Canary Numbers:
Design for Light-weight Online Testability of True Random Number Generators

Vladimir Rožić, Bohan Yang, Nele Mentens and Ingrid Verbauwhede

ESAT/COSIC and iMinds, KU Leuven,
Kasteelpark Arenberg 10, 3001 Heverlee-Leuven, Belgium

Abstract. We introduce the concept of canary numbers, to be used in health tests for true random number generators. Health tests are essential components of true random number generators because they are used to detect defects and failures of the entropy source. These tests need to be lightweight, low-latency and highly reliable. The proposed solution uses canary numbers which are an extra output of the entropy source of lower quality. This enables an early-warning attack detection before the output of the generator is compromised. We illustrate the idea with 2 case studies of true random number generators implemented on a Xilinx Spartan-6 FPGA.

1 Introduction

Random number generators (RNGs) are essential components of security systems. They are used for generating session keys, challenges, masks and for padding messages. A failure or improper use of the randomness source can lead to a security failure even when strong cryptography is used [6, 4, 3]. True random number generators (TRNGs) are hardware modules that produce random bits based on the outcome of an unpredictable physical process such as meta-stability or thermal noise. Due to the sensitivity of security applications, TRNGs used in secure systems have to be equipped with testing modules for failure detection during the operation.

1.1 Problem statement

True random number generators (TRNG) can be compromised due to aging, changes in operating conditions or active attacks. Health tests of the noise source are a countermeasure recommended by NIST [1] and BSI [5] standards for TRNG design and evaluation. Important design criteria for health tests are: compact implementation, low latency (i.e. low number of input bits required by the test), high attack detection capability and low false-positive error rate.

1.2 Our contribution

In this paper we propose an improvement of testability by modifying the entropy source architecture. This paper contains the following contributions:

- The concept of canary numbers is introduced. The main idea of this paper is to improve the testability of the noise source by producing two bit streams: raw numbers to be used by the application, and canary numbers to be used solely for the testing purpose. Canary numbers have lower statistical quality than the raw numbers and they are more susceptible to changes in operating conditions. For this reason, monitoring canary numbers can be used for an early-warning failure detection, since the statistical quality of the canary numbers drops before the failure affects the raw numbers in a significant way.
- The proposed concept is illustrated on the example of an elementary ring-oscillator based TRNG
- The second example used to illustrate the proposed concept is the carry-chain based TRNG [7].

1.3 Organization

The rest of the paper is organized as follows. Section 2 introduces the terminology and presents the background on TRNGs and health tests. Section 3 introduces the concept of canary numbers and explains their benefits for developing fast, lightweight and effective tests for randomness. We propose 2 methods for designing health tests based on canary numbers. We illustrate the proposed concept using two case studies. In Section4 we apply our methodology to design health tests for an elementary ring-oscillator based TRNG. The stochastic model of this generator is used to justify the design decisions made for designing the health test based on the canary numbers. A similar procedure is applied to a carry-chain based TRNG [7]. The results are presented in Section 5. Conclusions are presented in Section 6.

2 Terminology, Notation and Background

The model of the entropy source presented in [1] is shown in Figure 1. The essential component of the entropy source is the noise source. This the only component of the TRNG that generates randomness. The noise source can be implemented as a meta-stable element, a ring oscillator, a noisy resistor or any other component whose behavior cannot be reliably modelled and predicted. We denote the design parameters of this component with $n_1, n_2, ...$ These are, for example, the number of stages of a ring oscillator or the resistance of a noisy resistor.

The digitization block is used to convert the output of the noise source into digital data. This block also has its design parameters (for example the sampling frequency) which are denoted with $d_1, d_2, ...$

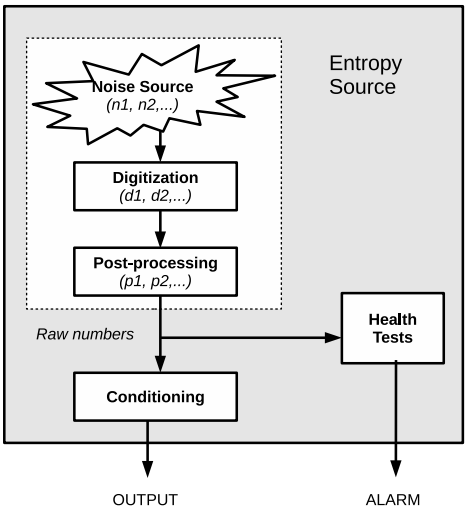


Fig. 1: NIST SP 800-90B model of the entropy source.

