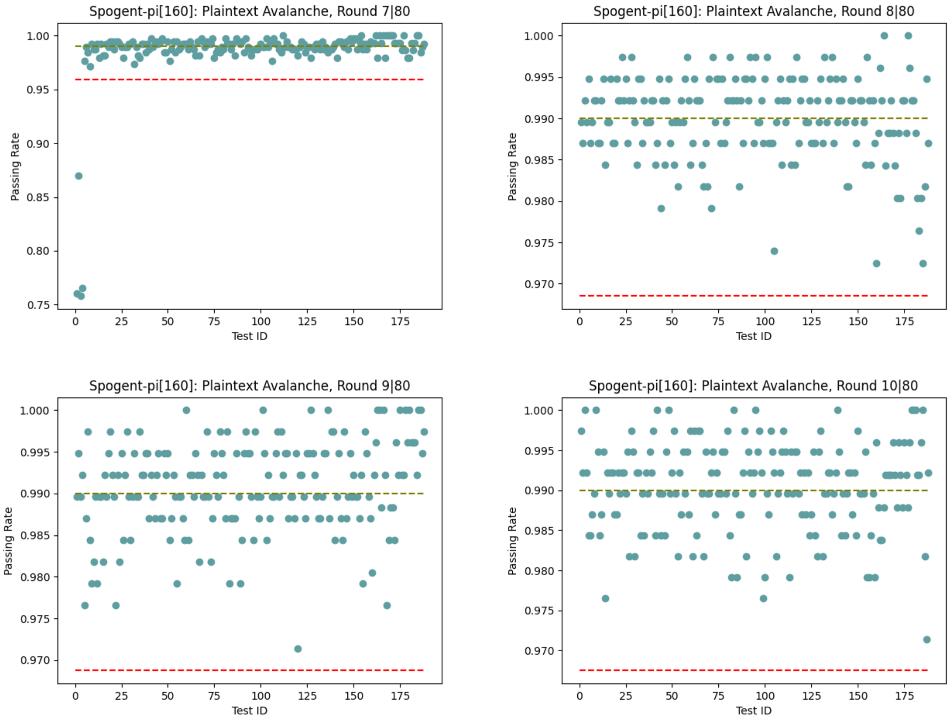


Figure 2: Test for Spongent-π[160] with avalanche datasets from round 7 to 10.

A.3 Spongent-π[176]

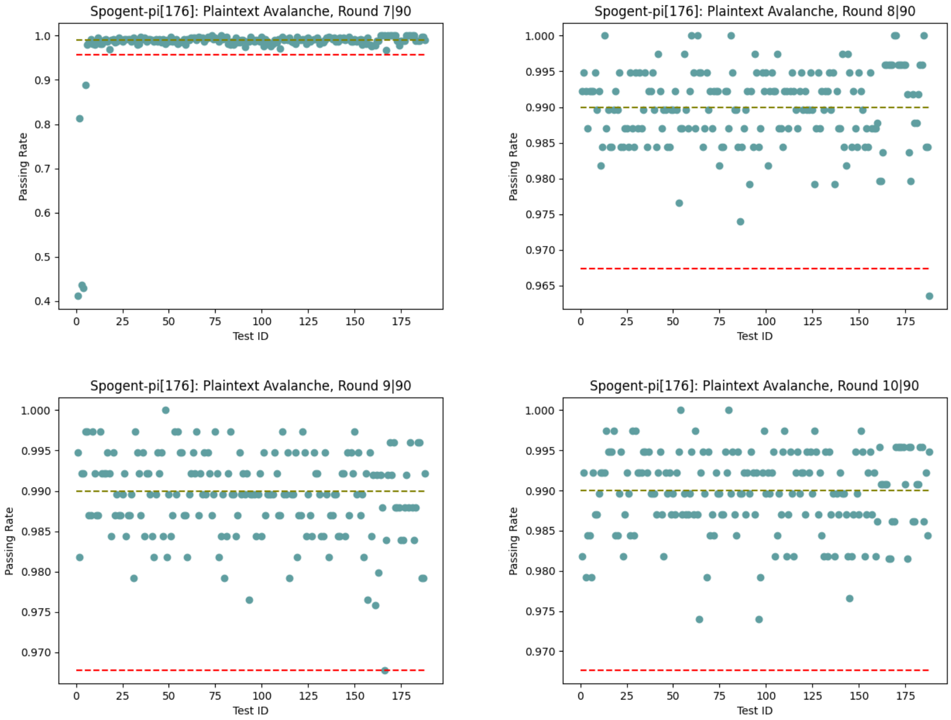


Figure 3: Test for Spongent-π[176] with avalanche datasets from round 7 to 10.

A.4 Keccak-f[200]

