Quantum Random Number Generators

Muhammet Ali Can, Fikret Hacızade, Atilla Hasekioğlu, Thomas Pedersen, and Mustafa Toyran
TÜBİTAK, BİLGEM

***Abstract*—The aim of this position paper is to argue that quantum random number generators are the appropriate choice for critical cryptographic systems. We also discuss some important issues in the design of quantum random number generators.**

***Index Terms*—Quantum cryptography, random number generators, random extractors**

I. Introduction

The ability to make random choices is fundamental in many fields of science and engineering. In cryptography, random bits are used to create cryptographic keys and nonces. In some numerical methods, such as Monte-Carlo algorithms, randomness is used. In simulation, random numbers are used to imitate stochastic processes from the simulated system. In this paper we give a brief overview of some of the techniques used to create and extract randomness.

Random number generators are divided into three categories: pseudo-random number generators (PRNGs), "true" random number generators (TRNGs), and quantum random number generators (QRNGs). PRNGs are deterministic processes whose output is difficult to predict. Strictly speaking it is thus wrong to say that they generate random numbers. However, in many applications using unpredictable numbers suffice. In contrast, TRNGs extract randomness from unpredictable noise of a physical system (stochastic processes). QRNGs are special cases of TRNGs where the source of randomness is intrinsic randomness from a quantum system. Whereas noise can, at least in principle, be predicted or estimated, quantum randomness is intrinsic, and can never be predicted. In this paper, we will always use TRNG to denote hardware random number generators based on noise sources. In Section II, we discuss some details about commonly used sources of randomness, and give a brief discussion of some of the drawbacks and benefits of each. In Section III, we go into more details with different approaches to create randomness from quantum phenomena, which, as we argue, provides the most suitable random number generators for critical security systems, and in Section VI we compare some of the state-of-the-art techniques from all three categories of random number generators.

The typical requirements of a random number generator is that the output be independent and uniformly distributed bits. However, when using a physical system as the source of randomness, the generated randomness may not be uniform. The well-studied field of *randomness extractors*[1] addresses this issues. The task of randomness extractors, first explicitly defined by Nisan and Zuckerman in [18], is to extract independent uniformly distributed bits from an imperfect (non-uniform, low entropy) source of randomness. Randomness extractors are commonly used in "true" and quantum random number generators, where the source of randomness may not always be uniform. We give a brief overview of randomness extraction in Section IV.

When using a random number generator in a critical security system, where unpredictability of the generated number is essential, it is also important to consider the robustness of the generator against attacks. The simplest form of attack is to attempt to predict the future output of the generator by observing past outputs. This is particularly effective against PRNGs, where future output only depend on the current internal state (which is correlated with past outputs). For "true" and quantum random number generators another concern is the tamper resistance of the generator. If not designed properly, the output of these generators can be affected by external manipulation such as heat, power surges, or radiation.

II. Sources of Randomness

A. Pseudo RNG

PRNGs are *deterministic* algorithms which produce a sequence of bits which are, in practice, hard to predict when given only a limited number of previously generated bits.

PRNGs consist of an internal state and a (deterministic) generator function which maps the internal state to a short (typically 32 or 64 bits) sequence of random looking bits, and a new state. Initially the internal state is set to a, supposedly, random value — the so called *seed* of the generator. By successively applying the generator function, an arbitrarily long sequence of bits can then be generated. Since the internal state has a finite size, it is clear that any PRNG will, after some time, repeat itself. The number of bits which are generated before the generator returns to its original internal state is called the *period* of the generator.

The output of a good PRNG will have a lot of the same statistical properties as a sequence of bits chosen independently at random. As a minimum, the frequency of zero and one must be close to $1/2$. Standardized tests (such as [6], [16], [4]) are commonly used to verify bit-frequency and many other statistical properties which let us believe that the sequence shares as many statistical features as possible with a uniformly distributed sequence of bits. However, it is important to remember that any PRNG generates a *deterministic* sequence of bits. Anyone who learns the internal state of a PRNG can predict any future output of that generator.

