QEEP Analysis: Security, Performance, and Power Consumption

QEEP Benchmark Testing Report

18 December 2020

quantropi.com

QEEP Analysis: Security, Performance, and Power Consumption

QEEP Benchmark Testing Report

Yu Rang Kuang
Co-founder and Chief Scientific Officer
Quantropi Inc.
Ottawa, ON, Canada

Eslam G. AbdAllah
Postdoctoral Fellow
Carleton University
Ottawa, ON, Canada

ABSTRACT

The power of quantum computers depends on Heisenberg uncertainty principle and the superposition capability of quantum systems. This quantum power has significant impact on breaking current public key cryptographic techniques. Quantropi's vision is that information system is a simple quantum system and can be protected by uncertainty principle using existing infrastructure. Quantropi proposes a novel approach governed by uncertainty principle for repeatable perfect secure quantum communications by using quantum permutation gates. This approach can be applied in quantum and classical information systems. Quantropi's approach provides the repeatable perfect secrecy at endpoints, regardless of the communication medium. In this report, we present a sample implementation of Quantropi's approach using permutation matrices. Quantropi's implementation is named **Q**uantum **E**ntropy **E**ncoding **P**rotection (QEEP).

In this report, we analyse QEEP from a security, performance, and power consumption perspective. We compare QEEP with the current industry standard cryptographic algorithms, Advanced Encryption Standard (AES) and Advanced Encryption Standard New Instructions (AES-NI), which is a hardware accelerated version of AES. Through our experiments, we measure QEEP output randomness, as well as encoding and decoding speed, and power consumption. Our experiments are carried out on different processor platforms and with different encryption workloads. Our experimental results show that QEEP achieves up to 18x speed over AES, and up to 2.4x speed over AES-NI at 40% (AES-NI) to 95% (AES) reduced energy consumption over AES and AES-NI. The low energy footprint makes QEEP an ideal candidate for quantum secure communication for IoT and other lightweight edge devices.

1 INTRODUCTION

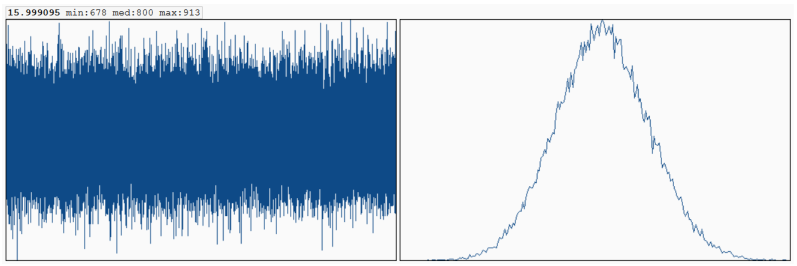
Quantropi developed **Q**uantum **E**ntropy **E**ncoding **P**rotection (QEEP) mechanism, which is a classical implementation (QEEP) for its novel approach in which quantum gates can be represented by permutation matrices. In QEEP, the transmitter selects a random permutation matrix based on pre-shared key with the receiver. QEEP consists of three main steps: key generation, encoding, and decoding. QEEP accepts a seed as a user input and generates a state matrix. Each row in the matrix consists of a permutation of all numbers between 0 to 255 (one of the total available 256! sets). QEEP uses the same algorithm for both encoding and decoding. Due to the enormous key space and the security it provides, QEEP uses computationally efficient permutation operators for encoding and decoding.

Quantropi develops a novel solution (QEEP) that achieves the following explicit design goals:

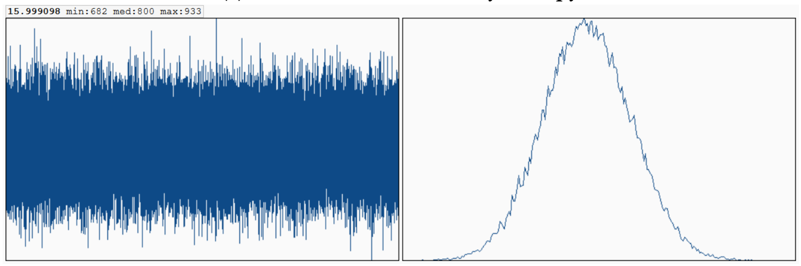
- Quantum attacks resistant. QEEP would be highly effective at preventing attacks from quantum computers that have exponential speed-up over classical computers.
- Lightweight. QEEP is designed with minimal code footprint size, in order to be applied in different applications including limited resources IoT devices.
- Low latency. QEEP is supposed to be very fast compared with current existing techniques, in order to be applied in critical environments such as autonomous vehicles that requires Ultra-reliable low latency communication (URLLC).
- Energy saving. QEEP is designed to consume much less power than existing techniques.

1.1 QEEP Characteristics

QEEP efficiently encode/decode large amount of data with the following characteristics.
Software Implementation. We measure QEEP execution speed, performance across a variety of platforms, and variation of speed with key size. QEEP achieves superior results over AES and AES-NI in all platforms. QEEP can encode/decode around 2 GB/S in some platforms.
Restricted-Space Environments. QEEP is perfect for applications, such as smart cards, limited IoT devices, where relatively small amounts of RAM and/or ROM are available for such purposes as code storage. QEEP requires 2.46 KB for code footprint.
Minimal power usage. QEEP requires minimal power to achieve these high-speed encoding and decoding process for large data sizes. QEEP requires just 1.4 J/GB in some platforms.
Randomness. QEEP generates a random ciphertext that is indistinguishable from true random number generators. The distribution of the ciphertext follows a Gaussian distribution. QEEP uses a strong and lightweight random number generator and provides an additional source of randomness when used with regular plain text sources such as natural language text. QEEP also uses multi-dimensional permutation matrix to increase the randomness of the generated ciphertext. This matrix is efficiently used to encode large amounts of data.
Encoding vs. Decoding. QEEP uses the same algorithm for both encoding and decoding.
Key Agility. QEEP has the ability to change keys quickly and with a minimum of resources. This includes subkey generation when initial seed is provided. QEEP accepts any key size with no limitation to the size.
Other Versatility and Flexibility. QEEP has parameter flexibility includes ease of support for different key and data sizes and

