FROM VISION TO TECHNOLOGY

REDEFINING RANDOMNESS

QUANTIS

WHEN RANDOM NUMBERS CANNOT BE LEFT TO CHANCE

# ID Quantique

# Technical Paper on

# Randomness Extractor

Version 1.0
September 2012

ID Quantique SA
Chemin de la Marbrerie 3
1227 Carouge/Geneva
Switzerland
T +41 22 301 83 71
F +41 22 301 83 79
info@idquantique.com
www.idquantique.com

# A randomness extractor for the Quantis device

M. Troyer and R. Renner[*]

September 19, 2012

## Abstract

This document describes the use of a randomness extractor to post-process the raw random bits generated by the Quantis random number generator. In particular, we detail the implementation of the extractor function as well as the choice of the necessary parameters. The latter depend on the quality of the raw random bits, which is quantified in terms of the entropy per bit. We propose a method to obtain an estimate for this entropy.

## 1 The extractor

### 1.1 Extractor design

The basic idea of the extractor is to compute $k$ output bits $y_i$ with high randomness from $n > k$ input bits $x_j$ with less randomness. Assuming that each bit of the input sequence has entropy $s$ ($s = 1$ for perfect randomness), the probability that the extractor output will deviate from a perfectly uniform $k$-bit string will be strictly bounded by

$$\varepsilon_{hash} = 2^{-(sn-k)/2} \tag{1}$$

(see Theorem 1 of Appendix A.4). The value $\varepsilon_{hash}$ can therefore be seen as a measure for the quality of the random bits output by the extractor. As outlined in Appendix A.1, the quantity $\varepsilon_{hash}$ has a direct operational interpretation: it can be seen as the maximum probability by which a deviation from a perfectly random sequence can be noticed. While $\varepsilon_{hash}$ can be made arbitrarily small, a value of $\varepsilon_{hash} = 0$ is generally unachievable. Our aim is to keep $\varepsilon_{hash}$ below $2^{-100}$ or $10^{-30}$, implying that even using millions of devices one will not see any deviation from perfect uniform randomness in a time longer than the age of the universe. The value of $\varepsilon_{hash}$ could even be lowered further (e.g., by reducing the size of $k$, see below) if needed.

The extractor is a simple bit-matrix-vector multiplication with a random matrix $m$ (also called *seed*)

$$y_i = \sum_{j=1}^{n} m_{ij} x_j \tag{2}$$

performed modulo 2. Multiplication of bits is easiest implemented by a bitwise AND operation, additions by a bitwise exclusive OR, and sums by efficient bit count algorithms.

### 1.2 Choice of parameters $n$ and $k$

When choosing $n$ and $k$ one has to keep in mind two conflicting requirements:

- Keeping the bound for failure probability $\varepsilon_{hash}$ fixed, the efficiency (in terms of the number of generated bits per input bit), $k/n = s - 2\log_2(1/\varepsilon_{hash})/n$, can be raised by increasing $n$.
- The computational complexity for the matrix-vector multiplication is $O(nk)$, and hence $O(n)$ operations need to be performed per generated random bit.

After study of different values for $n$, we find that, $n = 1024$ or $n = 2048$ are good choices for the Quantis and with the implementation of the randomness extractor in Quantis software package. These values gives a good balance between extractor efficiency and computational power requirement. Using $n = 1024$ and $k = 768$ one loses 25% of the bits, while with $n = 2048$ and $k = 1792$ one loses only 12.5%. Either is much better than the 75% loss using von Neumann de-biasing, which

[*]This document has been edited by IDQ to reflect the settings actually implemented in the Quantis software package.

additionally does not eliminate correlations. Thus, from each block of 1024 or 2048 bits the extractor generates 768 or 1792 bits of provably good random numbers.

To keep $\varepsilon_{hash} < 2^{-100}$ one should choose $k < sn - 200$. The above choices are thus acceptable as long as for a given device the entropy per raw bit satisfies $s > 0.946$ for $n = 1024$ or $s > 0.973$ for $n = 2048$.[1] The entropy of raw sequence generated by Quantis is guaranteed to be always above this value, thanks to continuous monitoring of the performance of the device.

Others values for $n$ and $k$ can be chosen, but we must use multiples of 64 for $k$ for performance reasons.

### 1.3 Generating the random matrix

The quality of the random bit matrix $m$ is crucial for the extractor to work. Fortunately, it needs to be determined only once and can be reused for all devices. To simplify the substitution of the random bit matrix by the user, it is hard-coded in a file delivered within the Quantis software package. For the two choices above we need 768 kbits or 3584 kbits respectively.