The major advantages of PRNGs is that they can be implemented in software, and that the bit generation rate is very high.

[1] In some papers the term noise widening is used synonymously to random extractor.

B. True RNG

TRNGs avoid the main problems of PRNGs: the finite period and deterministic behavior. TRNGs extract randomness from a source of noise. The noise source is some physical phenomena which is hard to predict, such as thermal noise.

TRNGs have some disadvantages. If the noise source is sampled faster than the noise frequency, the measurement outputs, and thus the random bits, will be correlated. TRNGs are, therefore, typically limited in speed.

Strictly speaking, TRNGs cannot be said to be truly random. We can mainly divide TRNGs into two categories, depending on the source of noise they use: Generators using thermal noise, and generators using chaotic systems. The "randomness" of chaotic systems comes from the impracticality of determining the exact state of the system. Any error in estimating the state of a chaotic system will make it difficult to predict the evolution of that system. Even thermal noise is, at least in principle, deterministic. The "randomness" in thermal noise comes from the statistical approximation of a large, complicated, but essentially deterministic, physical system. Even though TRNGs are not random in the real sense, it is, however, very unlikely that anyone can predict a well-designed TRNG. It is a matter of choice whether one will accept TRNGs as providing sufficiently trustworthy randomness.

The noise source used in a TRNG may not produce uniformly distributed bits, and may, indeed, have a bias. The bias may even change over time depending on influences from the surrounding environment, such as applied voltage, temperature, and radiation. To prevent this bias from influencing the quality of the output, a random extraction algorithm is applied. The random extraction algorithm takes a number of "low-quality" random bits as input, and produces a smaller amount of "good-quality" random bits as output.

C. Quantum RNG

Quantum mechanics states that some physical phenomena are intrinsically random. There is no way — even in principle — to predict, for instance, the exact decaying time of a radioactive nuclei. Furthermore, these phenomena are not sensitive to changes in the environment. This makes quantum randomness ideal for random number generation.

A typical example of a QRNG can be seen in Fig. 1. A source of laser light (LD) emits photons, which are shot at a semi-transparent mirror (BS). A photon which is reflected by the mirror will be detected by a single-photon detector (The APD labeled 0), while a photon which is transmitted through the mirror will be detected by the other detector (The APD labeled 1). Whether a photon is reflected or transmitted is, by the laws of quantum mechanics, a random process — it can never be predicted. The probability of reflection, p, is not very sensitive to changes in the environment, so if the digital logic (DL) of the generator outputs 0 when one detector detects a photon, and 1 when the other detector detects a photon we are guaranteed that the result is a Bernoulli process with parameter p (under the assumption that the detectors are perfect).

Just as for TRNGs, QRNGs will need a random extractor to generate uniformly distributed bits from the output distribution of the process (Bernoulli in the above example).

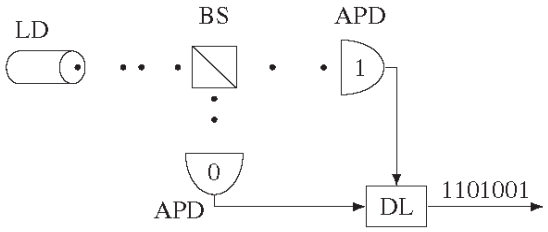

Fig. 1. A simple quantum random number generator

While the underlying randomness is unquestionable in QRNGs, the detection system is subject to influences from noise from the environment. However, where TRNGs uniquely rely on thermal and other sources of noise, QRNGs can be designed to avoid noise as much as possible.