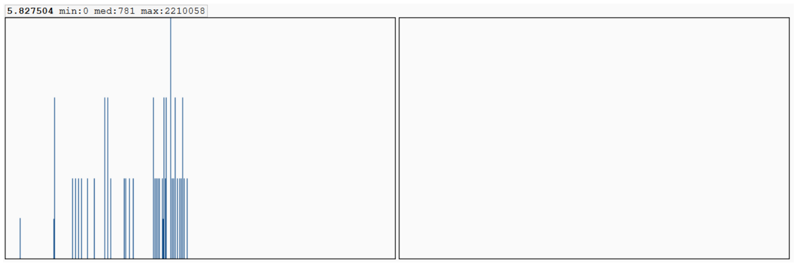


(a) True random number key entropy

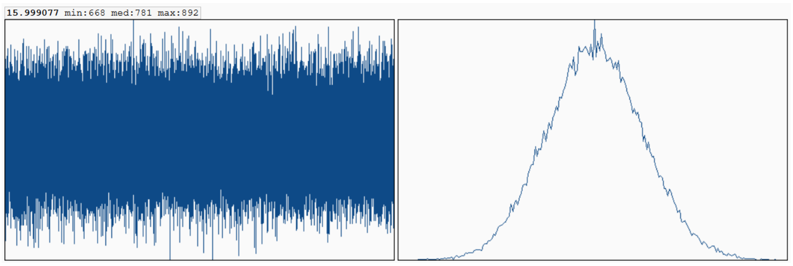


(b) QEEP entropy

Figure 1: Entropy and randomness for unbiased input (true random number key)



(a) Plaintext entropy



(b) QEEP entropy

Figure 2: Entropy and randomness for biased input (regular plaintext)

ease of increasing the number of permutation matrices in order to cope with newly discovered attacks. QEEP also has implementation flexibility to optimize cipher elements for environments.

1.2 QEEP Applications

QEEP can be applied in different applications such as the following examples.

Key distribution. QEEP can be applied for secure key distributions (QEEP-KD) by taking the security advantage of the uncertainty principle. QEEP-KD can be deployed over existing network infrastructure, for point-to-point, app-to-app, device-to-device for a highly secure and cost-effective solution. QEEP can be integrated with TLS 1.3 for low latency and secure communications. One of the major improvements of TLS 1.3 is its 0-RTT (zero round trip time) with the pre-shared key (PSK) but concerns from experts are pretty clear due to possible re-play attacks. QEEP key distribution

can help to remove the possibility of replay attacks and makes TLS 1.3 to be quantum ready.
IoT. It is estimated that over 70% of IoT devices lack adequate security. QEEP can provide lightweight security solutions for devices which lack the resources needed to run traditional cryptographic schemes.
Web security. QEEP can be integrated with a web application and web framework to add quantum security to today's web platforms. Its lightweight and low-latency features will allow the existing user experience to be maintained.

In the following sections, we present QEEP security, performance, and power consumption experiments and results. Performance and power consumption results are carried out on different processor platforms and with different encryption workloads.

2 SECURITY ANALYSIS

In this section, we present our security experiments and results. We test QEEP vs. AES-256-CBC and AES-NI-256-CBC. We measure the randomness based on entropy and randomness online tester, Dieharder and NIST statistical test suite.

Our results show that QEEP and AES-256-CBC pass all tests and have similar security level. QEEP achieves that with minimal footprint size, much faster, and using much less power than AES.
Entropy and randomness test. This experiment makes use of 16 bits Shannon entropy calculator to test serial correlation of binary files [1]. It uses gnuplot to create the frequency and distribution graphs useful for testing normality. Gnuplot is a command-line program that can generate two- and three-dimensional plots of functions, data, and data fits. The results are used to estimate the strength and quality of random number generators.

As depicted in Figure 1 and Figure 2, the graphs show that QEEP output has no bias for certain data values (left) and the output cipher is random and follow a Gaussian normal distribution (right). For key distribution, a transmitter uses a quantum random number generator (QRNG) that generates true random keys based on quantum principle, in electrical signal, i.e. classical bits. Then the transmitter passes the generated keys to QEEP. QEEP output entropy and randomness is almost the same as the input entropy and randomness, as shown in Figure 1. The true random input in this test is generated by ANU quantum random numbers [2] with size 100MB.

In case the data to be transmitted is not randomly generated but biased regular plaintexts. QEEP provides a solution to pre-randomize plaintexts by using the initial seed, as well as random initial vector (IV). By generating PRNG output, the plaintext XOR'ed with PRNG is provided as inputs to QEEP. Figure 2 shows QEEP entropy and randomness when used with a biased plaintext of size 100MB.
Dieharder. The main goal of dieharder is to test random number generators for research and cryptography purposes [3]. The tool is built entirely on top of the Gnu Scientific Library's (GSL) random number generator interface and uses a variety of other GSL tools (e.g. sort, erfc, incomplete gamma, distribution generators) in its operation. The full suite consists of 114 tests. It has three outputs for each test: pass, weak, and fail.

From the test results shown in Table 1, QEEP and AES have similar results and can be considered as perfect random number generators or encryption algorithms. The weak test do not describe an attack opportunity, but are a side-effect of the random test process, even the baseline random number generators given in the dieharder tool also have some weak or fail tests. The test is performed using the same input. The input size is 100MB.

Table 1: Dieharder and NIST results for QEEP and AES

Test	Technique	Pass	Weak	Fail
Dieharder (114 tests)	QEEP	113	1	0
	AES	113	1	0
NIST (15 tests)	QEEP	15	0	0
	AES	15	0	0

NIST suite. A statistical test suite for random and pseudorandom number generators for cryptographic applications [4]. This test suite is widely used in the evaluation and validation of industry standard cryptographic algorithms before they are considered safe for adoption. It consists of fifteen tests: frequency test, test for frequency within a block, runs test, test for the longest run of ones in a block, random binary matrix rank test, discrete fourier transform (spectral) test, non-overlapping (aperiodic) template matching test, overlapping (periodic) template matching test, maurer's universal statistical test, linear complexity test, serial test, approximate entropy test, cumulative sum (cusum) test, random excursions test, and random excursions variant test.