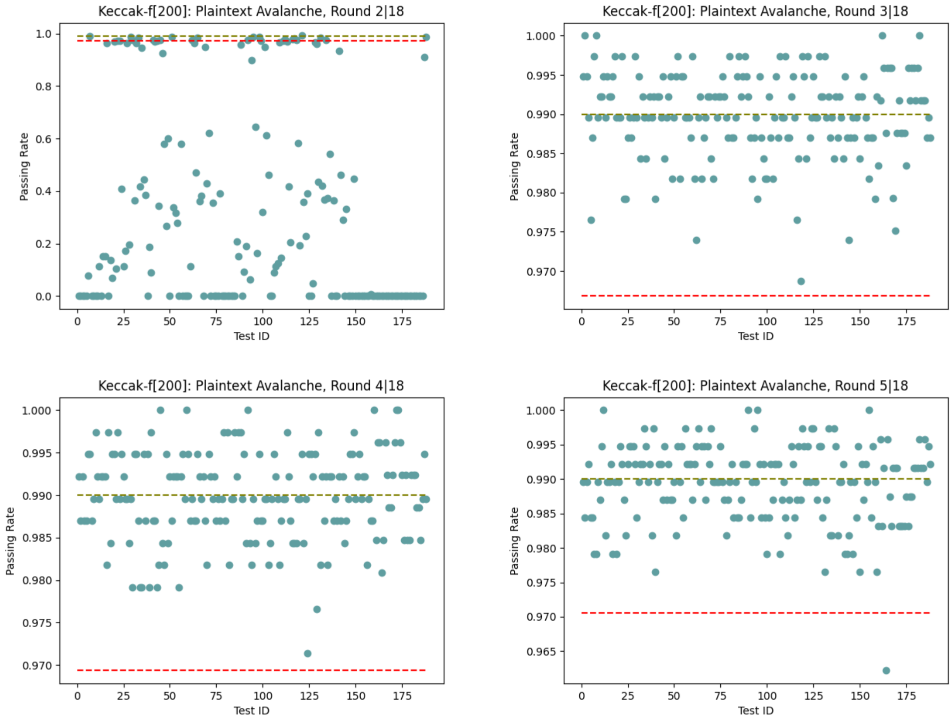


Figure 4: Test for Keccak-f[200] with avalanche datasets from round 2 to 5.

A.5 Keccak-p[400]

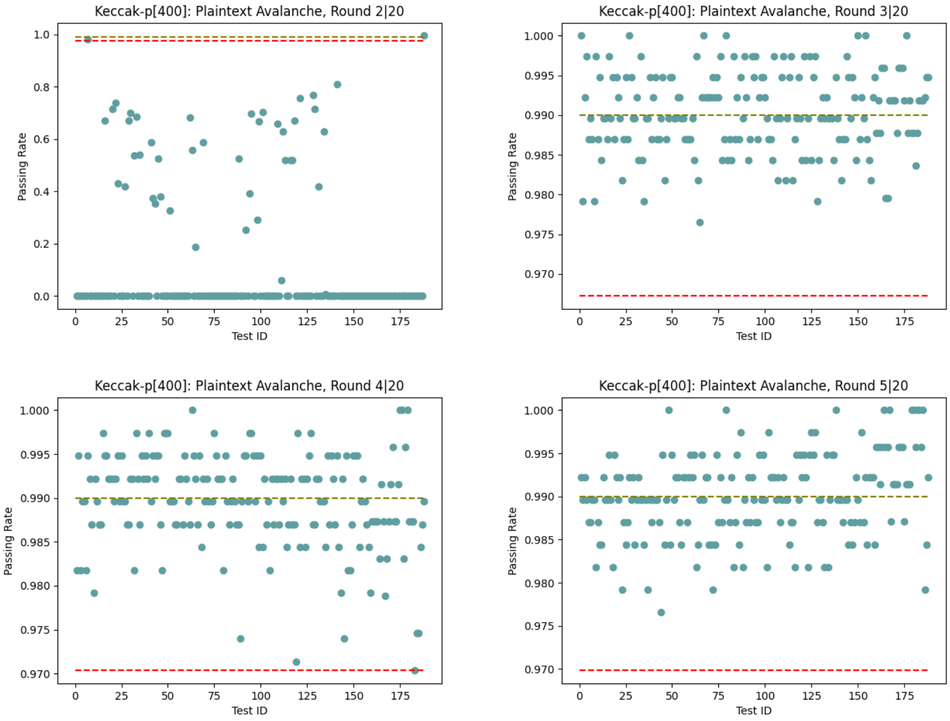
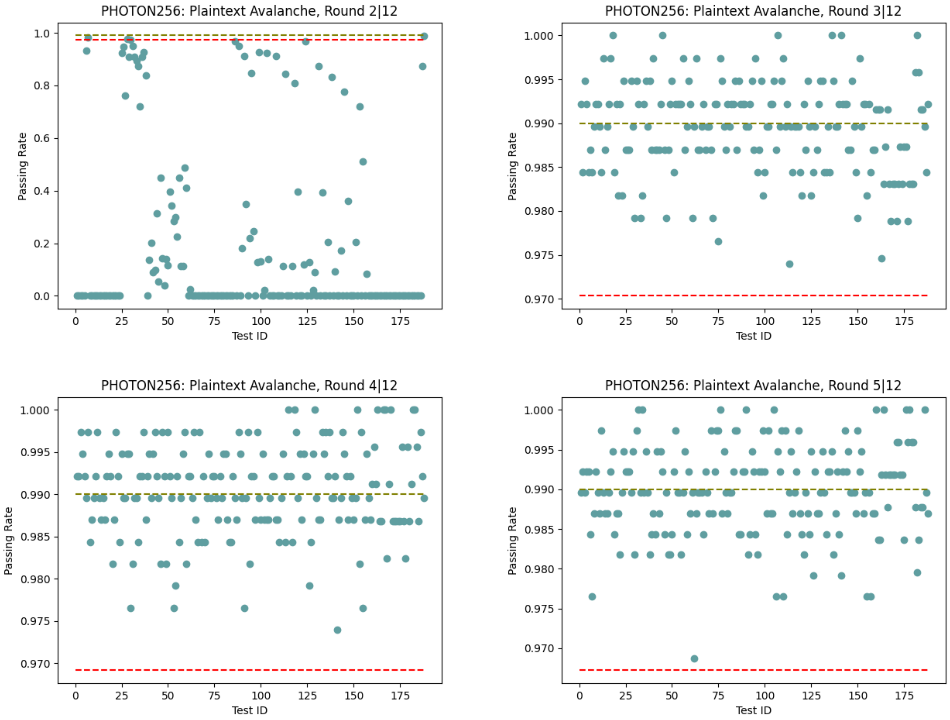


Figure 5: Test for Keccak-p[400] with avalanche datasets from round 2 to 5.