To generate these random bits in practice, a safe (but costly) method is to generate each of the bits by adding (modulo 2) $r$ (weakly) random bits from $r$ mutually independent sources. According to Lemma 4 in Appendix A.3, the resulting bits can be made arbitrarily close to uniform by choosing $r$ sufficiently large. (The distance from perfect uniform bits is exponentially small in $r$.)

Ideally, the $r$ individual sources used in the described procedure to generate the matrix $m$ should be obtained from different sources. In addition, we recommend to take the following precautions:

- take only bits separated by times much longer than the autocorrelation time (we recommend a distance of 100 to 1000 between bits);
- use von Neumann de-biasing.

[1]The required bound on the entropy $s$ is slightly larger for the case $n = 2048$ because of the smaller loss of only 12.5%. If one is ready to tolerate a larger loss (25%), one may choose $k = 1536$, in which case it is sufficient to have $s > 0.848$.

In the Quantis software package, the random bit matrix has been generated from multiple Quantis devices.

## 2 Estimating the entropy

As already indicated above, the quality $\varepsilon_{hash}$ of the final bits depends on the entropy $s$ per bit of the raw randomness generated by the device. In general, the relevant entropy measure in this context is the min-entropy $H_{\min}$. The min-entropy is closely related to the collision entropy $H_2$ (see Lemma 3 in Appendix A.2 for details). For the particular extractors we use here (based on two-universal hashing), slightly tighter bounds can be obtained by direct use of the collision entropy. (For other extractor constructions, e.g., Trevisan's extractor [Tre01], it is however necessary to use the min-entropy.)

In this section, a quick lower bound that can be used to obtain a rough estimate for the entropy is given. This bound depends on the bias of the individual bits only. We will then discuss a more elaborate method that can be used to get a better (higher) bound by running additional tests on a device. Besides the generally better bound, a main advantage of this method is that it relies on weaker assumptions, as discussed in Appendix A.5.

### 2.1 Rough bound for the min-entropy based on the bias

First a quick way to estimate a lower bound for $H_2$ is given, incorporating both the bias and autocorrelations of an imperfect source of random bits $X_i$. The bias $b$ measures the deviation of the distribution of each bit $X_i$ from uniform, i.e., the values 0 and 1 are taken with probability $1/2 \pm b$. In the following we assume that the bias $b$ has been measured with a statistical error $\delta$.[2] Then, to be on the safe side we choose the probabilities to be $p_0 = 1/2 - b - 9\delta - 0.001$ and $p_1 = 1/2 + b + 9\delta + 0.001$ The term $9\delta$ makes sure that the bias estimate has an error of less than $10^{-30}$, and the term 0.001 is a generous bound on the correlations. With these we can estimate the entropy $s = H_2(X_i)$ of each bit

[2]If the bias is determined from $N$ measurements, then the error is $\delta \approx 1/(2\sqrt{N})$.

produced by the source as

$$\begin{aligned} s &\gtrsim -\log_2(p_0^2 + p_1^2) \\ &= -\log_2(1/2 + 2(b + 9\delta + 0.001)^2). \end{aligned}$$

## 2.2 Accurate determination of the min-entropy

To determine the min-entropy we have to consider the asymptotic entropy gain when adding another bit to a bit string $C = (X_1, \dots, X_m)$ of $m$ bits, and thus look at the difference of the collision entropy of bit strings of lengths $m$ and $m + 1$:[3]

$$s_m := H_2(X_1, \dots X_{m+1}) - H_2(X_1, \dots, X_m) \quad (3)$$

for large values of $m$, i.e., $s = \lim_{m\to\infty} s_m$.

The collision entropy $H_2(X_1, \dots X_m)$ of bit strings of length $m$ is estimated by recording normalized histograms $h^{(m)}(c)$, counting how often each possible bit string $c = (x_1, \dots, x_m)$ appears. After normalizing so that $\sum_c h^{(m)}(c) = 1$, $H_2(X_1, \dots, X_m)$ is estimated as

$$H_2(X_1, \dots, X_m) \approx -\log_2 \sum_c h^{(m)}(c)^2 \; . \quad (4)$$

To obtain $s$ we thus record histograms $h^{(m)}$ for bit strings of the longest length $m$ to be considered. In our tests we used bit strings of length up to $m = 16$, but found that for the current Quantis device $m = 8$ seems sufficient. Longer $m$ requires exponentially more memory and sampling time. From that histogram, the histograms for shorter bit strings can be obtained by partial sums:

$$h^{(m-1)}((x_1, \dots, x_{m-1})) = \sum_{x_m=0}^{1} h^{(m)}((x_1, \dots, x_m)). \quad (5)$$

One can then calculate and plot $s_m$ as a function of $m$, check for convergence and obtain the limiting value $s$ . We have observed very fast convergence, and for practical purposes it is sufficient to estimate $s$ by $s_m$ for the largest $m$ considered.