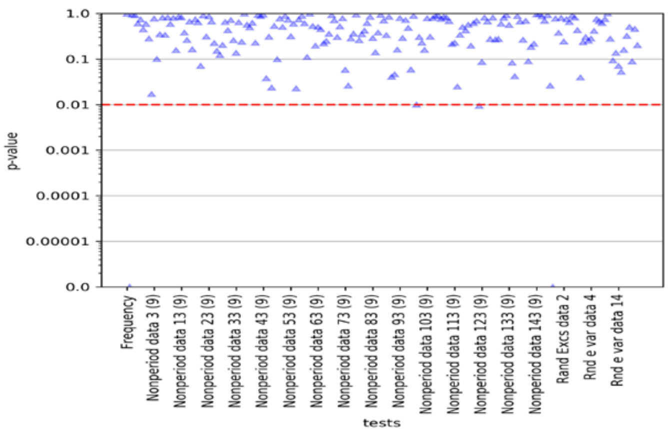
The results presented in Table 1 show that QEEP and AES-256-CBC pass all tests. Figure 3 shows the output p-values for each subtest in the fifteen tests. QEEP has better distribution and only two p-values are on the border line, while AES has two points that are below the border line as shown in Figure 4. Figure 5 shows the generated histogram for QEEP and AES. QEEP maximum value is 25 while AES reaches 30, which shows the QEEP p-values distribution is better than AES. The test is performed using the same input. The input size is 100MB.

Overall, QEEP has equivalent cryptographic characteristics as the current industry standard algorithm, AES, however providing significantly increased key space entropy and thus security under quantum compute attacks.

3 PERFORMANCE ANALYSIS

In the performance related experiments, we measure QEEP speed in different platforms and with various data sizes: 16 bytes, 64 bytes, 256 bytes, 1 KB, 8 KB, and 16 KB as shown in Table 2 to Table 6 and Figure 6 to Figure 10. We compare between QEEP, AES-256-CBC, and AES-NI-256-CBC. We measure AES and AES-NI speed using openssl tool [5]. OpenSSL is a robust, commercial-grade, and full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. It is also a general-purpose cryptography library. We test the speed on the following platforms:

- Intel(R) Pentium Silver J5005 CPU@1.5GHZ OS: Linux v18.10 (Table 2 and Figure 6)
- Intel(R) Core(TM) i5-8250U CPU@1.6GHZ OS: Linux v19.04 (Table 3 and Figure 7)


(a) QEEP p-values

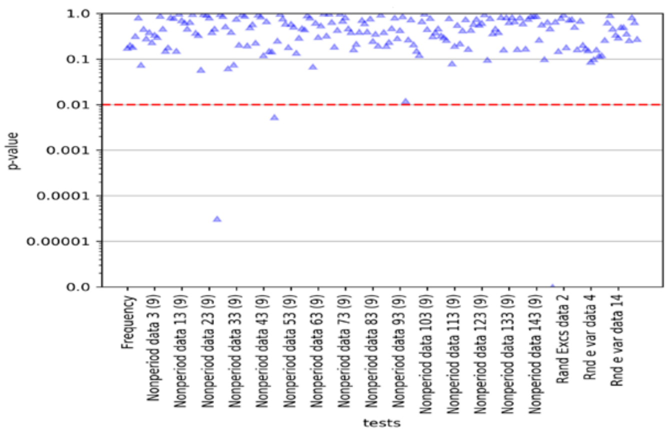
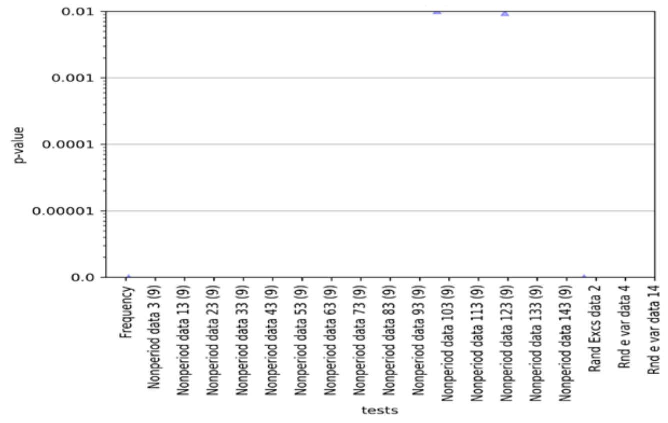

(b) AES p-values

Figure 3: NIST statistical test suite: p-values chart


(a) QEEP zoomed p-values

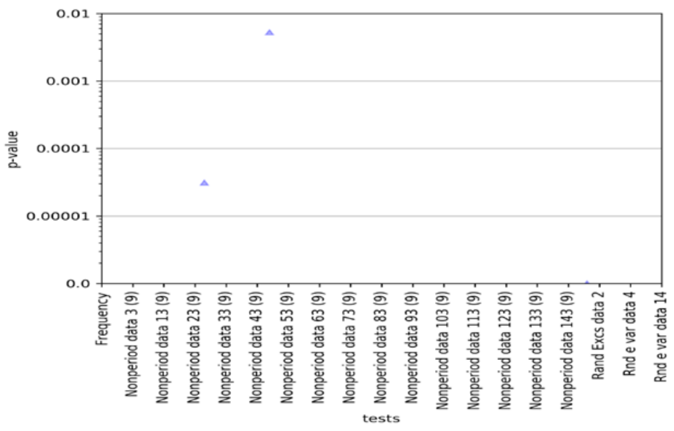
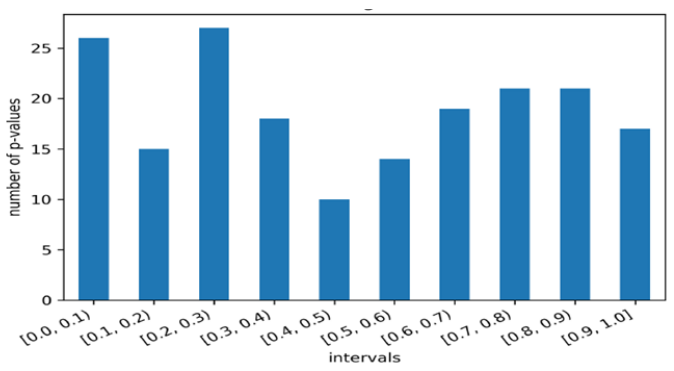