A.6 PHOTON$_{256}$


PHOTON256: Plaintext Avalanche, Round 2|12

PHOTON256: Plaintext Avalanche, Round 3|12

PHOTON256: Plaintext Avalanche, Round 4|12

PHOTON256: Plaintext Avalanche, Round 5|12

Figure 6: Test for PHOTON$_{256}$ with avalanche datasets from round 2 to 5.

A.7 Xoodoo

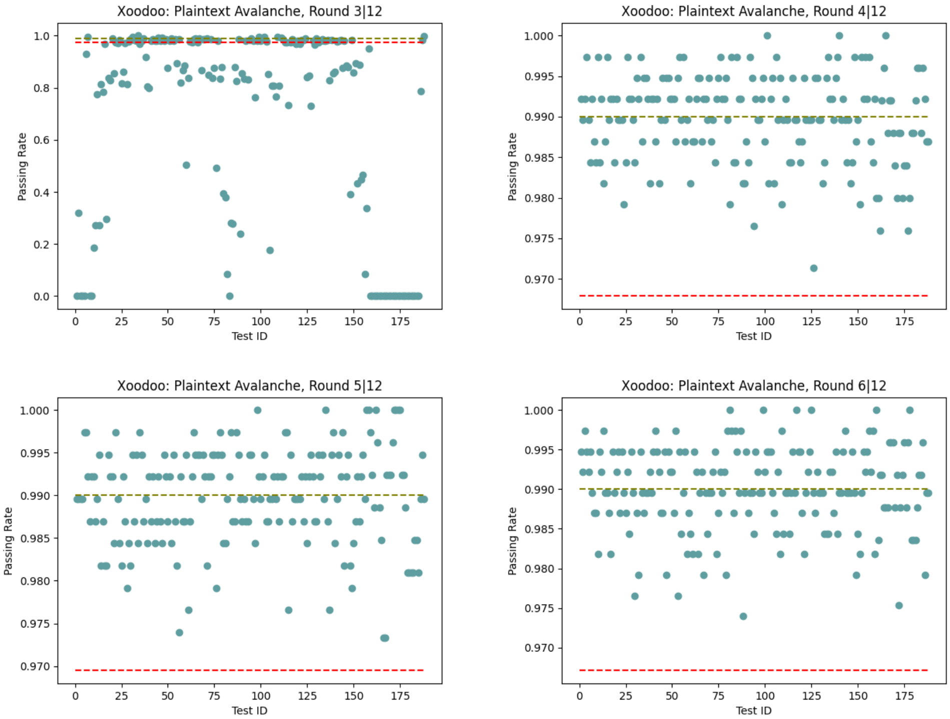


Figure 7: Test for Xoodoo with avalanche datasets from round 3 to 6.

A.8 TinyJambu-128 P

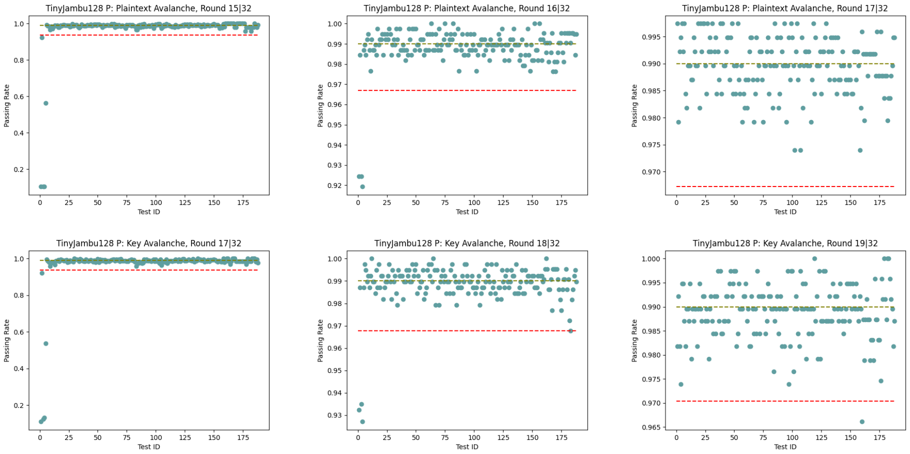


Figure 8: Test for TinyJambu-128 P with plaintext/key avalanche datasets from round 15 to 17 and 17 to 19.

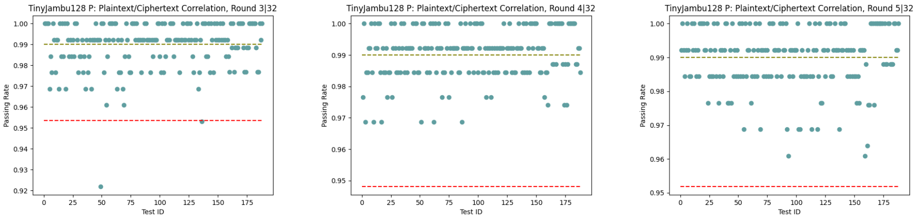


Figure 9: Test for TinyJambu-128 P with plaintext/ciphertext correlation from round 3 to 5.

