Quam Bene Non Quantum: Bias in a Family of Quantum Random Number Generators

Darren Hurley-Smith and Julio Hernandez-Castro

School of Computing, University of Kent, Canterbury CT2 7NF, Kent, UK
dh433@kent.ac.uk, jch27@kent.ac.uk

Abstract. Random number generation is critical to many security protocols, a basic building block on which it rests the robustness of many security solutions. Quantum physics, on the other hand, offers a very attractive approach to True Random Number Generation, based on the inherent randomness of some physical phenomena. Naturally, there are a number of quantum random number generators in the market. In this work, we present the first analysis of a popular commercial family called Quantis, designed and manufactured by ID Quantique. We subject their output to three batteries of statistical tests, for evaluating its performance. Dieharder and NIST STS 2.1.2 are included in many certification schemes, whilst ENT provides a free, simple and powerful means of expanding on the previous tests. The Quantis devices under examination have achieved METAS and other independent certifications and indeed the results over the Dieharder and NIST batteries confirm that the certifications awarded are based on an acceptable performance on both sets of tests. However, ENT finds strong evidence of significant biases in the Quantis devices. These biases are analyzed to identify their traits and attempt to isolate their root cause. We end with a discussion on the need to expand testing strategies to incorporate lesser-known tests that regularly detect problems that the commonly accepted batteries do not.

Keywords: quantum random number generation · entropy · cryptography · statistical analysis

1 Introduction

True random number generators (TRNG), harness circuitry designed to gather entropy from a hardware source, process the output and present it to a host device. USB devices such as the Chaos Key [1], Araneus II and Swift Key are examples of such devices. Many aim to extend the capabilities of PC's, which may not gather sufficient entropy from system resources and user interaction to provide reliable PRNG output over extended periods of time. TRNGs tend to make use of noise generated using classical physical phenomena as their entropy source.

Quantum random number generators (QRNG) are a more recent innovation. Rarity et al. discuss the possibility of combining quantum random number generation and key sharing as far back as 1994 [2]. Stefanov et al. outline the scientific principles and design of an optical quantum random number generator in 2000 [3]. In their work, they propose a quantum phenomenon by which photons may either pass-through or reflect off of a partially mirrored surface. The probability with which it will achieve either result is approximately 0.5, and the output stream is inherently unpredictable. Its use as a source of randomness is intuitive.

ID Quantique were the first to harness quantum phenomena as an entropy source in a commercial product, in 2001[1] [4]. This Swiss company has commercialized devices that use an optical quantum process as a source of randomness [5]. This has allowed them to generate random numbers at a speed of 4Mbit/s and 16Mbit/s on PCI-E and USB hardware devices. The basic principles of the quantum entropy source used by ID Quantique can be seen in Figure 1.

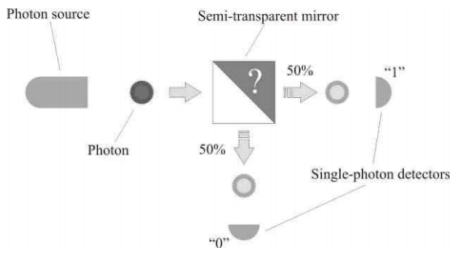


Fig. 1: Quantis RNG functionality overview [5]

Figure 2 outlines the different subsystems of the Quantis QRNG. A source is used to generate individual photons, which are directed towards a semi-transparent mirror. This mirror has a 0.5 probability of emitting or reflecting the photons directed towards it. Two photon detectors, labeled "0" and "1", are incorporated to track which of these eventualities occurs. If the photon is emitted a "1" will result. If it is reflected a "0" is generated.

An unbiasing algorithm is used to ensure that the each of the outputs of the QRNG has a probability of 0.5. It is not possible to guarantee that this probability will arise naturally, so the unbiasing function is needed to ensure that values do not drift. An additional status monitoring scheme is used to ensure that the QRNG hardware (emitter and detectors) remains operational and in optimal working condition. If errors are detected, the device will notify the user and stop generating output.

[1] http://www.idquantique.com/random-number-generation/

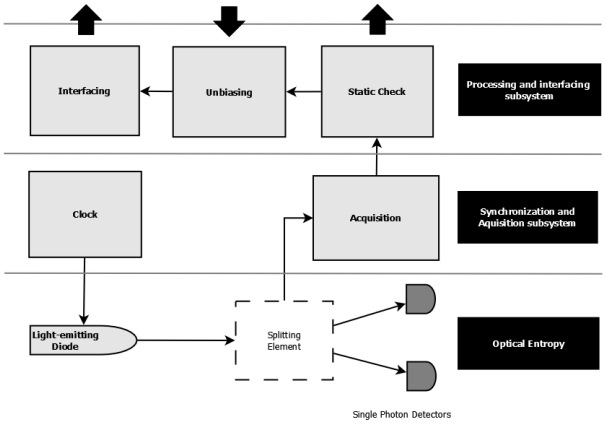


Fig. 2: Quantis RNG block diagram [5]

Quantis devices, particularly the PCI-E 16M, PCI-E 4M, and USB 4M, have been *certified* internally[2][3]. External certification by the Swiss Federal Office of Metrology (METAS) has also been awarded, with the Diehard tests[4] forming the basis of their testing strategy [7]. The Compliance Testing Laboratory (CTL) has also certified these devices as suitable for use in gaming and gambling, under the auspices of the UK Gambling Commission's standards[5].

These modules are marketed as suitable for cryptographic use, lotteries, and online gaming. However, ID Quantique explicitly states that these devices are not BSI AIS 31 certified [8]. Dodis et al. comment on the impossibility of cryptography with imperfect randomness, raising questions regarding the suitability of any sufficiently biased RNG for cryptographic purposes [9]. A notable use-case cited in ID Quantique's press resources is the utilisation of Quantis devices similar to the ones under scrutiny by the Loterie Romande[6]. Mechanical draws, being complex and time-consuming, were not sufficient to meet the demands of this lottery, and so the Quantis platform was selected as part of their modernization efforts. METAS certification was augmented with independent mathematical analyses, with Loterie Romande deeming the device suitable for their purposes. To add context, this Swiss lottery generated a revenue of $100 million in 2010, a significant sum of money which is heavily dependent on the trustworthiness of the randomness of Quantis.