(b) AES zoomed p-values

Figure 4: NIST statistical test suite: zoomed p-values chart


(a) QEEP histogram

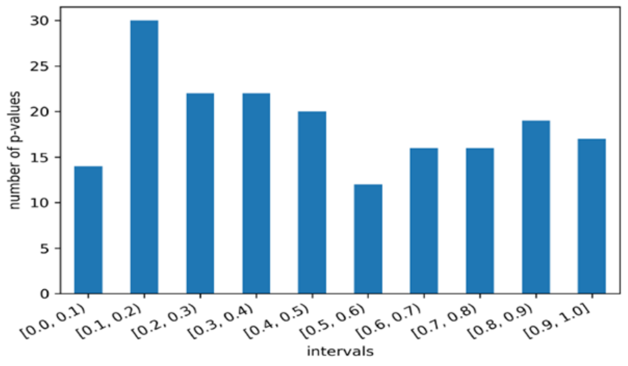
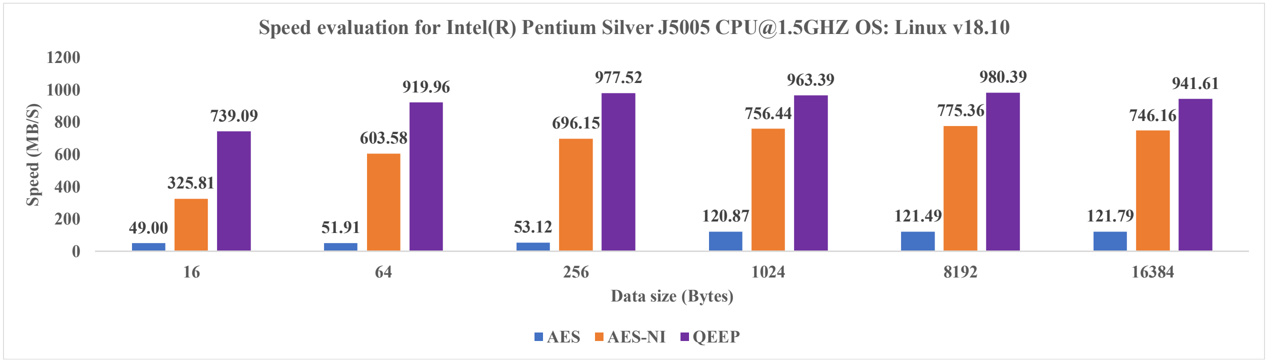

(b) AES histogram

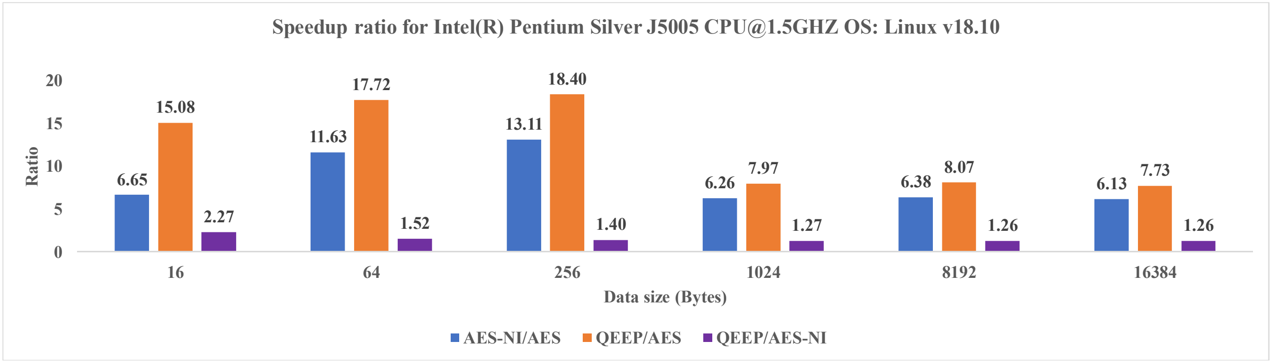
Figure 5: NIST statistical test suite: histogram for all tests

Table 2: Performance Comparison with different data sizes
Platform: Intel(R) Pentium Silver J5005 CPU@1.5GHZ OS: Linux v18.10

		16 bytes	64 bytes	265 bytes	1024 bytes	8192 bytes	16384 bytes
Speed (MB/S)	AES	49.00	51.91	53.12	120.87	121.49	121.79
	AES-NI	325.81	603.58	696.15	756.44	775.36	746.16
	QEEP	739.09	919.96	977.52	963.39	980.39	941.61
Ratio	AES-NI/AES	6.65	11.63	13.11	6.26	6.38	6.13
	QEEP/AES	15.08	17.72	18.40	7.97	8.07	7.73
	QEEP/AES-NI	2.27	1.52	1.40	1.27	1.26	1.26