To obtain error estimates this procedure should be repeated several times and the mean value $\overline{s}$ and statistical error $\Delta s$ estimated by standard statistical methods. The final estimate for $s$ needs to take into account these statistical errors: the probability that the true entropy is below our estimate needs to be smaller than the bound on the failure probability $\varepsilon_{hash}$ that we tolerate. To achieve this we use

$$s = \overline{s} - \alpha \Delta s, \quad (6)$$

where the value $\alpha$ is chosen such that the complementary error function $\mathrm{erfc}(\alpha) < \varepsilon$.

## 2.3 Calculating autocorrelations

The histogram $h^{(m)}$ can also be used to calculate the autocorrelation function

$$C(t) = \langle x_i x_{i+t} \rangle - \langle x_i \rangle \langle x_{i+t} \rangle \quad (7)$$

for distances $t \le m - 1$. To do so we calculate

$$\begin{aligned} \overline{x_i} &= \sum_{c=(x_1,\dots,x_m)} x_i h^{(m)}(c) \quad (8) \\ C(t) &\approx \sum_c \frac{1}{t} \sum_{i=1}^{m-t} \left[ (x_i x_{i+t} - \overline{x_i}\; \overline{x_{i+t}}) h^{(m)}(c) \right]. \end{aligned}$$

Statistical errors on $C(t)$ are calculated in the standard way, or preferably by a jackknife or bootstrap method to obtain the smallest possible reliable errors.

# A Appendix

## A.1 Statistical distance

**Definition 1.** The *statistical distance* between two probability distributions $P_X$ and $P_{X'}$ with the same alphabet is defined as

$$\delta(P_X, P_{X'}) := \frac{1}{2} \| P_X - P_{X'} \|_1 \; .$$

.

*Proof.* A proof of this statement is given in [RK05] (see Lemma 1). □

The statistical distance has an operational interpretation in terms of probabilities, as asserted by the following lemma.

[3]We use here the collision entropy to simplify the presentation. To derive rigorous statements, it is advantageous to use the smooth collision entropy, as briefly explained in Appendix A.5.

**Lemma 1.** *Let $P_X$ and $P_{X'}$ be two probability distributions with distance $\varepsilon = \delta(P_X, P_{X'})$. Then there exists a joint distribution $P_{XX'}$ such that*

$$\Pr[X \neq X'] \leq \varepsilon \ .$$

In other words, if two probability distributions $X$ and $X'$ are $\varepsilon$-close, we may think of a common random experiment where the values $X$ and $X'$ coincide except with probability $\varepsilon$. one of the practically relevant implications of this is formalized by the following simple lemma. It shows that if an application (e.g., a simulation algorithm or a cryptographic scheme) is proved to work well with perfect random bits $X$, then it can also be used safely with bits $X'$ whose distribution is not exactly uniform.

**Lemma 2.** *Let $\Pi$ be a stochastic process that takes as input a random value $X$ and may fail with a probability $p_{\text{fail}}(Pr(\Pi(X))$. If the input $X$ is replaced by $X'$ then the failure probability can increase by at most $\varepsilon := \delta(P_X, P_{X'})$, i.e.,*

$$p_{\text{fail}}(\Pi(X')) \leq p_{\text{fail}}(\Pi(X)) + \varepsilon \ .$$

*Proof.* We may describe the stochastic process as a function $f$ which takes as input $X$ as well as some additional randomness $R$, and outputs either "fail" or "success". Obviously, the outputs $f(X, R)$ and $f(X', R)$ can only deviate if $X \neq X'$. Since, according to Lemma 1, the probability that this happens is bounded by $\varepsilon$, the assertion follows. □