Simple post-processing operations (e.g. down-sampling, xor, parity filter) may be used to improve the statistical properties of the generated data. The produced data are referred to as the *raw numbers*. The design parameters of the processing block are denoted with $p_1, p_2, ...$

Raw numbers are usually not statistically perfect, and an optional conditioning block is used to transform the sequence of raw random numbers into a sequence of ideal random bits. Health tests are hardware modules that verify the quality of the produced raw bits. Their purpose is to trigger an alarm when abnormal behavior or attacks are detected. Health tests are often implemented as statistical tests that operate on a sequence of consecutive raw numbers. Two types of errors should be considered when designing statistical tests: false positives when the alarm is signaled but no attack happens, and false negatives when the attack happens but it goes undetected by the tests. The probability of a false positive error is a design parameter and the NIST recommendation is to keep this probability below 2^{-40} because triggering the alarm leads to locking the device, after which some form of manual reset is required.

In every type of statistical testing it is impossible to avoid false positive and false negative errors. However, it is possible to trade one type of error probability for another, i.e. decreasing the false positive error rate and increasing the false negative error rate which decreases the test usefulness. When an extremely low false-positive error rate is required, the usefulness of the test for detecting attacks and faults is very low. Better trade-offs between the false alarm rate and the test usefulness can be obtained when longer sequences are used for testing and when more complex tests are applied. Unfortunately, both of these are in contradiction

with the design requirements for light-weight implementations and low latency. Possible trade-offs between test efficiency, latency and area can be somewhat improved by tailoring the tests for a particular entropy source or a particular attack scenario [9]. However, a very low decrease of entropy remains difficult to detect with lightweight tests.

Another problem that remains unsolved with this strategy is guaranteeing robustness of the generator. Statistical tests that operate on raw random numbers can detect weaknesses only after the entropy has decreased. This means that some of the compromised numbers may already be sent to the application at the time the fault is detected. Possible methods to provide robustness include providing more aggressive post-processing to account for the drop in entropy (which has a penalty of reduced throughput) and providing enough storage to buffer the generated numbers (which has a penalty in increased area).

3 Canary numbers based health testing

In this work, we aim to improve the trade-offs between test efficiency and latency while using very simple lightweight tests. Rather than tailoring the tests for a particular entropy source, we design the source for improved testability. The central idea of this paper is that the noise source produces 2 bit streams: *the raw numbers* and *the canary numbers*. The canary numbers are not sent to the application, but only to the testing module. The statistical quality of these bits is weakened by design in order to increase the susceptibility to attacks.

We note that similar concepts are used in other areas of security. For example, in software security, canary values are used to prevent buffer overflow attacks and, in hardware security, a method called *canary logic* [8] is used for detecting fault attacks by deliberately increasing the critical paths of redundant hardware modules. The name originates from the analogy with the role of the canary in a coal mine as an early detection of reduced oxygen levels. The goal is to sacrifice a part of the design that is not important in order to obtain an early warning of the fault and to save the part of the design that is important. The same principle is applied in this work. By applying statistical tests on the canary bits it is possible to detect attacks at an early stage, so that the alarm can be triggered before the entropy of the raw bits drops significantly.

The statistical features (entropy, bias, probabilities of generating a given pattern) of the raw numbers and the canary numbers are functions of environment parameters such as jitter variance, delays of individual logic circuits or noise strength. We will denote these parameters as $e_1, e_2, ...$ Usually, only one of the environment parameters is critical for entropy generation. For example the strength of the accumulated jitter is the main contributor to entropy in all jitter based TRNG designs. We will denote the critical environment parameter with e_c and its value at the operating point with $e_{c,op}$. When $e_c = e_{c,op}$ the generator is guaranteed to produce raw numbers with enough entropy. The attack is performed by changing the environment parameters, for example by reducing the

ambient temperature, thereby reducing the amount of jitter below an acceptable level.

To guarantee the robustness of the design: entropy of the raw numbers at the operating point should not be sensitive to small changes of the critical parameter, i.e. the slope of the min-entropy estimation H_{raw} should be very small.

$$\left.\frac{\partial H_{raw}}{\partial e_c}\right|_{e_c=e_{c,op}} \approx 0. \tag{1}$$

High testability is obtained in precisely the opposite situation, i.e. a statistical feature measured by the health test should change when the value of the critical parameter changes. If we denote this feature by f, we can formally define testability as:

$$testability = \left.\frac{\partial f}{\partial e_c}\right|_{e_c=e_{c,op}}. \tag{2}$$

This value should be maximized. Intuitively, it seems difficult to find a feature with high testability if the condition of Equation 1 holds. While it may be possible to find such a feature, it is probably easier to re-design the entropy source to produce two bit streams: *raw numbers* with high robustness and *canary numbers* with high testability.