(a) Speed evaluation



(b) Speedup ratio

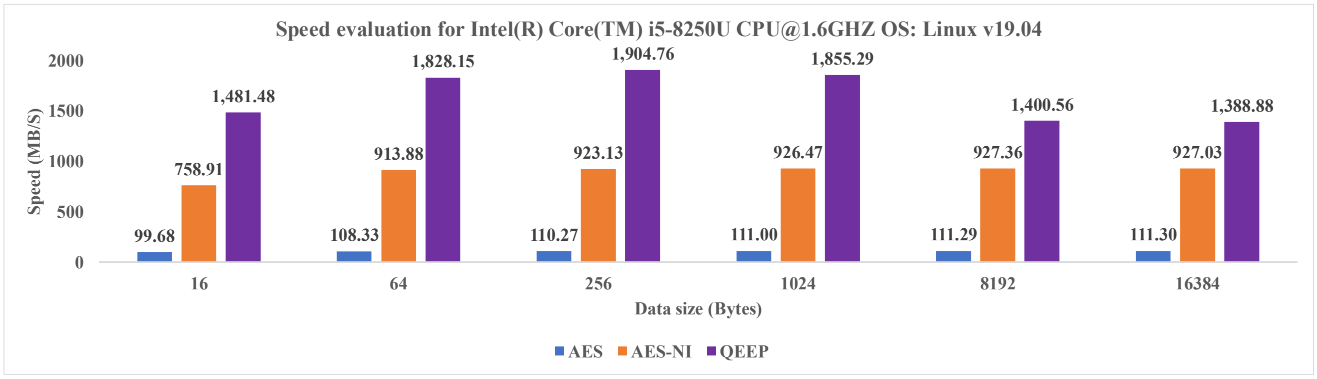
Figure 6: Platform: Intel(R) Pentium Silver J5005 CPU@1.5GHZ OS: Linux v18.10

- Intel(R) Pentium Silver J5005 CPU@1.5GHZ OS: Windows 10 (Table 4 and Figure 8)
- ARMv8 Processor rev 1 (v81) OS: JetPack 4.2 (L4T 32.1) (Table 5 and Figure 9)
- ARMv7 Processor rev 4 (v71) OS: Raspbian GNU/Linux 9 (Table 6 and Figure 10)

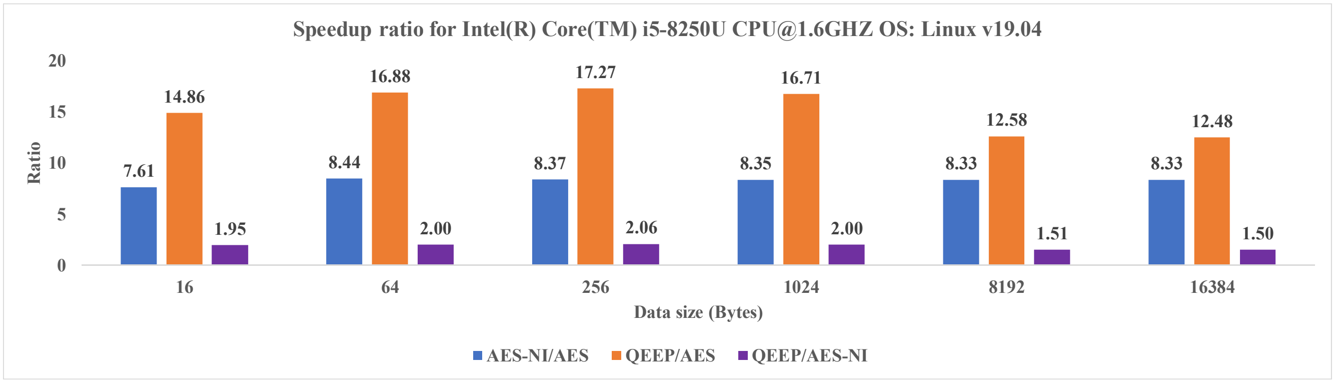
The results as depicted in the performance analysis tables and figures, show that QEEP is significantly faster than AES and is faster than AES-NI on all platforms. QEEP on Intel(R) Core(TM) i5 processor can encode around 2GB/S. QEEP is up 18.4 times faster than AES and up to 2 times faster than AES-NI. Speed is measured by megabytes per second and the second half of the tables show the ratios between AES-NI and AES, QEEP and AES, QEEP and AES-NI, respectively. Results also show that the performance on linux in general for AES, AES-NI, and QEEP is better than windows OS for the same CPU.

Table 3: Performance Comparison with different data sizes
Platform: Intel(R) Core(TM) i5-8250U CPU@1.6GHZ OS: Linux v19.04

		16 bytes	64 bytes	265 bytes	1024 bytes	8192 bytes	16384 bytes
Speed (MB/S)	AES	99.68	108.33	110.27	111.00	111.29	111.30
	AES-NI	758.91	913.88	923.13	926.47	927.36	927.03
	QEEP	1,481.48	1,828.15	1,904.76	1,855.29	1,400.56	1,388.88
Ratio	AES-NI/AES	7.61	8.44	8.37	8.35	8.33	8.33
	QEEP/AES	14.86	16.88	17.27	16.71	12.58	12.48
	QEEP/AES-NI	1.95	2.00	2.06	2.00	1.51	1.50