[2] The NIST SP800-22 Test Suite is used by ID Quantique [6]

[3] http://marketing.idquantique.com/acton/attachment/11868/f-0117/1/-/-/-/-/-/Quantis%20Certifications%20Collection.pdf

[4] Diehard tests performed over 10 x 100MB samples from Quantis QRNG devices. Passing these tests seems to be sufficient grounds to award the certificate

[5] Certificate issued 30/03/2011 after successfully passing required tests

[6] http://marketing.idquantique.com/acton/attachment/11868/f-0042/1/-/-/-/-/-/Loterie%20Romande%20QRNG%20for%20Swiss%20Lottery.pdf

This paper provides the first in depth analysis of the Quantis QRNG 16M, 4M and USB modules [5], developed and sold by ID Quantique. Going far beyond the testing regiments needed to achieve certification by the Swiss Federal Office of Metrology (METAS), Dieharder, NIST 2.1.2 and ENT are used to give a thorough statistical analysis of the output of these devices. Comparative analysis is provided through a study of the Chaos Key TRNG (a hardware random number generator in a USB package), and *urandom* (the kernel level PRNG provided in Linux operating systems). Significant biases are identified in the Quantis QRNG modules, and further analysis is conducted in an attempt to characterize these biases. To the best of the authors' knowledge, these findings are novel and highlight a previously unknown weakness in the studied QRNGs which could potentially have a major impact in certain applications. Some additional discussion is included regarding the testing strategies used to certify RNG products and the need for more stringent tests.

The rest of the paper is laid out as follows. Section 2 discusses methodology, with a technical breakdown of the devices and testing strategies employed in the course of this research. Section 3 reports on the results of statistical testing and presents an analysis of the results. Section 4 concludes the paper, with subsection 4.1 providing an overview of current and future work that will expand on the findings reported in this paper.

2 Methodology

Three Quantis quantum random number generators (QRNG) were selected for testing. These devices come in two types, PCI-E and USB connected modules. The PCI-E modules, shown in Figures 3(a) and (b) possess an output speed of 16Mbit/s and 4Mbit/s, respectively. The USB module shown in (c) has an output speed of 4 Mbit/s.

(a) PCIE 16M (b) PCIE 4M (c) USB 4M

Fig. 3: Quantis devices used in this research

The 4Mbit/s devices have a single module (each containing an optical quantum entropy source), while the 16Mbit/s model has 4x4M modules and a mixing

algorithm implemented in hardware, to provide the required entropy for its faster rate. The data sheet for these devices states that they are suitable for cryptography, statistical research, and PIN number generation. These devices are retailed at a relatively high price, commanding between €2990 and €990 for the devices previously listed[7].

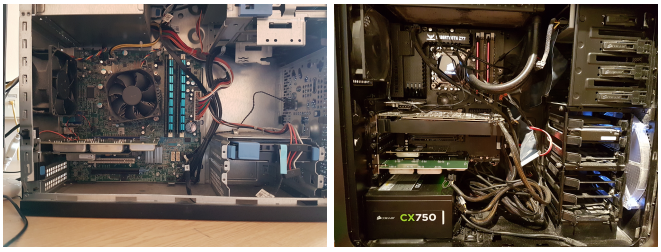
Each module has been issued a certificate that explicitly states that the devices are not BSI AIS 31 certified. An example of this can be seen in Figure 4. A unique identifier for the device in question is provided on the certificate. It is stressed that these QRNGs have been rigorously tested to ID Quantique's exacting specifications. Furthermore, it is stated that these devices are in compliance with the certificate n. 151-04687 of the Swiss Federal Institute of Metrology (METAS) [10].

Fig. 4: Quantis Certificate

The Quantis modules were tested on two different operating systems, on two different machines. An Ubuntu workstation with an Intel i7 3770k processor and 16GB of RAM is shown in Figure 5 (a). A Windows 10 machine with an i7 3770k processor and 16GB of RAM can be seen in Figure 5 (b).

The purpose of using two separate machines was to isolate, and possibly identify, the role of a given operating system or machine in producing positive or negative results. The use of machines with a similar processor and RAM specifications was intended to minimise the role of differing local hardware that may be important in the gathering and caching of data from each Quantis device.

[7] Prices from http://www.idquantique.com/random-number-generation/order-online/ at time of writing: Quantis PCI-E 16M: €2990. Quantis PCI-E 4M: €1299. Quantis USB 4M: €990



(a) Ubuntu Workstation (b) Windows PC

Fig. 5: Different platforms and Operating Systems were used for testing the output of the Quantis QRNGs

On both Windows 10 and Ubuntu 16.04, the Quantis 'Easy Quantis' application was used to collect 2GB[8] of random data from each module. Figure 6 shows the Windows 10 interface for this application.

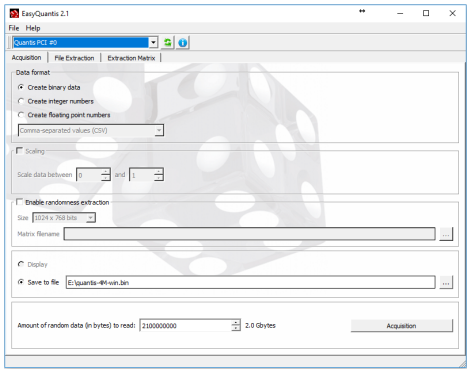


Fig. 6: Easy Quantis Application (Win 10)

Data will also be collected from a Chaos Key[9] (Figure 7), which is another USB package containing a (non-quantum) hardware random number generator. The *urandom* function under Ubuntu 16.04 LTS will also be tested, as this is a commonly used, high-speed software PRNG.