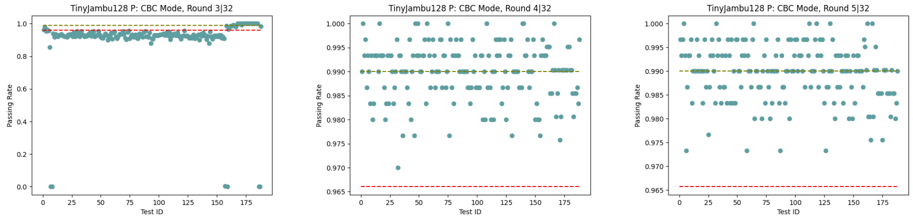


Figure 10: Test for TinyJambu-128 P with CBC mode from round 3 to 5.

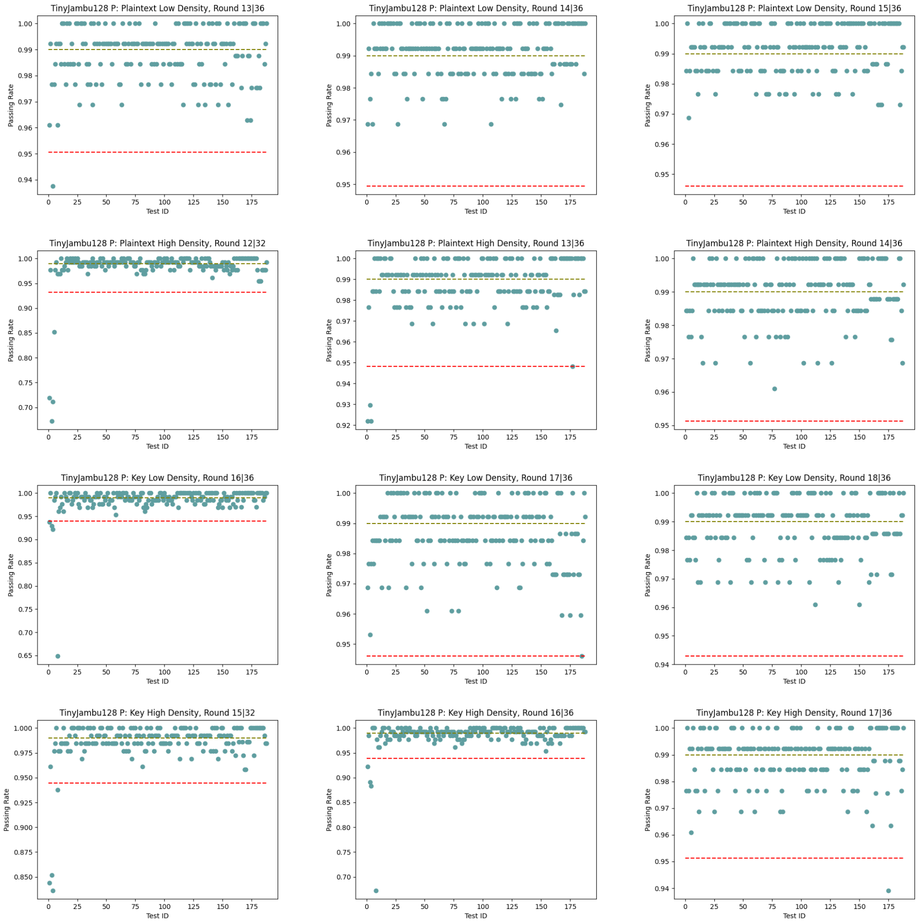
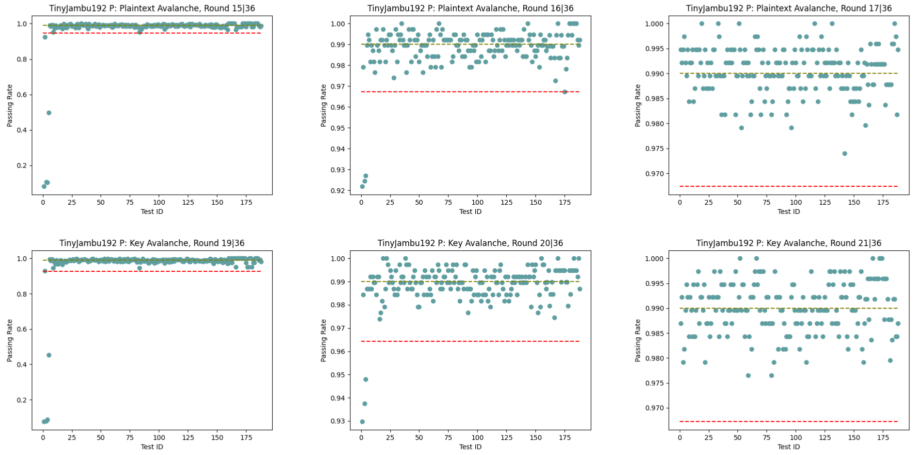


Figure 11: Test for TinyJambu-128 P with plaintext/key low/high density datasets.