(a) Speed evaluation

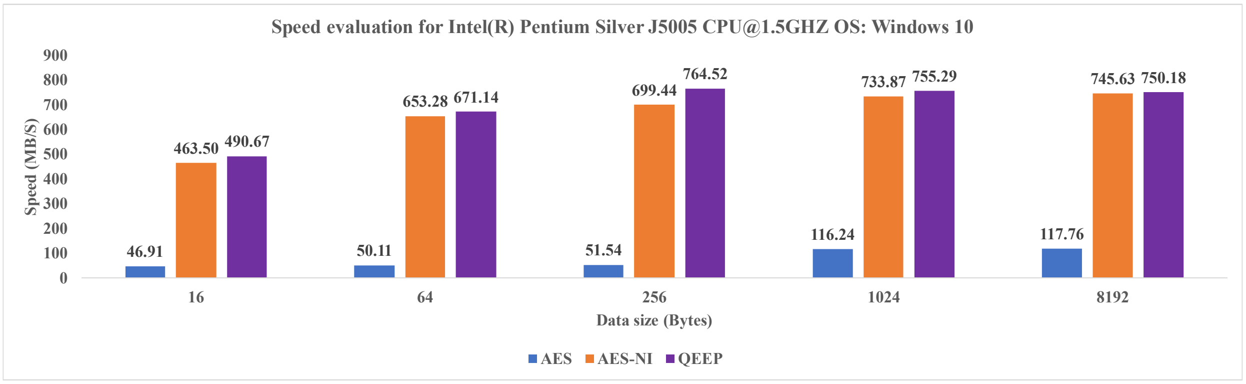


(b) Speedup ratio

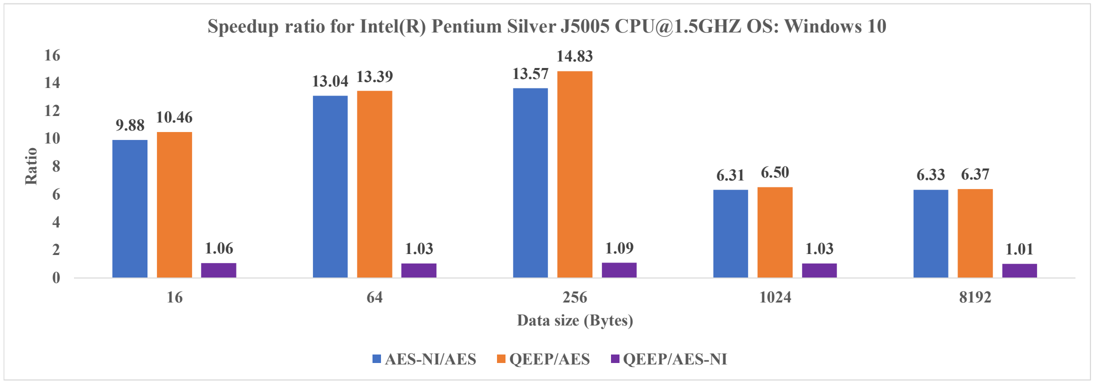
Figure 7: Platform: Intel(R) Core(TM) i5-8250U CPU@1.6GHZ OS: Linux v19.04

Table 4: Performance Comparison with different data sizes
Platform: Intel(R) Pentium Silver J5005 CPU@1.5GHZ OS: Windows 10

		16 bytes	64 bytes	265 bytes	1024 bytes	8192 bytes
Speed (MB/S)	AES	46.91	50.11	51.54	116.24	117.76
	AES-NI	463.50	653.28	699.44	733.87	745.63
	QEEP	490.67	671.14	764.52	755.29	750.18
Ratio	AES-NI/AES	9.88	13.04	13.57	6.31	6.33
	QEEP/AES	10.46	13.39	14.83	6.50	6.37
	QEEP/AES-NI	1.06	1.03	1.09	1.03	1.01



(a) Speed evaluation

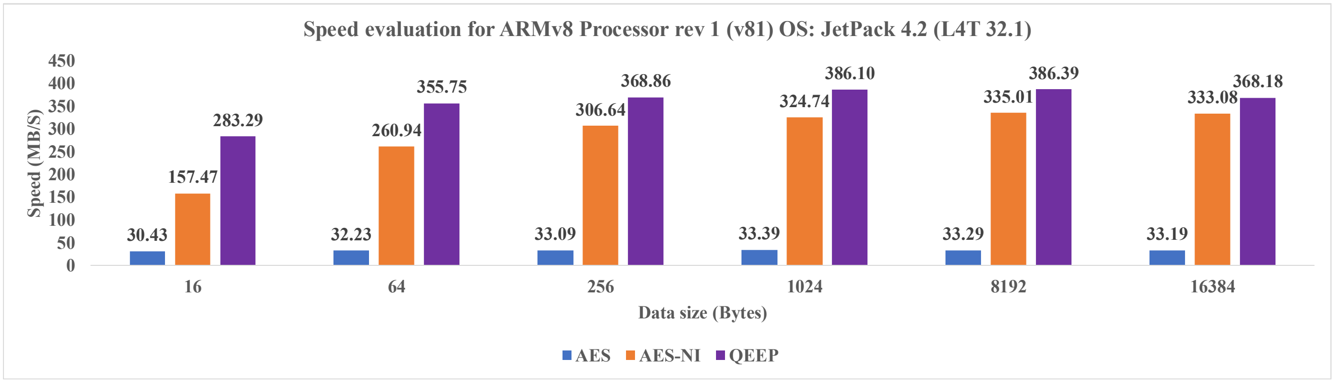


(b) Speedup ratio

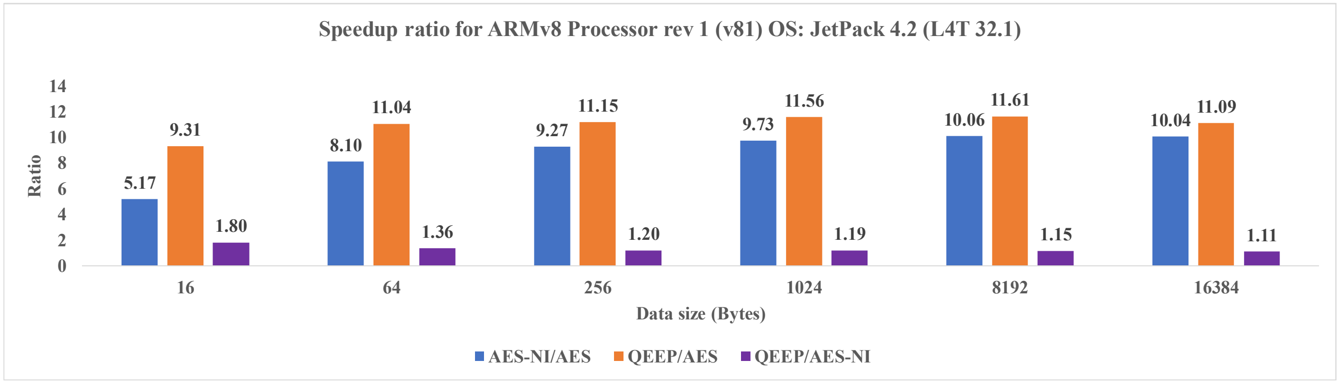
Figure 8: Platform: Intel(R) Pentium Silver J5005 CPU@1.5GHZ OS: Windows 10

Table 5: Performance Comparison with different data sizes
Platform: ARMv8 Processor rev 1 (v81) OS: JetPack 4.2 (L4T 32.1)