The data collection method for both the Chaos Key and *urandom* involved the use of the "dd" utility to print random values from each device to a binary file. Sequences of 128 bytes were extracted 16,406,250 times to achieve a final file size of 2GB for both RNGs. This is an important difference to note with the Quantis devices, which provide an application for data extraction, while the

[8] This is the maximum size allowed by the application.
[9] The Chaos Key retails at €40 from a variety of third-party resellers

Fig. 7: The Chaos Key USB TRNG [1]

Chaos Key and *urandom* are both supported by the Linux kernel. The Quantis devices also require driver installation and a roll back to kernel v3 otherwise they will throw a 'version not recognized' error when compiling. To ensure similar testing conditions under Ubuntu, all the Quantis devices, *urandom* and the Chaos Key were all tested with the v3 of the kernel.

The 2GB samples extracted from these RNGs will be used for comparative analysis of the Quantis devices with unrelated hardware and software RNGs. The following data was collected, in the listed amounts and from the named operating systems:

- Quantis 16M PCI-E module: 2GB x 3 (Ubuntu), 2GB x 1 (Windows)
- Quantis 4M PCI-E module: 2GB x 3 (Ubuntu), 2GB x 1 (Windows)
- Quantis USB 4M module: 2GB x 3 (Ubuntu), 2GB x 1 (Windows)
- Chaos Key TRNG USB module: 2GB x 1 (Ubuntu)
- *urandom* PRNG: 2GB x 1 (Ubuntu)

The speed of random number generation was benchmarked for all three devices, with the results shown in Table 1.

Table 1: Time taken to extract 2GB of data from all tested RNGs

	Time 1 (s)	Time 2 (s)	Time 3 (s)	Mean time (s)	Mean data rate (Mbit/s)
Quantis PCI-E 16M	1011.50	1024.72	1140.45	1058.90	15.87
Quantis PCI-E 4M	4321.45	4403.55	4303.37	4342.79	3.86
Quantis USB 4M	4300.66	4219.05	4203.70	4241.14	3.96
Chaos Key	4523.67	4399.98	4316.75	4413.47	3.80
urandom	330.15	334.67	331.47	332.10	50.59

These results confirm that the devices function close to their marketed speeds, and well within the 10% bound specified in the product specification document. The maximum time taken for the USB and 4M modules to provide a 2 GB sample was 1 hour and 7 minutes. By comparison, the 16M generated 2GB of

data in approximately 17 minutes. Similar values were achieved during Windows 10 data extraction.

The Chaos Key and *urandom* have been subject to the same three-sample speed test to provide a fair comparison with the Quantis devices. The Chaos Key performs almost as quickly as the 4M modules, while *urandom* is significantly faster. The primary issue with *urandom* is, of course, that it does not wait for its entropy pool to replenish (it is nonblocking). This may mean that for larger samples, and in low entropic environments such as virtual machines, it may generate low-quality output without stopping or reporting this behavior though this was not observed during our experiments.

A variable to bear in mind is the relative cost of these devices. The Quantis QRNG modules are relatively expensive when compared with other hardware random number generators. The Quantis PCI-E 16M is priced at €2990, while the PCI-E 4M costs €1299. The Quantis USB 4M module is priced at €990. The Chaos Key, by comparison, is a mere €40, in a much smaller package.

The collected data will be tested in accordance with the statistical testing scheme outlined in the following subsection.

2.1 Randomness tests

The output of the Quantis devices, the Chaos Key and *urandom* were subject to a battery of statistical tests. The Quantis devices were tested using Dieharder, NIST STS 2.1.2, and ENT. All tests will be conducted on an Ubuntu work station to ensure consistent testing conditions.

Diehard Tests. The Dieharder battery is an extension of the original Diehard tests. It incorporates a variety of statistical tests and forms the basis for RNG evaluation. Dieharder is not exhaustive in its testing, despite possessing over a dozen individual test types with multiple permutations of each. However, it is still considered a good gauge of the robustness of a generator, with failure on any test being an indication of a flawed RNG. It is a part of many certification test strategies, including BSI's AIS 31.

The Quantis devices, *urandom* and Chaos Key will be analyzed using Dieharder. All tested samples were run through Dieharder at their full length of 2GB. Default values are used for all testing, and the file is rewound to the beginning for each test.

NIST STS 2.1.2. The National Institute of Standards and Technology (NIST) offers the Statistical Testing Suite (STS) for the analysis of randomness [11]. This software provides a more intuitive user interface than Dieharder and allows the size and number of bit-streams to be defined. It is possible to select which tests are used, but for the purposes of this research, all tests have been employed.

Due to the significant time it takes for larger files to be analyzed by this test, the Quantis devices, *urandom* and the Chaos Key have been analyzed for a stream size of 1,680,000 bits and 100 bit-streams.

ENT. ENT is a utility for evaluating pseudo-random number generators [12]. The entropy, compression, chi-square test, arithmetic mean, Monte Carlo value for π and serial correlation coefficient are included. The most stringent of these is the chi-square test, which is used to determine if a data source is uniform [13]. In previous work, we have found ENT to be an under-rated bias detection suite [14].

ENT is not part of any recommended RNG testing strategy but has been used to find significant and persistent biases in the DESFire EV1 RNG despite it passing all recommended tests. As a result, this battery will be applied to all data collected during this research. All files were analyzed for their full length of 2GB.

3 Results

We present in this section the concise version of each test battery's results. These are discussed in their respective subsections, with analysis of unusual results following after. The length of most test reports makes displaying them in this format ill-advised. As a result, only a short form of the results will be shown where possible.

Due to the limitations of print on displaying extensive statistical reports, a GitHub repository with our results has been made available for general viewing[10]. The sample data used for these tests is of a substantial size (a total of 28GB in 14 binary files) but is available upon request to the authors.

3.1 Dieharder

The Quantis 16M PCI-E module reports between two and three weak results, all rgb lagged sum tests, across its Ubuntu samples. The n-tuple value associated with the weak tests varies from file to file, with no clear pattern, but it is always this test that is reported as weak. However, the 16M does not fail any tests, under Ubuntu.