A.9 TinyJambu-192 P


TinyJambu192 P: Plaintext Avalanche, Round 15|36

TinyJambu192 P: Plaintext Avalanche, Round 16|36

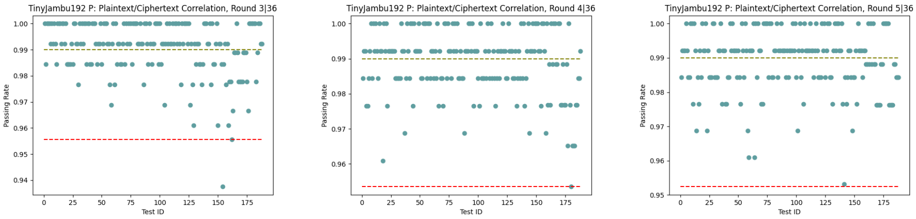
TinyJambu192 P: Plaintext Avalanche, Round 17|36

TinyJambu192 P: Key Avalanche, Round 19|36

TinyJambu192 P: Key Avalanche, Round 20|36

TinyJambu192 P: Key Avalanche, Round 21|36

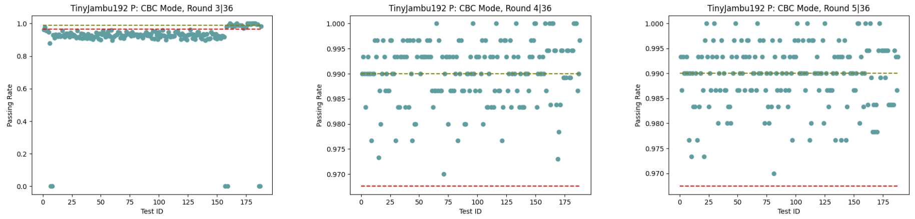
Figure 12: Test for TinyJambu-192 P with plaintext/key avalanche datasets from round 15 to 17 and 19 to 21.


TinyJambu192 P: Plaintext/Ciphertext Correlation, Round 3|36

TinyJambu192 P: Plaintext/Ciphertext Correlation, Round 4|36

TinyJambu192 P: Plaintext/Ciphertext Correlation, Round 5|36

Figure 13: Test for TinyJambu-192 P with plaintext/ciphertext correlation from round 3 to 5.


TinyJambu192 P: CBC Mode, Round 3|36

TinyJambu192 P: CBC Mode, Round 4|36

TinyJambu192 P: CBC Mode, Round 5|36

Figure 14: Test for TinyJambu-192 P with CBC mode from round 3 to 5.

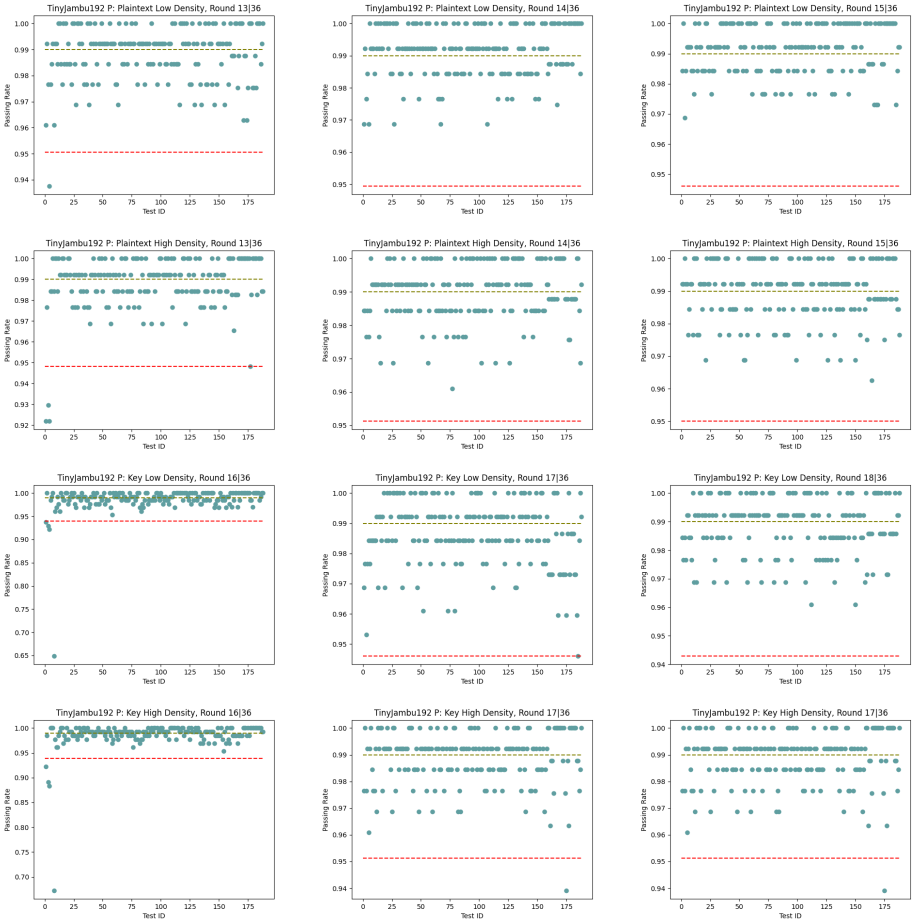


Figure 15: Test for TinyJambu-192 P with plaintext/key low/high density datasets.