		16 bytes	64 bytes	265 bytes	1024 bytes	8192 bytes	16384 bytes
Speed (MB/S)	AES	30.43	32.23	33.09	33.39	33.29	33.19
	AES-NI	157.47	260.94	306.64	324.74	355.01	333.08
	QEEP	283.29	355.75	368.86	368.10	386.39	368.18
Ratio	AES-NI/AES	5.17	8.10	9.27	9.73	10.06	10.04
	QEEP/AES	9.31	11.04	11.15	11.56	11.61	11.09
	QEEP/AES-NI	1.80	1.36	1.20	1.19	1.15	1.11



(a) Speed evaluation

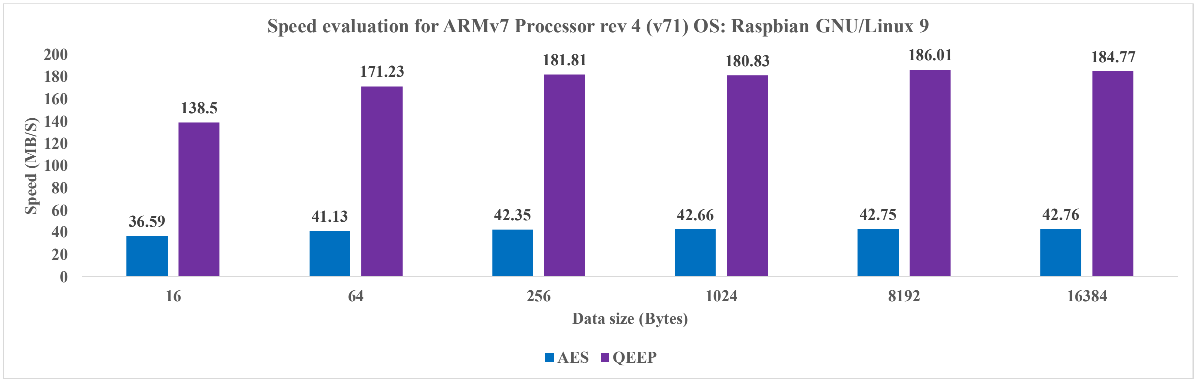


(b) Speedup ratio

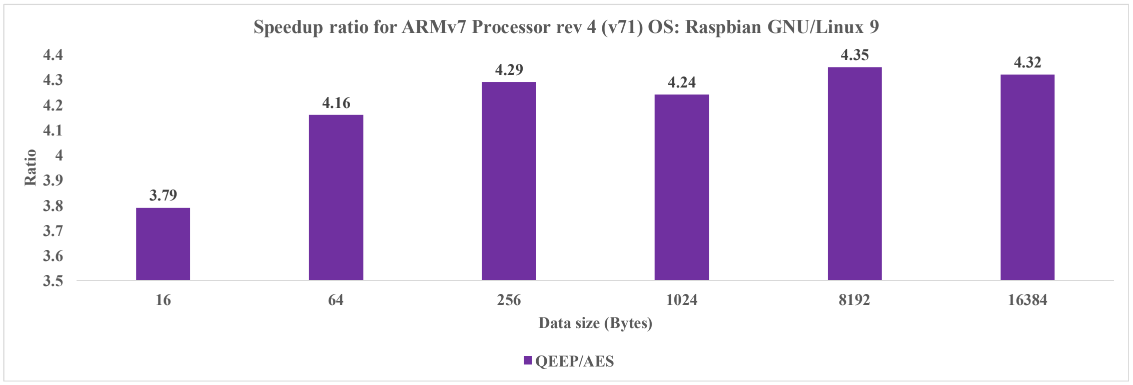
Figure 9: Platform: ARMv8 Processor rev 1 (v81) OS: JetPack 4.2 (L4T 32.1)

Table 6: Performance Comparison with different data sizes
Platform: ARMv7 Processor rev 4 (v71) OS: Raspbian GNU/Linux 9

		16 bytes	64 bytes	265 bytes	1024 bytes	8192 bytes	16384 bytes
Speed (MB/S)	AES	36.59	41.13	42.35	42.660	42.75	42.76
	QEEP	138.50	171.23	181.81	180.83	186.01	184.77
Ratio	QEEP/AES	3.79	4.16	4.29	4.24	4.35	4.32



(a) Speed evaluation



(b) Speedup ratio

Figure 10: Platform: ARMv7 Processor rev 4 (v71) OS: Raspbian GNU/Linux 9

4 POWER CONSUMPTION ANALYSIS

In the power consumption experiments, we measure QEEP power consumption and compare it with AES-256-CBC and AES-NI-256-CBC. For ARM processors, we use a physical power meter device to measure the power consumption and for intel processor, we use powerstat tool [6] to measure the consumed battery power.
We test the speed in the following platforms:

- ARMv7 Processor rev 4 (v71) OS: Raspbian GNU/Linux 9 (Table 7)
- ARMv8 Processor rev 1 (v81) OS: Ubuntu 18.04.2 LTS (Table 8)
- Intel(R) Core(TM) i5-8250U CPU@1.6GHZ OS: Linux v19.04 (Table 9)

Energy is a desirable property for IoT space, where longer lasting devices deployed in the field can create a significant competitive advantage. Also, higher speed, and lower power footprint mean reduced thermal footprint, which is desirable in constraint environments. QEEP is an optimal energy saving solution. The results as depicted in the power consumption analysis tables and Figure 11, show that QEEP uses much less power than AES and less than AES-NI on all platforms. QEEP on Intel(R) Core(TM) i5 processor uses around 1.4 J/GB. QEEP saves up to around 95% of consumed power by AES and saves up to around 40% of consumed power by AES-NI. AES consumes between 20 to 30 joule per Gigabyte, AES-NI consumes between 2.2 to 2.7 joule per Gigabyte, while QEEP consumes between 1.4 to 4.3 joule per Gigabyte. AES-NI is not implemented on ARMv7.