The sample extracted under Windows 10, however, reports multiple weak tests and several failures under Dieharder. The failed tests include:

- diehard bitstream (ntuple 0)
- sts serial (a total of 5 failures over ntuple 13, 15, 15, 16, and 16)
- rgb lagged sum (ntuple 31)

This is a significant degree of failure and warrants further analysis. Retesting of the sample file reported the same results, indicating that the failures are associated with the generator and not anomalous test conditions. At this stage of the research, it is likely that this is a result of the QRNG producing a set of odd values, as the other three samples do not appear to have any problems with Dieharder. The other modules similarly have no significant difference in results between their Ubuntu and Windows samples. Further research is needed.

[10] https://github.com/DarrenHurleySmith/QuantisRNGData.git

The 4M module doesn't share these results. For the Windows 10 sample of the 4M, a single sts serial test reports a weak result, as do three rgb lagged sums tests, but it passes the battery under the same conditions that the 16M fails it. Thee Ubuntu results for this module look much the same: one sample reports 7 weak results, but none report any failures. This indicates that the 4M has passed the Dieharder tests, but may require further scrutiny to ascertain why it is considered weak for so many tests in some of its sample outputs.

The Quantis USB module reports 6 weak results and no failures, for the Windows 10 sample. It also passes the battery, with a high of 5 weak results on its Ubuntu samples, with the other two samples reporting 4 weak results. As before, this indicates that the Dieharder test suite has been passed by this generator.

The Chaos Key passes all Dieharder tests. Three weak results are reported: rgb minimum distance (n-tuple 2), rgb lagged sum (n-tuple 27), and dab monobit (n-tuple 12). These are not indicators of failure, but highlight that the results are too close for comfort and further analysis would be beneficial. The Chaos Key can be considered as passing the Dieharder battery.

The *urandom* RNG passes the Dieharder battery. Only three tests report weak results, two in rgb lagged sums tests and 1 in sts serial (n-tuple 11). As a result *urandom* can be considered as having passed the Dieharder battery, despite the previously discussed issues with *urandom* being a nonblocking algorithm. It appears that even with a large file and fast polling speed, *urandom* looks sufficiently random for this test.

These results show that generally, the Quantis devices pass the Dieharder battery, as does the Chaos Key. The failures of the 16M module under Windows must be analyzed further, but at present appear to be anomalous: they do not correspond with the behavior of the module across three Ubuntu samples, nor does it match the results of the other modules' Windows 10 sample results.

3.2 NIST STS 2.1.2

The Quantis 4M reports a borderline result for the RandomExcursionsVariant test in each Ubuntu sample. It reports a borderline NonOverlappingTemplate result for the Windows 10 sample. All of these borderline results are 1 point below threshold (95/100 instead of 96/100). They do, however, have reasonable p-values, indicating that the status of the test is uncertain. With only one failed (borderline) test in each sample, the 4M is a cause for concern, but not yet statistically proven to have any non-random characteristics.

The USB module passes all but one test in its Ubuntu samples. In each Ubuntu sample a Non-overlapping test is failed, but only to the degree seen in the 4M PCI-E module (1 point below the threshold and with good p-values). This indicates, as with the PCI-E module, that the RNG requires further scrutiny, but it may just be statistical noise. It should be noted that the Ubuntu samples report these borderline values at different stages (as there are multiple such tests in each run of STS 2.1.2). Therefore, it varies with differing data from the same device, indicating that if there is an underlying bias, it is not consistent.

Under Windows, the Quantis USB module reports failures for the FFT test and a single NonOverlappingTemplate test. The pass rates for both are 94/100 with p-values well within acceptable bounds. Although these results are only slightly worse than those of the Ubuntu samples, they do indicate a failure in the named tests. This indicates that there are issues with the randomness of the collected files and that additional scrutiny must be levied against this device. As ENT will perform a byte-by-byte analysis of the sample file, this is a good way to deepen the investigation of such results.

Once again, the 16M Windows 10 sample paints a very poor picture. This sample fails two serial tests completely, falling way below the acceptable threshold of 96/100 tests passed, reporting values of 82 and 88. No other tests report even borderline results: they are all passed with a safe margin. This weakness in serial tests matches that observed in the Dieharder results for this module. The 16M sample collected on the Windows 10 machine exhibits weakness to serial tests.

The serial test, for both NIST and Dieharder, analyses the frequency of each and every overlapping m-bit pattern. This is done across the entire sequence being tested. Failing this test indicates that the generator has produced output with a greater number of 2m overlapping m-bit patterns than would be expected in a truly random sequence. It must be stressed, however, that this failure is confined to the single sample. All other devices and all other 16M samples appear to perform well in this test.

The 16M passes every single test in each Ubuntu sample. This further reinforces the earlier intuition that the Windows sample is in some way anomalous, despite being generated under the same variables, and with the same speed, as all of the Ubuntu samples. As a result, the Windows 10 Ubuntu sample will henceforth be considered unrepresentative of the 16M PCI-E module in general, but useful for comparative analysis. Further tests on Windows machines will be required to ascertain if this is typical behavior or not.

The Chaos Key reports two borderline results for NonOverlappingTemplate tests. It has this in common with the Quantis 4M and USB modules. Both tests barely fail, with p-values within the threshold, but test pass rates are 95/100 instead of 96/100. *urandom* passes all of the tests in this suite. No borderline or failed results are reported. In this respect, it has surpassed all of the QRNGs and TRNGs modules tested in this research, a surprising finding considering the cost inherent to hardware random number generators when compared to a free kernel module PRNG.

3.3 ENT Results

Table 2 displays the output of ENT for all tested devices. The Quantis devices are separated into their respective samples, *urandom* and the Chaos Key both provide a single entry each.