III. Quantum Random Number Generators

A great advantage of basing random number generation on quantum mechanics is that quantum mechanics is intrinsically random. Quantum mechanics guarantees that the outcome of measurements of certain quantum mechanical phenomena are random by nature. No improved measurement apparatus or refined knowledge of the underlying physical system will allow an observer to predict the outcomes of the next measurement. This is in contrast to TRNGs which rely on noise arising from imperfect measurements and/or limited knowledge about the physical system.

One of the first methods proposed for QRNGs was based on radioactive decay. The main disadvantage of this system is that it is slow.

Most current QRNGs are based on the quantum behavior of photons. There are basically two properties of photons which can be used to extract randomness: the uncertainty in the emission time of photons, and the uncertainty in the location of a photon. Fig. 1 is an example of the latter, where the randomness comes from the uncertainty about whether the photon is transmitted or reflected by the semi-transparent mirror. The photon generation of a laser source is Poisson distributed — or, equivalently, the time between generation of two successive photons is exponentially distributed. This uncertainty of photon generation is commonly used in high-speed QRNGs.

The first widely available commercial QRNG was Quantis by ID Quantique[27]. Quantis is based on the principle described in Fig. 1, and has an output bit-rate of 4 Mbps. The Quantis generator also has a more cost-effective 16 Mbps version, where four QRNG components are placed on the same board.

One drawback of using location uncertainty as the source of randomness is that the spatial resolution is directly proportional to the number of detection elements used in the system. This limits the number of output bits per detection by a detection element, thus increasing the overall cost of the generator. When using uncertainty in time, on the other hand, the amount of randomness per detection is only limited by the accuracy and speed of the detection timing. Typically systems based in time uncertainty use only one detection element.

In [25] a QRNG based on the randomness of photon generation time is presented. The QRNG has an output rate of more than 110 Mbps. In order to extract more than one bit of randomness from each detected photon, time is split into equal-size buckets, and the number of the bucket (in binary) in which a photon arrives is used as random sample. The probability of a photon occurring in each bucket is made almost uniform by changing the applied voltage to the laser in such a way that the exponentially distributed generation times become almost uniform up to a certain time interval. A random extractor (SHA256) is applied to the random sample to improve the quality of the randomness.

Another QRNG based on generation time uncertainty is presented in [9]. As in [25], the authors propose a sampling method which guarantees almost uniform bits, thus avoiding any post-processing. Their prototype reaches an output rate of 50 Mbps, and is the basis of the commercial quRNG QRNG[28].

In both location and time based QRNGs single photon detectors are used. The speed of these detectors are the bottleneck for these systems. The commercially available APD single photon detectors typically have a time resolution in the order of tenths of nano seconds[12], [10].

A recent QRNG which is neither based on location nor time uncertainty is presented in [26]. The source of randomness in their proposal is the phase noise of laser light. A benefit of this approach is that phase noise is not measured by slow single photon detectors, but by fast continuous detectors. The speed of their system is limited by electrical noise induced in the detectors which becomes apparent when sampling the noise too fast. The raw output of their randomness source is 1 Gbps, however, due to inefficient randomness extraction, the final output rate of their prototype is 500 Mbps.

Even though the source of randomness in QRNGs is guaranteed by quantum mechanics, any system build to harvest that randomness will be subject to noise which is non-quantum in nature. The number of photons generated per second depends, to some extend, on the voltage applied to the laser. And the efficiency by which a detector detects individual photons is affected by both voltage and heat. In a carefully designed system, however, it is possible to make the effects of noise very small.

To distinguish quantum randomness from noise, so called certifiable QRNGs have been proposed[19]. In quantum mechanics stochastic processes can show correlations which are not possible in classical (non-quantum) physics[3], [8]. This phenomena can be used to certify that a stochastic process is indeed quantum. This provides the user of the QRNG with a guarantee of the genuine randomness of the system.

IV. Randomness Extraction

In TRNGs and QRNGs we often have sources of randomness which are not uniformly distributed, but still have some amount of randomness. The process of turning this "low quality randomness" into "good quality randomness" is the topic of randomness extraction.