5 CONCLUSION

Quantropi provides a novel usage of quantum classical logic behaviour that can be represented by permutation gates for implementing reusable perfect secure communications. Quantropi's approach can be implemented in quantum and classical computing systems. In quantum implementation, the classical logic behavior, can be represented by real quantum permutation gates and implemented in pure quantum registers to perform permutation operations on qubits system. In classical implementation, the classical behaviour can be represented by permutation matrices (virtual quantum permutation gates). This novel approach can be used in various quantum and classical communication and security use cases.

In this report, we show a sample of our effective classical implementation, called Quantum Entropy Encoding Protection (QEEP). We evaluate QEEP from security, performance and power consumption perspectives. In security experiments, we measure the output randomness based on entropy and randomness online tester, Dieharder and NIST statistical test suite. Performance and power consumption are tested across multiple platforms. Our results show that QEEP achieves the same security level as AES-256-CBC with better performance and less power consumption than AES and AES-NI.

ACKNOWLEDGMENT

This research is partially funded by Mitacs Canada https://www.mitacs.ca/en

REFERENCES

[1] Entropy and randomness online tester, online at https://servertest.online/entropy, (2019).
[2] Centre for quantum computing and communication technology, welcome to the ANU quantum random numbers server. online at https://qrng.anu.edu.au/, (2019).
[3] Brown, R. G. Dieharder: a random number test suite. online at https://webhome.phy.duke.edu/~rgb/General/dieharder.php, (2019).
[4] Sýs, M. Říha, Z. & Matyáš, V. Algorithm 970: optimizing the NIST statistical test suite and the berlekamp-massey algorithm. ACM Transactions on Mathematical Software, Association for Computing Machinery **43**, 3, 27-37 (2017).
[5] OpenSSL cryptography and SSL/TLS toolkit, online at https://www.openssl.org/.
[6] Powerstat. online at http://manpages.ubuntu.com/manpages/xenial/man8/powerstat.8.html, (2015).

ABOUT THE AUTHORS

Yu Rang Kuang is the Co-founder and Chief Scientific Officer of Quantropi. Randy holds a doctorate degree in quantum physics and has worked with small and large IT companies, including Nortel. In 2009, he co-founded inBay Technologies Inc., serving as CTO and leading the development of a cloud-based, passwordless authentication and authorization platform. A prolific inventor, he has been granted 29 U.S. patents in broad technology fields, including key concepts and technologies for quantum key distribution, specifically addressing industrial applications.

Eslam G. AbdAllah is a Postdoctoral Fellow in Carleton University. Eslam received the PhD degree from the School of Computing, Queen's University, Kingston, ON, Canada on 2017. He is a postdoctoral fellow at Quantropi Inc. and Carleton university. His research interests include cryptography, network security, quantum security, information centric networks, autonomous vehicles, and RFID. He received the best paper award in IEEE DASC 2015, Liverpool, UK.

Table 7: Power Consumption for ARMv7 Processor rev 4 (v71) OS: Raspbian GNU/Linux 9

	Baseline	AES	QEEP
Voltage (V)	5.19	5.19	5.19
Current (A)	0.52	0.73	0.67
Power (W)	2.69	3.74	3.46
Power usage	-	1.05	0.77
Speed (1K data size) (MB/S)	-	42.62	182.28
Energy per GB (J/GB)	-	24.65	4.3

Table 8: Power Consumption for ARMv8 Processor rev 1 (v81) OS: Ubuntu 18.04.2 LTS

	Baseline	AES	AES-NI	QEEP
Voltage (V)	5.16	5.16	5.16	5.16
Current (A)	0.31	0.6	0.57	0.67
Power (W)	1.6	3.1	2.95	3.2
Power usage	-	1.5	1.35	1.6
Speed (1K data size) (MB/S)	-	51.76	504.89	593.11
Energy per GB (J/GB)	-	28.9	2.7	2.7

Table 9: Power Consumption for Intel(R) Core(TM) i5-8250U CPU@1.6GHZ OS: Ubuntu 18.04.2 LTS

	Baseline	AES	AES-NI	QEEP
Voltage (V)	-	-	-	-
Current (A)	-	-	-	-
Power (W)	5.19	7.37	7.31	7.67
Power usage	-	2.18	2.12	2.48
Speed (1K data size) (MB/S)	-	110.07	931.52	1,855.29
Energy per GB (J/GB)	-	20	2.2	1.4

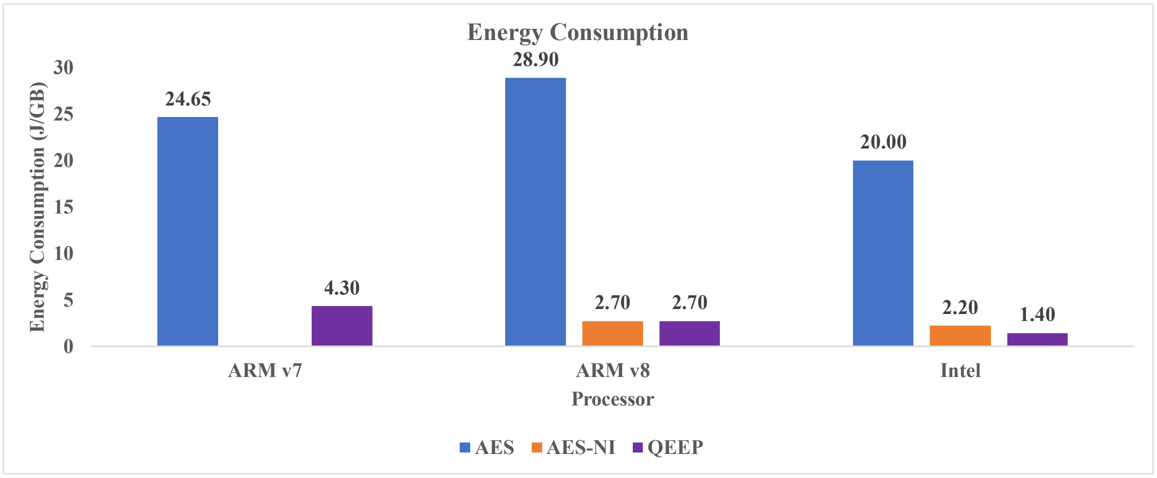


Figure 11: Energy Consumption