Chaos Key and *urandom* perform well across all of the tests. Notably, they have chi-square values close to 256, the optimal value. Their p-values are similarly

Table 2: ENT results for Quantis, Chaos Key and *urandom*

	Chi-square	p-value	Entropy	Arith. Mean	π	Serial Corr.
16M 1 (Ubuntu)	339.98	$3.40 \cdot 10^{-4}$	8	127.49	0%	$1.50 \cdot 10^{-6}$
16M 2 (Ubuntu)	381.81	$5.48 \cdot 10^{-7}$	8	127.49	0%	$7.00 \cdot 10^{-6}$
16M 3 (Ubuntu)	305.14	$1.89 \cdot 10^{-2}$	8	127.49	0%	$1.00 \cdot 10^{-6}$
16M (Windows)	373.12	$2.37 \cdot 10^{-6}$	8	127.49	0%	$2.40 \cdot 10^{-6}$
4M 1 (Ubuntu)	536.41	$<1.00 \cdot 10^{-15}$	8	127.49	0%	$1.00 \cdot 10^{-6}$
4M 2 (Ubuntu)	464.65	$2.86 \cdot 10^{-14}$	8	127.49	0%	$8.00 \cdot 10^{-6}$
4M 3 (Ubuntu)	450.18	$7.42 \cdot 10^{-13}$	8	127.49	0%	$6.00 \cdot 10^{-6}$
4M (Windows)	553.77	$<1.00 \cdot 10^{-15}$	8	127.49	0%	$1.40 \cdot 10^{-5}$
USB 1 (Ubuntu)	507.04	$<1.00 \cdot 10^{-15}$	8	127.49	0%	$1.10 \cdot 10^{-6}$
USB 2 (Ubuntu)	450.35	$7.15 \cdot 10^{-13}$	8	127.49	0%	$2.40 \cdot 10^{-6}$
USB 3 (Ubuntu)	404.03	$9.82 \cdot 10^{-9}$	8	127.49	0%	$1.20 \cdot 10^{-6}$
USB (Windows)	436.95	$1.30 \cdot 10^{-11}$	8	127.50	0%	$3.20 \cdot 10^{-5}$
Chaos Key	262.41	$3.78 \cdot 10^{-1}$	8	127.50	0%	$9.00 \cdot 10^{-6}$
urandom	259.50	$4.10 \cdot 10^{-1}$	8	127.51	0%	$1.80 \cdot 10^{-5}$

greater than 0.01, putting them just in the center of the 0.01-0.99 range deemed acceptable for this test.

This is not true of the Quantis devices. Every single one of them, barring one exceptional 16M (Ubuntu) result, fails the chi-square test. They report good results for every other test, passing by a good margin on each. But the chi-square scores and p-values are exceptionally high. Tests on smaller samples derived from the files used for these tests found that many of these devices will pass the chi-square test at sizes of under 500MB (some even at 1GB or less), but the fact that *urandom* and Chaos Key both pass at a 2GB sample size shows that the Quantis devices should also be able to pass these tests if they are as robust a source of randomness.

The 4M PCI-E and USB modules perform most poorly. Both report scores in excess of 500, with their lowest scores being 450.17 (4M PCI-E), and 404.02 (4M USB). They report similar values from their Windows 10 samples, showing that there's no detectable influence caused by the choice of operating system. The 16M PCI-E module performs better in general but fails all but one Chi-square test on both Windows and Ubuntu. It is one of the Ubuntu samples that passes in a most borderline manner, with a score of 305.13 and a p-value of 0.07. This is barely a pass and is still a cause for concern.

Previous research into RFID based hardware random number generators, undertaken by the authors, has identified that seemingly robust RNGs can fail on these simplest and often overlooked tests [14]. This is especially true of hardware random number generators. A poor chi-square result indicates that there is a low degree of uniformity among byte values in the sample. The occurrence of values may be clustered around a specific point, or a pattern may have emerged, leading to a small but detectable structure in the frequency with which given byte values occur.

In our previous work with the DESFire EV1 TRNG this took the form of a sinusoidal structure, in which half of the possible byte values occurred fractionally more frequently than the other half, leading to a perceptible trend in the plotted occurrences of said bytes. To identify the presence of such a trend, further analysis is required and follows in the next subsection.

Analysis of the Quantis bias. To analyze the biases associated with the devices that failed the Chi-square test, the byte indices and rate of their occurrence in the sample files were collected. From this, the bias of each byte value for a given file can be computed, providing a visualization of the distribution of values throughout a file at the byte level.

Figure 8 shows the biases for the Quantis PCI-E 16M module. Figure 8 (a) shows the first set of data, with an exceptionally high occurrence of low-index bytes. The greatest bias is $4.69 \cdot 10^{-6}$, occurring on byte 0. Further biases that significantly deviate from the norm by more than $\pm 4 \cdot 10^{-6}$ can be observed for values 64, 65, 181, 184, and 231. This threshold value has been selected to focus our observations on the most extreme biases and to highlight the abnormal frequency with which they occur in what should be a randomly distributed sample. Even with the vagaries of 'true' randomness, in a sufficiently large sample, the distribution of byte values should converge to a normalized state. Significant biases indicate that this is not the case here.

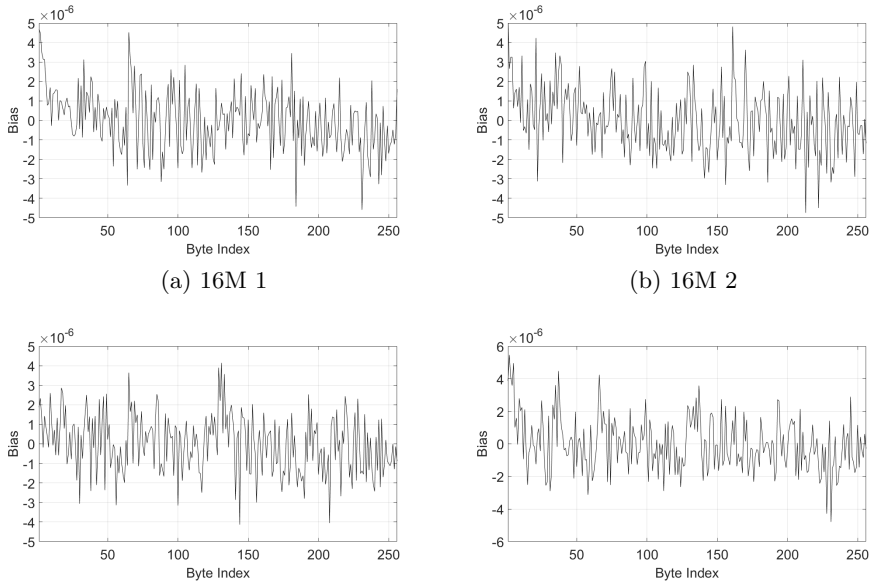
Graph (b) shares the value of its highest bias with all of the other samples for the 16M, with byte 0 possessing the maximum bias of $5.44 \cdot 10^{-6}$. Further biases above the $\pm 4 \cdot 10^{-6}$ threshold, can be observed at indices 21, 161, 213, and 222. The positions of these biases do not correlate with those observed in the first sample.