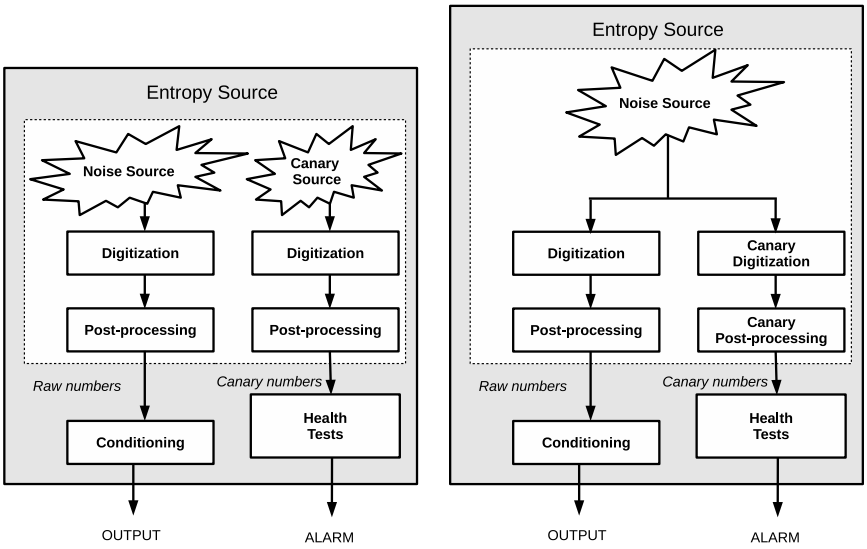
Figure 2 shows the 2 proposed architectures of the entropy source using the canary-based testing.

Replica based architecture This architecture (shown in Figure 2a) is based on designing a weaker replica of the entropy source for testing purposes. This replica, called *the canary source* is simply a copy of the noise source with different design parameters. The design parameters of the digitization and the post-processing blocks can optionally also be changed to improve testability.

This architecture can be useful for detecting global attacks (such as underpower or temperature attacks). However, it cannot detect localized attacks that affect only the entropy source. For this reason, this countermeasure should be used only in addition to other health testing techniques.

Canary extraction based architecture This architecture is based on the notion that digitization and post-processing are forms of signal processing designed with the goal of extracting entropy. Our idea is to re-design these components with high testability as the design goal. The design parameters of the digitization block $(d1, d2, ...)$ and post-processing block $(p1, p2...)$ should be tuned to maximize testability rather than min-entropy. New digitization and post-processing techniques are then applied to the same signal (produced by the noise source) that is used to generate raw numbers. The resulting canary bits are sent to the health test.

This architecture has higher applicability than the replica based architecture because the same noise source is used to produce both the raw numbers and the



(a) Replica based architecture.

(b) Canary extraction based architecture.

Fig. 2: The proposed architectures of the entropy source with canary-based testing.

canary numbers. The replica based architecture should be used only if it is not possible to use a canary extraction based architecture for a given noise source.

4 Case Study 1: Elementary RO based TRNG

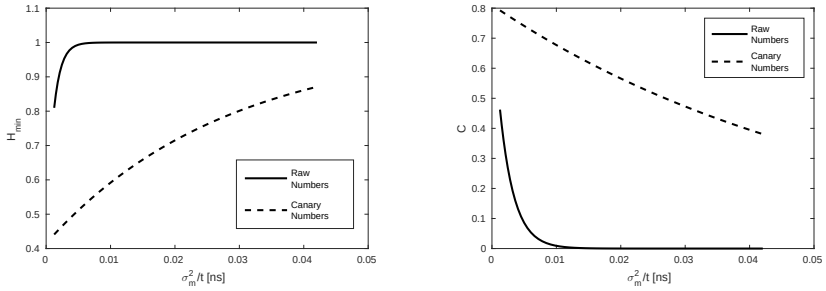
In this section we illustrate the replica-based architecture on the example of an elementary ring-oscillator (RO) based TRNG. The noise source of this generator is a free-running ring oscillator of period T_1. The oscillator output is sampled by a slow clock of period T_2. Entropy per bit increases with the sampling period because longer jitter accumulation time results in higher unpredictability of the sampled bit.