## A.2 Definitions of entropies and basic properties

**Definition 2.** The *min-entropy* $H_{\min}$ and the *collision entropy* $H_2$ of a random variable $X$ with distribution $P_X$ are defined by

$$H_{\min}(X) = -\log_2 \|P_X\|_\infty$$
$$H_2(X) = -\log_2 \|P_X\|_2^2 \ .$$

**Definition 3.** For any $\mu \geq 0$, the *smooth min-entropy* $H^\mu_{\min}$ and the *smooth collision entropy* $H^\mu_2$ of a random variable $X$ are defined by

$$H^\mu_{\min}(X) = \max_{P_{X'} \in \mathcal{B}^\mu(P_X)} H_{\min}(X')$$
$$H^\mu_2(X) = \max_{P_{X'} \in \mathcal{B}^\mu(P_X)} H_2(X') \ .$$

where $\mathcal{B}^\mu(P_X)$ is the set of all distributions $P_{X'}$ which have at most distance[4] $\mu$ from $P_X$.

The two entropy measures defined above are essentially equivalent. This means that any estimate for the (smooth) min-entropy is a good estimate for the collision entropy, and vice versa.

**Lemma 3.** *For any $\mu \geq 0$ and $\mu' > 0$,*

$$H^\mu_{\min}(X) \leq H^\mu_2(X) \leq H^{\mu+\mu'}_{\min}(X) + \log_2 \frac{1}{\mu'} \ .$$

*Proof.* The first inequality follows from the general inequality $\|P_X\|_\infty \geq \|P_X\|_2^2$ (which holds for normalized $P_X$), and the second is a special case of Lemma I.3 of [RW04]. □

## A.3 XOR of independent sources

The following lemma asserts that good randomness can be obtained by combining bits generated by a number of mutually independent weak random sources. While this method is costly and slow, it may be used for generating the initial random seed (i.e., the matrix $m$ in the construction described in Sections 1.1; see also Section 1.3), which only needs to be determined once.

**Lemma 4.** *Let $X_1, \dots X_r$ be $r$ independent random bits such that, for some $b > 0$,*

$$\delta(X_i, U) \leq b \quad \forall i \ .$$

*Then the bit $X = \sum_i X_i \mod 2$ satisfies*

$$\delta(X, U) \leq \frac{1}{2}(2b)^r \ .$$

*Proof.* A simple calculation shows that, for two independent bits $X_1$ and $X_2$ with $b_i = \delta(X_i, U)$, and with $X = X_1 + X_2 \mod 2$,

$$\delta(X, U) = 2b_1b_2 \ .$$

The claim then follows by recursive application of this rule. □

[4]The recent literature on smooth entropies uses the *purified distance* instead of the *statistical distance*. The purified distance is a more natural choice in the context of quantum information. However, since the results reported here are purely classical, it is more convenient and technically easier to use the statistical distance.

## A.4 Randomness extraction by two-universal hashing

**Definition 4.** A family $\{f_s\}_{s\in\mathcal{S}}$ of functions with codomain $\mathcal{U}$ is called *two-universal* if for any distinct inputs $x \neq x'$

$$\left\langle \delta_{f_s(x),f_s(x')} \right\rangle_s \leq \frac{1}{|\mathcal{U}|},$$

where $\langle\cdot\rangle_s$ denotes the expectation value over $s$ chosen uniformly at random from the set $\mathcal{S}$

In other words, for any fixed pair of distinct inputs, the collision probability for a function chosen at random from a two-universal family is low. An explicit construction of a family of functions with this property is given in Section 1.1, and further constructions can be found in [CW79] and [WC81].

**Theorem 1.** *For any random variable $X$ with alphabet $\mathcal{X}$, for any two-universal family $\{f_s\}_{s\in\mathcal{S}}$ of functions from $\mathcal{X}$ to $\{0,1\}^k$, and for $S$ distributed uniformly (over $\mathcal{S}$) and independently of $X$,*

$$\delta\big(P_{f_S(X)S}, P_U \times P_S\big) \leq \varepsilon := 2^{-\frac{1}{2}(H_2(X)-k)} ,$$

*where $P_U$ denotes the uniform distribution over $\mathcal{U}$.*

*Proof.* First versions of this theorem have been proved in [BBR88], [ILL89], and [BBCM95]. The version stated here (in terms of statistical distance and collision entropy) can be obtained as the classical special case of Theorem 5.5.1 of [Ren05]. □