Figure 8 (c) bucks the trend set by the previous two samples, by not having value 0 as its highest bias. The maximum bias for this sample is $4.14 \cdot 10^{-6}$ for value 132. Beyond the $\pm 4 \cdot 10^{-6}$ threshold are values 144 and 208. As befitting this sample's reasonably good chi-square score, it exhibits far fewer extreme biases than the other 16M samples.

The Windows 10 sample, shown in (d), has its highest bias is at index 1, with a magnitude of $5.46 \cdot 10^{-6}$. This differs from (a) and (b) but the maximum is again attained at a very low value, matching the established trend closely. The rest of the sample possesses biases greater than $\pm 4 \cdot 10^{-6}$ at indices 2, 3, 36, 65, 227, and 230.

Figure 9 shows the biases of the Quantis PCI-E 4M module. Graph (a) demonstrates the biases of the first sample of 4M data, where index 129 has the highest bias of $5.92 \cdot 10^{-6}$. The next largest bias is on index 83, with the value $-5.77 \cdot 10^{-6}$. Indices 1, 2, 3, 4, 8, 65, 66, 83, 129, 172 and 187 all exceed the previously defined threshold value, indicating that the biases are more pronounced in this sample than in any of those of the 16M. The first indices of the sample are abnormally high, another element the 4M has in common with the 16M.

Graph (b) demonstrates the recurring low-index biases seen in previous results. Index 1 has the largest bias of this sample, with a magnitude of $5.39 \cdot 10^{-6}$.



(a) 16M 1 (b) 16M 2

(c) 16M 3 (d) 16M Windows

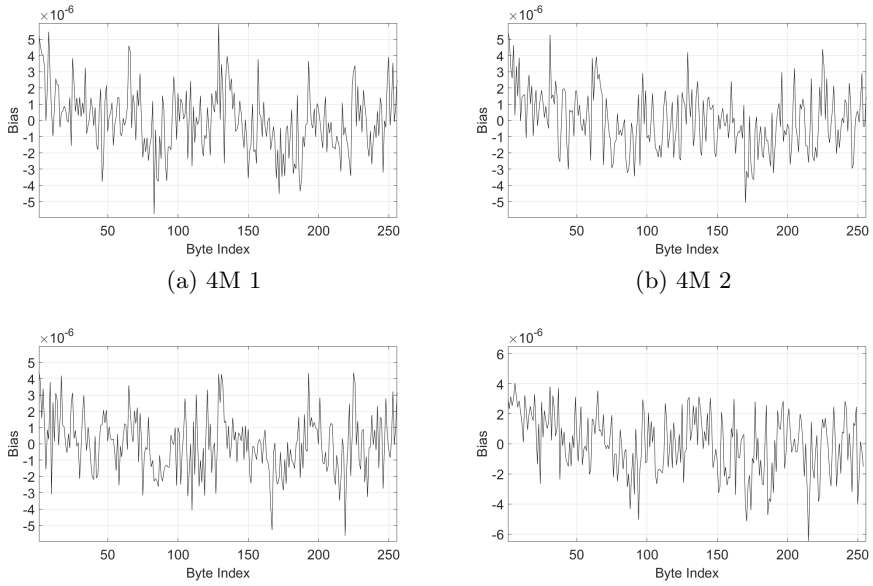
Fig. 8: Bias of Quantis 16M PCIE Module - Ubuntu machine

Indices 1, 2, 5, 31, 129, 170, and 225 all fall outside the $\pm 4 \cdot 10^{-6}$ threshold. This lesser number of extreme biases (compared with (a)), corresponds with the substantially lower (but still poor) chi-square score for this sample.

Graph (c) again exhibits highly biased low index values, index 1 possessing a bias of $4.41 \cdot 10^{-6}$. However, the largest bias is of magnitude $-5.63 \cdot 10^{-6}$, on index 219. Biases beyond the established threshold are found on indices 17, 110, 129, 131, 167, 193, 219, and 225.

Figure 9 Graph (d) shows the Windows 4M sample. Oddly, this sample has a large number of negative biases, with a relatively low number of biases in the positive range. It starts with biased low index values, as previously observed in the majority of 16M and 4M samples collected. The maximum bias observed has a magnitude of $-6.46 \cdot 10^{-6}$, at index 215. Indices 88, 94, 170, 171, 174, 186, 215, and 230 all possess large negative biases, with only index 6 possessing a large positive bias. This is distinct from all other results observed for the 4M, or indeed any of the tested RNG implementations.