The first random extraction algorithm was proposed by von Neumann[24], and provides a good example: Suppose that we have a sequence of independent Bernoulli trials with identical parameter p. To turn this into a sequence of independent uniform bits, we can pair the input bits and output 0 if the input is 01, and 1 if the input is 10. If the input is either 00 or 11 we do not produce any output, but continue to the next input pair. Since the probability for the events 01 and 10 are both equal to $p(1-p)$ the output will be 0 and 1 with identical probability.

The problem with von Neumann's extractor is that the output sequence is less than half the size of the input sequence. Modern extractors perform much better than that.

It is easy to prove that the largest number of uniformly distributed bits that can be extracted from a stochastic variable, X, equals the min-entropy of X (See for instance [11]). The min-entropy of an n-bit binary random variable is defined as

$$H_\infty(X) = \min_{x \in \{0,1\}^n} -\log_2(\Pr[X = x]). \quad (1)$$

On the other hand, it has also be proven that *no deterministic extractor* can extract even one bit of uniform randomness from a stochastic variable of which only the min entropy is known[22]. Any randomness extractor must be randomized, which seems to defy the purpose. The good news is that the extractor can use a relatively small amount of uniform randomness to extract a large amount of uniform randomness from a stochastic variable with high min-entropy.

A random extractor is thus a function, $E : \{0,1\}^n \times \{0,1\}^d \rightarrow \{0,1\}^m$, which takes a sample from an n-bit stochastic variable, X, with min entropy at least k, and d uniformly random bits, such that the variational distance:

$$\|E(X, U_d) - U_m\| < \epsilon, \quad (2)$$

for an appropriately small ϵ, where U_m is the uniform distribution over d-bit strings[18]. We want m to be as large as possible, and d to be as small as possible.

An obvious problem with randomness extractors is to get the bits for the uniformly distributed seed. Fortunately, there are constructions which will convert an extractor, which uses uniform seed, into an extractor which uses another min-entropy source as seed. The only requirement is that the two weak randomness sources should be independent of each other[2]. While independence may be difficult to guarantee in practice, it provides a good heuristic method.

A common approach used in the literature of "true" and quantum random number generators for extracting randomness is to apply a cryptographic hash function to the output of the physical generator. Besides being overly complex, this approach has only recently been proven to guarantee that the output is uniformly random[7]. Much simpler and more efficient random extractors exists. Currently, the best random extractor for many scenarios is Trevisans extractor[23] which can extract almost $H_\infty(X)$ bits by using a random seed which is only poly-logarithmic in n. There are several surveys going into more depth with randomness extractors, such as [15], [17], [21].

One side-effect of using a randomness extractor, is that it becomes very difficult to find systematic errors in the hardware by applying statistical tests to the output of the random number generator. The randomness extractor effectively

masks systematic errors. A common way to overcome this problem used, for instance, in [25], [9] is to try to make the output of the randomness sampling as close to uniform as possible, and then apply statistical tests to the output of the randomness sampling. In those cases randomness extractors are either entirely omitted, or only used to make give extra robust. However, this approach may not always allow for the highest possible throughput: to maximize throughput one should maximize min-entropy per second generated by the source and not the number of bits close to uniform. It is important to realize that the output of the randomness sampling does not need to be close to uniform in order to be useful for a random generator.

V. Verifying Randomness

Any proposed random number generator must be tested to verify that the output looks random. There are a wide range of standardized tests available [16], [4], [6], etc. Essentially a randomness test is a function which takes n bits as input, and produces some (real) value as output. The test is said to be passed if the probability that n truly uniform random bits would give a value at least as extreme (deviating from the expected value by at least the same amount) is below a given threshold. See [16] for a more elaborate explanation of the design of randomness tests.