**Corollary 1.** *Theorem 1 remains valid if $\varepsilon$ is replaced by*

$$\varepsilon := \min_{\mu} 2^{-\frac{1}{2}(H_2^{\mu}(X)-k)} + \mu \quad .$$

*Proof.* For any $\mu$, let $X'$ be a random variable with $\delta(P_X, P_{X'}) \leq \mu$ such that $H_2(X') = H_2^{\mu}(X)$. Then, by Theorem 1,

$$\begin{aligned}\delta\big(P_{f_S(X')S}, P_U \times P_S\big) &\leq 2^{-\frac{1}{2}(H_2(X')-k)} \\ &= 2^{-\frac{1}{2}(H_2^{\mu}(X)-k)} .\end{aligned}$$

The assertion then follows from

$$\delta(P_{f_S(X)S}, P_{f_S(X')S}) \leq \delta(P_X, P_{X'}) \leq \mu$$

and the triangle inequality. □

**Remark 1.** *The distance on the left hand side of Theorem 1 can be rewritten as*

$$\delta\big(P_{f_S(X)S}, P_U \times P_S\big) = \big\langle \delta(P_{f_s(X)}, P_U)\big\rangle_s$$

*where $\langle\cdot\rangle_s$ denotes the expectation value over $s$ chosen uniformly at random from the set $\mathcal{S}$.*

Since these results are expressed using the statistical distance, it is easy to interpret them operationally, according to the discussion of Section A.1. In particular, together with Lemma 1, the above remark implies that (on average over the choices of $s$) there exists a uniformly distributed random variable $U$ such that the function output $f_s(X)$ is identical to $U$ except with probability $\varepsilon$. In other words, $\varepsilon$ is the maximum probability by which $f_s(X)$ deviates from a uniformly chosen value.

## A.5 Technical statement of result and assumptions

Combining the facts outlined above, we obtain the following statement. Consider the extractor specified in Section 1.1, which takes as input an $n$-bit string and outputs a $k$-bit string, and assume that the following conditions are satisfied for appropriately chosen parameters $\varepsilon_{\text{seed}}, \varepsilon_{\text{stat}}, \varepsilon_{\text{hash}} \geq 0$:

- The matrix $m$ used by the constructor is chosen using a sequence of random bits whose joint distribution has distance at most $\varepsilon_{\text{seed}}$ from a perfectly uniform distribution. We refer to Section 1.3 for a discussion of how this can be achieved using a sufficiently large number of mutually independent random number generators.
- The smooth collision entropy of the input bits, $H_2^{\varepsilon_{\text{stat}}}(X_1 \cdots X_n)$, is lower bounded by $sn$, with

  $$s := \frac{1}{n}\big(k + 2\log_2(1/\varepsilon_{\text{hash}})\big) .$$

  This is the case whenever

  $$\min_m s_m \geq s \qquad (9)$$

  holds for

  $$s_m := H_2^{\varepsilon_{\text{stat}}}(X_1, \ldots X_{m+1}) - H_2^{\varepsilon_{\text{stat}}}(X_1, \ldots, X_m) .$$

Condition (9) can be established using statistical methods (cf. Section 2.2), with a reliability bound of $\varepsilon_{\text{stat}}$ for the statistical estimate. (Note that an error probability of $\mu$ in the estimate for the collision entropy $H_2$ corresponds to estimating the smooth collision entropy $H_2^\mu$. This explains why we use this entropy measure here.)

Under these conditions, the probability that the output string deviates from a perfectly uniform $k$-bit string is upper bounded by $\varepsilon = \varepsilon_{\text{seed}} + \varepsilon_{\text{stat}} + \varepsilon_{\text{hash}}$.

## References

[BBCM95] C. H. Bennett, G. Brassard, C. Crépeau, and U. Maurer. Generalized privacy amplification. *IEEE Transaction on Information Theory*, 41(6):1915–1923, 1995.

[BBR88] C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988.

[CW79] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.

[ILL89] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions (extended abstract). In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, pages 12–24, 1989.

[Ren05] R. Renner. *Security of Quantum Key Distribution.* PhD thesis, Swiss Federal Institute of Technology (ETH) Zurich, 2005. Available at arXiv:quant-ph/0512258.

[RK05] R. Renner and R. König. Universally composable privacy amplification against quantum adversaries. In *Second Theory of Cryptography Conference TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 407–425. Springer, 2005.

[RW04] R. Renner and S. Wolf. Smooth Rényi entropy and applications. In *Proc. International Symposium on Information Theory*, page 233. IEEE, 2004.

[Tre01] L. Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48:860–879, 2001.

[WC81] M. N. Wegman and J. L. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22:265–279, 1981.