The 4M performs more poorly than the 16M, which is unsurprising given the dramatically inferior performance it achieves in the chi-square test. The predominance of low-index biases in many samples may indicate a systemic issue, but as of yet, there is insufficient proof of this to state so with confidence. Also, the biases migrate significantly between samples, there is no consistent bias



(a) 4M 1 (b) 4M 2

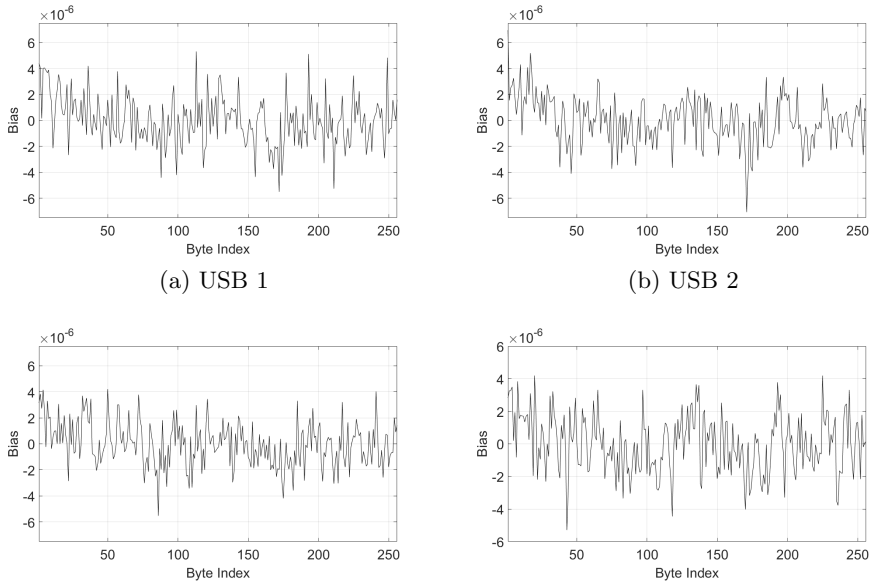
(c) 4M 3 (d) 4M Windows

Fig. 9: Bias of Quantis 4M PCIE Module

observed between samples, but large file sizes do cause this generator to fail a very simple statistical test.

Figure 10 presents the results of the Quantis 4M USB module. Graph (a) exhibits the biased low index values seen in most of the previous Quantis samples. The maximum bias is observed at index 172, with a magnitude of $-5.49 \cdot 10^{-6}$. Indices 1, 2, 4, 5, 36, 88, 99, 113, 155, 172, 174, 193, 211, and 249 all fall above or below the $\pm 4 \cdot 10^{-6}$ threshold. This is a greater number of biases than that seen in Figure 9 (a). However, this sample performs better on the chi-square test, though only barely. As with Figure 9 (c), it appears that a more extreme difference between the average and greatest bias values leads to a far more differentiated expression of the bias.

Graph (b) exhibits a slant towards negative bias values, similar to that seen in Figure 9 (d). The largest bias is found at index 171, with a magnitude of $-7.07 \cdot 10^{-6}$. Other biases exceeding the threshold set for this experiment can be found at 1, 10, 15, 17, 46, and 171. This is substantially fewer than that reported in (a), which may be linked to the lower chi-square score. The trend of (b) follows several curves, wherein indices report consistently positive or negative biases. These curves appear to have no relationship, being of varying period and magnitude. However, they do increase the distribution of bias values and reduce the number of extreme biases.



(a) USB 1 (b) USB 2

(c) USB 3 (d) USB Windows

Fig. 10: Bias of Quantis USB Module

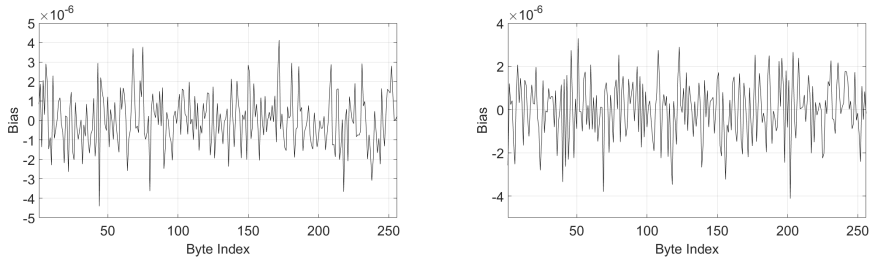
Graph (c) returns to the more even distribution. The largest bias is $-5.52 \cdot 10^{-6}$, at index 86. Biases beyond the noted threshold are found at indices 4, 50, 86, 175, and 241.

Graph (d) shows the results for the Windows sample for the Quantis USB module. This sample has the second best performance on the chi-square test. Its largest bias is $-5.29 \cdot 10^{-6}$, at index 43. At values 20, 42, 118, 170 and 225, the threshold of $\pm 4 \cdot 10^{-6}$ is exceeded. This is a greater number than observed in (c), but less than (a) or (b). This reinforces the observation that this is directly tied to the chi-square score.

An important observation to make is that despite its extremely poor performance in the NIST and Dieharder tests, the 16M is not that much different from the Ubuntu 16M samples. It is also not that different from the other Quantis modules, in terms of bias occurrences and magnitudes. This suggests that the issues identified by Dieharder and STS 2.1.2 are unrelated to the chi-square results, or at least not directly related. Extremely poorly performing 4M modules in terms of chi-square score still pass both the Dieharder and STS batteries, whereas the Windows sample from the 16M does not.

To allow a comparison with two better generators, Figure 11 shows the Chaos Key and *urandom* results. These two generators performed well on the Dieharder and STS batteries, especially *urandom* which passed all tests by a good margin. Graph (a) demonstrates the results for the Chaos Key TRNG. Its largest bias is

$-4.41 \cdot 10^{-6}$, at index 44. Indices 44, and 172 possess biases greater than $\pm 4 \cdot 10^{-6}$. This is an extremely low number of biases. At $\pm 3 \cdot 10^{-6}$ this increases to 7 indices with biases greater than the new threshold value. This is still less than many of the Quantis devices; demonstrating that the Chaos Key is far more evenly distributed in terms of its byte values than the Quantis devices.



(a) Chaos Key (b) *urandom*

Fig. 11: Bias of Chaos Key and *urandom*

Graph (b) shows the results for *urandom*. This PRNG has the lowest upper and lower bias values of all the samples collected for this research. Its largest bias is $-4.11 \cdot 10^{-6}$, and this is the only value that exceeds the threshold of $\pm 4 \cdot 10^{-6}$. Six indices fall outside a threshold of $\pm 3 \cdot 10^{-6}$, making it less biased than the Chaos Key. Both of these RNGs perform better in all respects than the Quantis devices, but *urandom* performs the best of all devices selected for this research. Considering the costs, speeds and reliability issues involved in selecting an appropriate random number generator, this goes to show that money cannot buy everything.