A very important point about randomness tests is that *no test can verify randomness*. The only thing a randomness can do is to reject the hypothesis that a sequence of bits comes from a uniformly random process. Passing a randomness tests gives no conclusion about the quality of the randomness. Randomness test should therefore only be used to detect problems with an RNG design. The tests cannot be used as a qualitative measure of "how random" the output of a generator is.

Randomness extraction, most often used by "true" and quantum random number generators, have the ability to make most binary sequences look random to a statistical test — in some sense they are, themselves, pseudo random number generators. This means that testing the final output of a TRNG or QRNG which applies a good randomness extractor is very unlikely to reveal problems with the generator. Instead of analyzing the final output of the generator, the output of the sampling method should be tested. If the sampling output is expected to be uniform, this is easy, and indeed this approach is used in many designs. However, in some designs the sampling method does not give uniform output. In these cases specialized tests needs to be performed.

VI. State-of-the-art Random Number Generators

The Mersenne twister PRNG[13] is one of the most popular PRNGs today, preferred for it's efficient implementation and very large period of $2^{19937} - 1$ bits. The Mersenne twister, however, is not considered secure for cryptographic usage, since the stream can be predicted after observing a small amount of output[13]. To amend this, a modified version of the Mersenne twister has been proposed[14]. In cryptography, however, a general construction introduced by Blum, Blum, and Shup[1] is often preferred. It has been proven impossible to distinguish the output of the BBS PRNG from a uniform distribution in polynomial time under the assumption that the quadratic residuosity problem is difficult.

TABLE I
List of some state-of-the-art random number generators.

	Generator	Throughput	Price
PRNG	Mersenne Twister	–	$0
	Blum Blum Shup	–	$0
TRNG	Protego SG100[20]	0.128 Mbps	$325
	Comscire R2000KU[5]	2 Mbps	$895
QRNG	Quantis[27]	4 Mbps	$1500
	QRBG121	12 Mbps	?
	quRNG[9]	50 Mbps	?

The Comscire R2000KU TRNG uses a combination of thermal and transistor noise to generate randomness at a rate of 2 Mbps.

We have already mentioned the state-of-the-art QRNGs in the previous section. Here it suffices to emphasize that during the last couple of years we have seen a surge in the speed of QRNGs.

Table I shows a few of the most widely available random number generators. We have not included the fastest QRNGs, since they are not commercially available yet. The PRNGs are freely available, so no cost is involved in their use. The throughput of PRNGs depends on the implementation and on the underlying hardware, so throughput is not listed. In general, though, PRNGs are much faster than both QRNGs and TRNGs.

VII. Conclusion

In this paper, we have presented evidence to support our claim that quantum random number generators should be the generator of choice for critical security systems. We have also pointed out some of the techniques that we believe are the most promising when creating QRNGs: We believe that QRNGs based on time uncertainty are likely to provide the most cost effective and fast systems. We have also pointed out the importance of choosing the appropriate randomness extraction method[2]. Finally, we gave a warning about some pitfalls when verifying true and quantum random number generators: Some researchers design the randomness sampling such that it passes the statistical tests, wheres the randomness sampling should aim at maximizing the min-entropy output per second. Secondly, we mentioned that passing a statistical test does *not* guarantee good randomness. Statistical tests can only be used to detect poor quality randomness.

References

[1] L. Blum, M. Blum, M. Shub, *A simple unpredictable pseudo random number generator*, In SIAM J. Comput., 15(2):364–383 May 1986.
[2] B. Barak, R. Impagliazzo, and A. Wigderson, *Extracting randomness using few independent sources*, In Proc. IEEE 45th Annu. IEEE Symp. Foundations of Computer Science (FOCS04), 2004, pp. 384–393.
[3] J. Bell, *On the Einstein Podolsky Rosen Paradox*, In Physics 1 (3):195–200, 1964.

[2]Currently we recommend using Trevisans extractor[23] for most cases.