A.10 TinyJambu-256 P

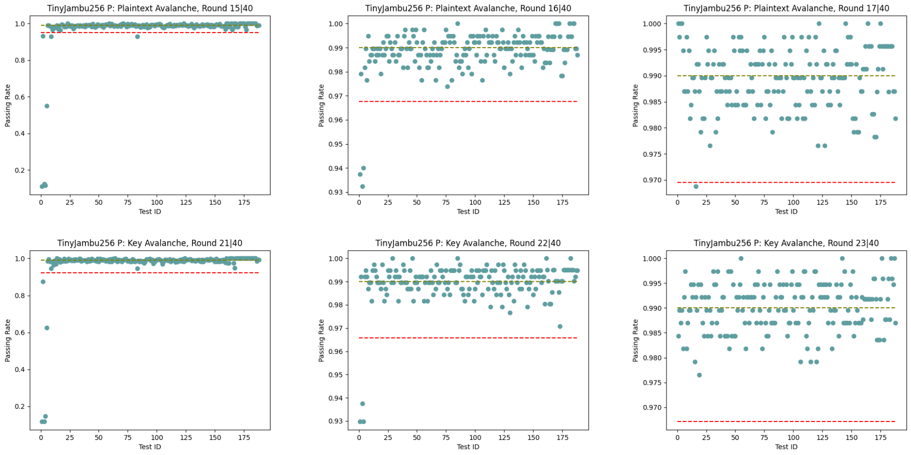


Figure 16: Test for TinyJambu-256 P with plaintext/key avalanche datasets from round 15 to 17 and 21 to 23.

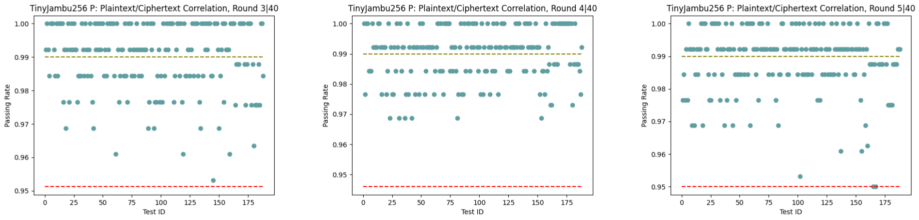


Figure 17: Test for TinyJambu-256 P with plaintext/ciphertext correlation from round 3 to 5.

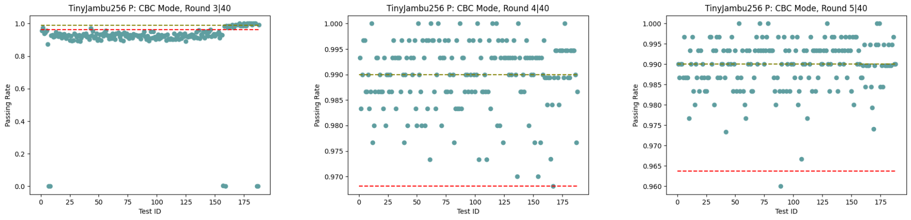


Figure 18: Test for TinyJambu-256 P with CBC mode from round 3 to 5.

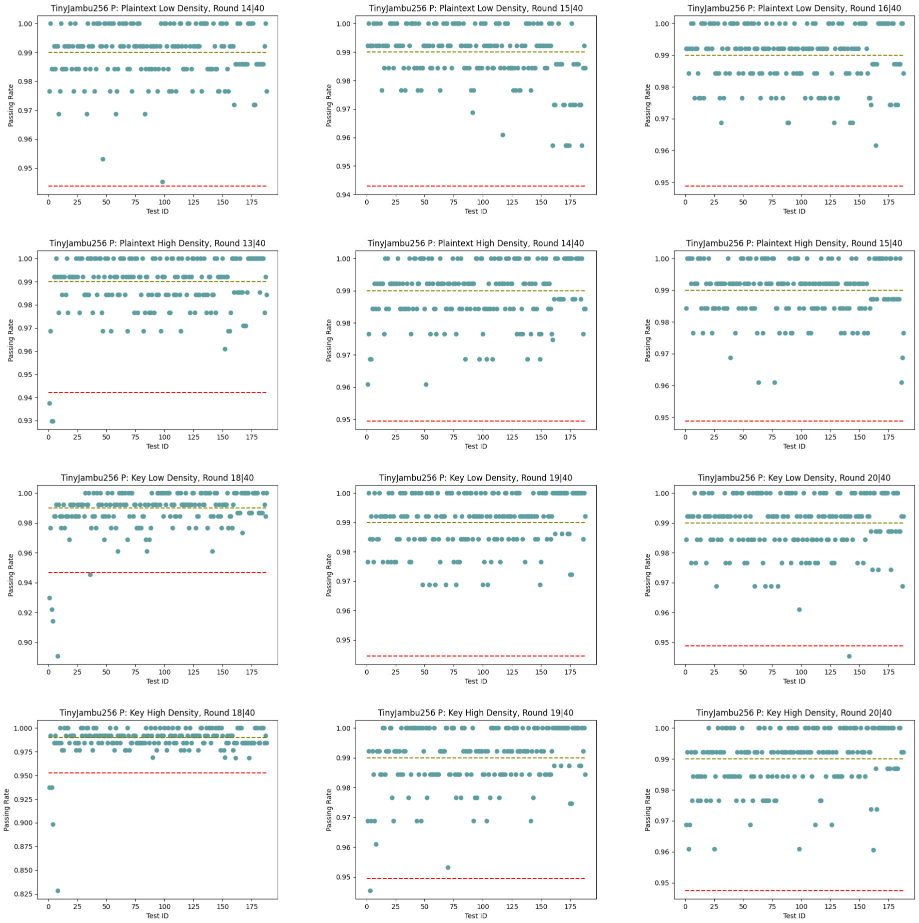


Figure 19: Test for TinyJambu-256 P with plaintext/key low/high density datasets.