4 Conclusion

This paper has presented some preliminary findings relating to the Quantis family of quantum random number generation modules. These are commercially available QRNGS, certified by the Swiss National Laboratory and Swiss Federal Institute of Metrology (METAS). Delivered in PCI-E or USB packages, these modules use quantum phenomena to provide entropy for random number generation. They are costly devices, ranging from €990 to €2990, and are suggested as being suitable for cryptography, lotteries and similar tasks.

This research finds that although these QRNG modules pass Dieharder and NIST STS 2.1.2 tests, for the most part, they fail the chi-square test of the ENT suite by a significant degree. This is an indicator of byte-level biases in tested files. In the interests of responsible disclosure, we have contacted Quantis with our findings.

METAS has awarded the Quantis devices a certification based on applying the Diehard tests over 10 data sets of 100MB for each tested device [10], as they pass for 2GB datasets with a good degree of consistency between results. However, such a certification only really certifies that the QRNG modules pass the Diehard tests. *urandom* and Chaos Key both pass these tests under similar conditions and are significantly less costly. Quantis itself verifies these devices, providing a certificate that states they passed the Compliance Testing Laboratory (CTL) [15] tests. ID Quantis specifically states that the devices tested in this paper are not certified under BSI AIS31. This is made clear on certificates provided with the QRNG modules.

The difference between the biases in each Quantis module sample suggests that the issue is not reproducible in any single form, but instead, a unique set of biases is generated in each sample. This shows that there is a degree of randomness provided by these devices, likely sufficient to appear random when tested under traditionally accepted suites such as FIPS 140-2, NIST STS, and Dieharder. However, a byte-level analysis shows that there is a consistent degree of failure and associated levels of bias, despite the changing form in which said bias presents itself between samples from a single device.

Further analysis will be required to characterize the bias, but this research highlights the fundamental issues found. Using small data samples and a limited selection of tests is, in essence, a form of confirmation bias, wherein a device may be constructed in such a way that it passes the a-priory known target tests but is flawed in many other ways. It is advisable to test innovative random number generators with as many available tests as possible, with as large a sample as technically possible.

4.1 Future Work

Testing will continue, with a wider range of data and statistical test batteries, including the Crush, Alphabits and Rabbit tests from TestU01 applied over large samples, to identify any further issues. An ongoing exploration of the suitability of certain statistical tests to specific classifications of random number generation will also benefit from the continuation of this research. The bias observed in the ENT results will be explored further using a combination of Masking and Avalanche tests to identify any potential underlying pattern that would conform to structured sequences of n-bits. Such testing has enhanced our previous work with the DESFire EV1 [14], and will take priority in our continuing study of hardware random number generators.

Acknowledgements

This work was funded by InnovateUK as part of the authenticatedSelf project, under reference number 102050. This work was partly sponsored by the ICT COST Action IC1403 Cryptacus in the EU Framework Horizon 2020.

This project has received funding from the European Unions Horizon 2020 research and innovation programme, under grant agreement No.700326 (RAMSES project). The authors also want to thank EPSRC for project EP/P011772/1, on the EconoMical, PsycHologicAl and Societal Impact of RanSomware (EMPHASIS), which supported this work.

References

1. Altus Metrum. Chaoskey true random number generator, June 2008.
2. JG Rarity, PCM Owens, and PR Tapster. Quantum random-number generation and key sharing. *Journal of Modern Optics*, 41(12):2435–2444, 1994.
3. André Stefanov, Nicolas Gisin, Olivier Guinnard, Laurent Guinnard, and Hugo Zbinden. Optical quantum random number generator. *Journal of Modern Optics*, 47(4):595–598, 2000.
4. IQ Quantique. *IDQ Random Number Generation.* IQ Quantique, http://www.idquantique.com/random-number-generation/, 2017.
5. ID Quantique. Id quantique white paper - random number generation using quantum physics, April 2010.
6. Lawrence E Bassham III, Andrew L Rukhin, Juan Soto, James R Nechvatal, Miles E Smid, Elaine B Barker, Stefan D Leigh, Mark Levenson, Mark Vangel, David L Banks, et al. Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications. 2010.
7. Robert G Brown, Dirk Eddelbuettel, and David Bauer. Dieharder: A random number test suite. *Open Source software library, under development*, 2013.
8. Bundesamt fur Sichterheit in der Informationstechnik. Evaluation of random number generators version 0.10. Technical report, Bundesamt fur Sichterheit in der Informationstechnik, 2013.
9. Yevgeniy Dodis, Shien Jin Ong, Manoj Prabhakaran, and Amit Sahai. On the (im) possibility of cryptography with imperfect randomness. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 196–205. IEEE, 2004.
10. Damian Twerendol and Philippe Richard. Certificate of conformity no 151-04687, May 2010.
11. National Institute of Standards and Technology. *NIST computer security resource center (CSRC).* Retrieved from: http://csrc.nist.gov/groups/ST/toolkit/rng/index.html 13:53 07/09/2016.
12. John Walker. *Ent. A pseudo-random number sequence testing program.* Retrieved from: https://www.fourmilab.ch/random/ 13:52 07/09/2016.
13. Walter Anderson. *A study of entropyuh.* Retrieved from: https://sites.google.com/site/astudyofentropy/background-information/the-tests 13:30 07/09/2016.
14. Darren Hurley-Smith and Julio Hernandez-Castro. Bias in the mifare desfire ev1 trng. In *Radio Frequency Identification: 12th International Workshop, RFIDsec 2016, Hong Kong, China, November 30-December 2, 2016*. Springer International Publishing, 2016.
15. Compliance Testing Laboratory. Certificate of compliance, March 2011.