[4] *Ais 20: Functionality classes and evaluation methodology for deterministic random number generators, v2.0, bsi*, https://www.bsi.bund.de/cae/servlet/contentblob/478150/publicationFile/30276/ais20 pdf, 1999.
[5] Comscire, http://comscire.com
[6] http://www.stat.fsu.edu/pub/diehard/
[7] Y. Dodis, R. Gennaro, J. Håstad, H. Krawczyk, and T. Rabin, *Randomness Extraction and Key Derivation Using the CBC, Cascade and HMAC Modes*, In Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference, 494–510, 2004.
[8] A. Einstein, B. Podolsky, N. Rosen, *Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?*, Physical Review 47(10):777, 1935.
[9] M. Furst, H, Weier, S. Nauerth, D. G. Marangon, C. Kurtsiefer, and H. Weinfurter, *High speed optical quantum random number generation*, In Optics Express, 18(12), June 2010.
[10] Hamamatsu APDs: http://sales.hamamatsu.com/index.php?id=13166473
[11] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby, *A Pseudorandom Generator from any One-way Function*, SIAM Journal on Computing, 28(4): 1364–1396, 1999.
[12] ID100 series of APDs from ID Quantique: http://www.idquantique.com/scientific-instrumentation/id100-silicon-apd-single-photon-detector.html
[13] M. Matsumoto and T. Nishimura, *Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator*, In ACM Trans. Model. Comput. Simul., 8(1):3–30, January 1998.
[14] M. Matsumoto, T. Nishimura, M. Hagita, and M. Saito, *Cryptographic Mersenne Twister and Fubuki Stream/Block Cipher*, http://eprint.iacr.org/2005/165.
[15] N. Nisan, *Extracting randomness: How and why: A survey*, In Proceedings of the Eleventh Annual IEEE Conference on Computational Complexity, pages 44–58, Philadelphia, Pennsylvania, 24–27 May 1996. IEEE Computer Society Press.
[16] National Institute of Standards and Technology, *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, Available online: http://csrc.nist.gov/publications/nistpubs/800-22-rev1/SP800-22rev1.pdf, April 2009.
[17] N. Nisan and A. Ta-Shma, *Extracting randomness: A survey and new constructions*, JCSS: Journal of Computer and System Sciences, 58, 1999.
[18] N. Nisan and D. Zuckerman, *Randomness is linear in space*, in Journal of Computer and System Sciences, 52(1):43–52, February 1996.
[19] S. Pironio, A. Acín, S. Massar, A. Boyer De La Giroday, D. N. Matsukevich, P. Maunz, S. Olmschenk, D. Hayes, L. Luo, T. A. Manning, and C. Monroe, *Random numbers certified by Bell's theorem.*, In Nature, 464(7291):10, 2010.
[20] Protego, http://protego.se
[21] R. Shaltiel, *Recent developments in explicit constructions of extractors*, Bulletin of the EATCS 77:67–95, 2002.
[22] M. Santha and U. V. Vazirani, *Generating quasi-random sequences from semi-random sources*, Journal of Computer and System Sciences, 33:75–87, 1986.
[23] L. Trevisan, *Construction of extractors using pseudorandom generators.* In Proceedings of the 31st ACM Symposium on Theory of Computing, 1999.
[24] J. von Neumann, *Various techniques used in connection with random digits*, In Applied Math Series, 12:36–38, 1951.
[25] M. A. Wayne and P. G. Kwiat, *Low-bias high-speed quantum random number generator via shaped optical pulses*, In Optics Express, 18(8), 12 April 2010.
[26] B. Qi, Y-M. Chi, H-K. Lo, and L. Qian, *High-speed quantum random number generation by measuring phase noise of a single-mode laser*, In Optical letters, 35(3), February 2010.
[27] The Quantis QRNG by ID Quantique. More information available at http://www.idquantique.com/images/stories/PDF/quantis-random-generator/quantis-whitepaper.pdf.
[28] quRNG by quTools, http://www.qutools.com/products/quRNG/