A.11 GIFT-128

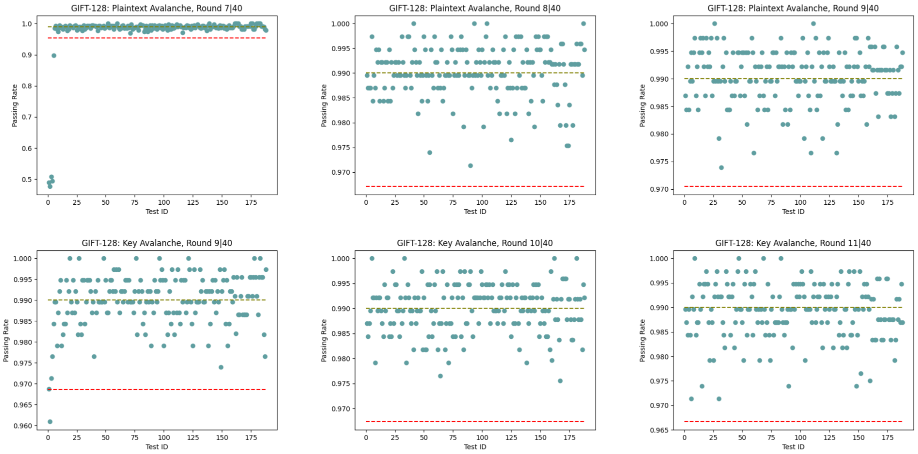


Figure 20: Test for GIFT-128 with plaintext/key avalanche datasets from round 7 to 9 and 9 to 11.

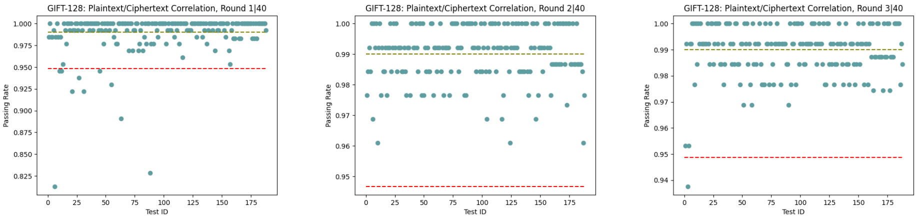


Figure 21: Test for GIFT-128 with plaintext/ciphertext correlation from round 1 to 3.

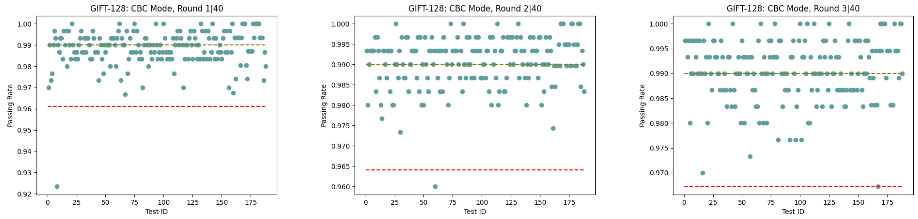


Figure 22: Test for GIFT-128 with CBC mode from round 1 to 3.

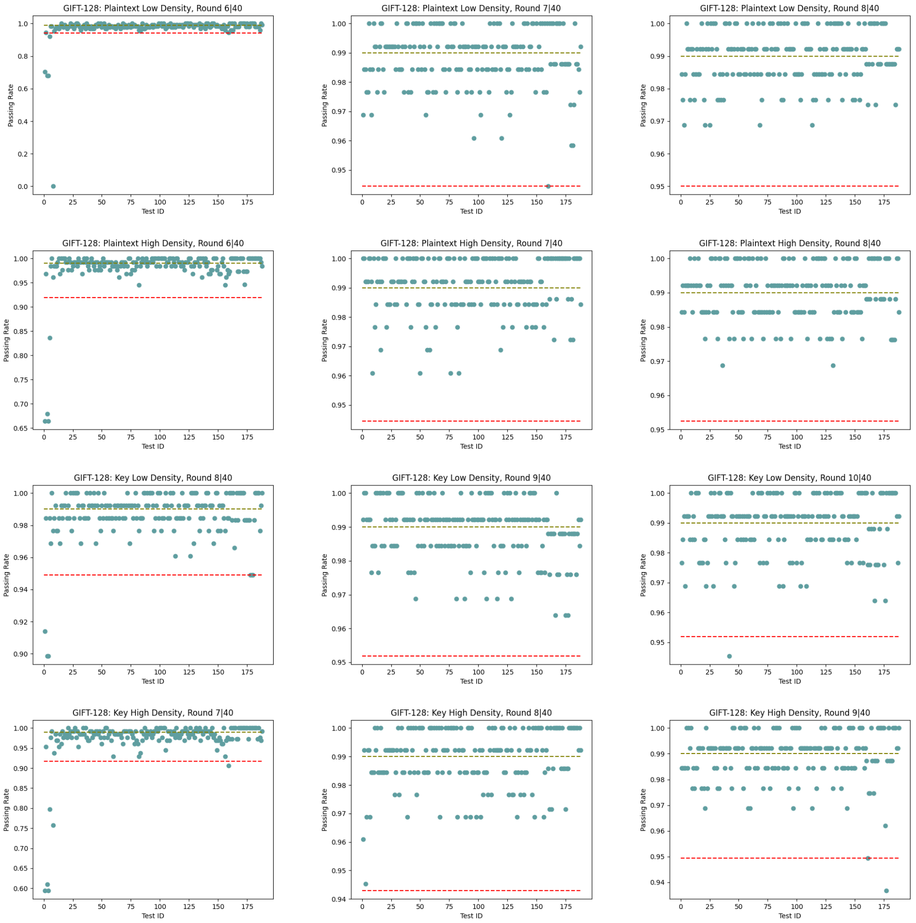


Figure 23: Test for GIFT-128 with plaintext/key low/high density datasets.

A.12 skinny-128-384+

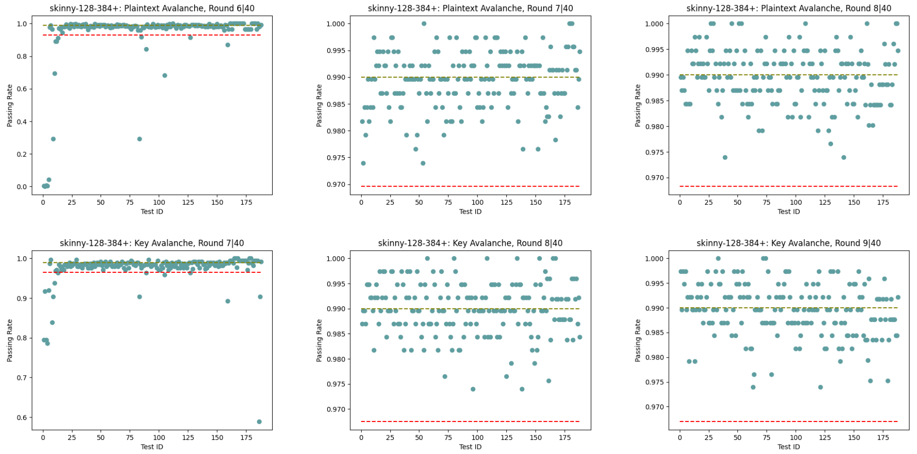


Figure 24: Test for skinny-128-384+ with plaintext/key avalanche datasets from round 6 to 8 and 7 to 9.

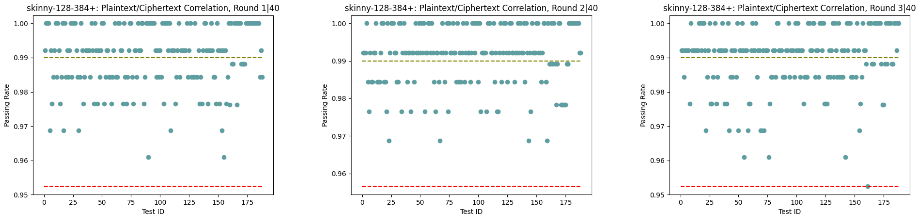


Figure 25: Test for skinny-128-384+ with plaintext/ciphertext correlation from round 1 to 3.

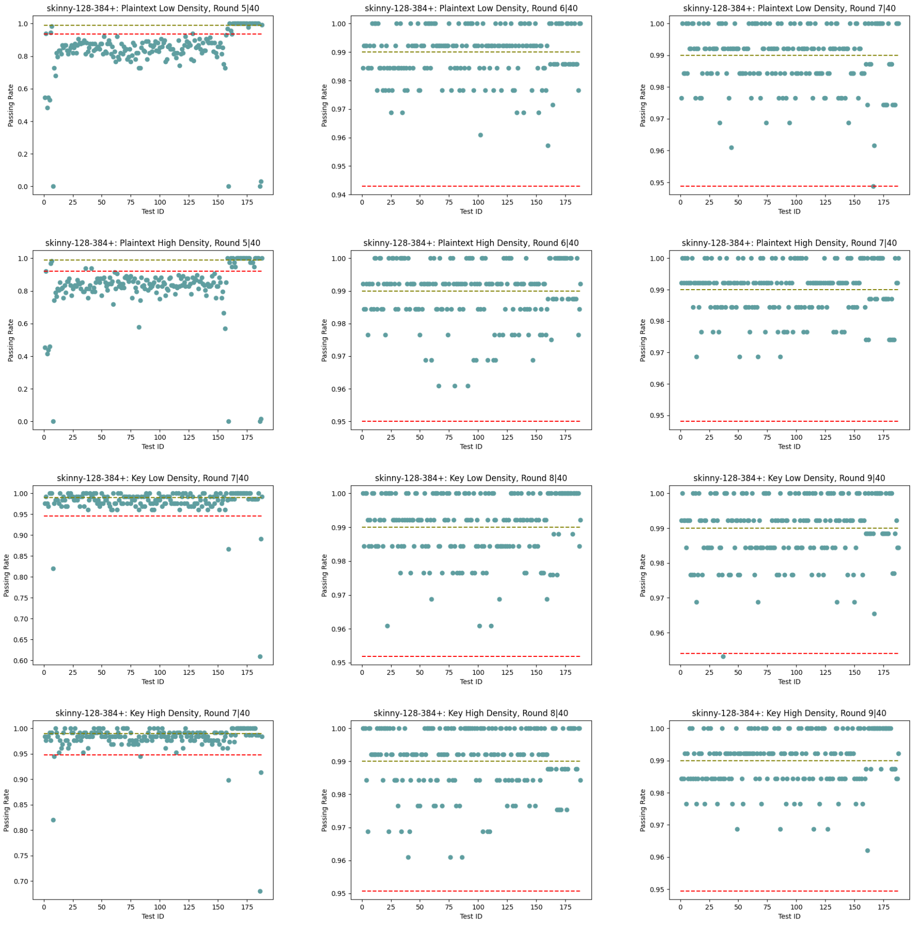


Figure 26: Test for skinny-128-384+ with plaintext/key low/high density datasets.