A stochastic model of this generator is presented in [2]. We use the findings of [2] to justify the choice of the testing strategy and to determine the design parameters of the canary source.

The paper [2] proposed the following estimation of the lower bound on entropy:

$$H > \frac{4}{\pi^2 ln(2)} e^{-4\pi^2 Q}, \tag{3}$$

where Q denotes the quality factor defined as the relative variance of the jitter accumulated during the sampling period $T2$



(a) Min entropy. (b) Auto-correlation.

Fig. 3: Estimations obtained from the stochastic model of the elementary ring-oscillator based TRNG.

$$Q = \frac{\sigma_m^2(T_2)}{T_1^2}. \tag{4}$$

The paper also suggest using the auto-correlation coefficient c as a test statistics for health testing.

$$c = \frac{\sum_{i=2}^{l}(-1)^{a_i+a_{i-1}}}{l-1}, \tag{5}$$

where l is the length of the sequence under test.

An estimation of c is given by:

$$c = \frac{8}{\pi^2} cos(2\pi\nu) e^{-2\pi^2 Q}, \tag{6}$$

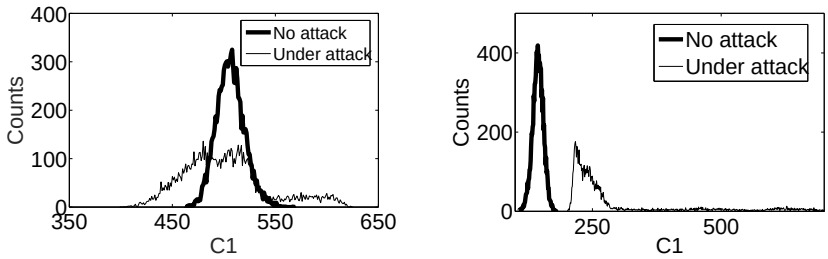
where $\nu = T_1/T_2$.

The entropy of this TRNG depends on the strength of the accumulated timing jitter. The variance of the accumulated jitter grows linearly with accumulation time. We will denote the scaling factor with σ_m^2/t. This factor is the critical environment parameter of this entropy source.

$$e_c = \frac{\sigma_m^2}{t}. \tag{7}$$

The canary source is designed by changing the parameters of the entropy source. In this case, the only design parameter is the number of stages of the ring oscillator. To justify our testing strategy, we look into the entropy and the auto-correlation estimations given by equations 3 and 6 for ring oscillators of different size.

A Xilinx Spartan-6 FPGA was chosen for implementation platform. A 3-stage ring oscillator is used for producing raw numbers. A 15-stage ring oscillator is used for producing canary numbers. The value of the critical parameter at



(a) Auto-correlation distributions of the raw bits under normal operating conditions and under attack.

(b) Auto-correlation distributions of the canary bits under normal operating conditions and under attack.

Fig. 4: Test results of the elementary ring-oscillator based TRNG.

room temperature and normal operating conditions is approximately $e_{c,OP} = 0.0144ns$.

Figure 3a shows the estimations of entropy per bit depending on e_c for raw numbers (full line) and canary numbers (dashed line). Estimations of auto-correlation are shown in Figure 3b. The statistical quality of the raw numbers is robust with respect to small changes of the critical parameter close to the operating point ($\sigma_m^2/t = 0.0144ns$). The entropy per bit around the operating point is almost 1 and the auto-correlation is close to 0. Only an extreme change in the operating conditions can cause a degradation of statistical quality of the raw bits. At the operating point, the canary numbers have lower statistical quality than the raw numbers but the testablity is clearly improved because the slopes of the entropy and auto-correlation graphs are clearly much higher.

In order to verify the proposed design of canary numbers, auto-correlation is used to analyze the bit dependence of raw numbers and canary numbers respectively. For every experiment, the auto-correlation value C_1 is computed on a sequence of 1024 bits.

$$C_1 = \sum_{i=2}^{l} a_i \oplus a_{i-1}. \tag{8}$$

The experiment is repeated 10000 times at room temperature (no attack) and with the FPGA cooled down using a freezer spay (under attack). Histograms of obtained data are shown in Figure 4. The auto-correlation distributions shown in Figure 4a are located very close to each other. On the other hand, as shown in Figure 4b, the auto-correlation distribution of the canary numbers under normal working conditions and under attack are clearly distinguishable.

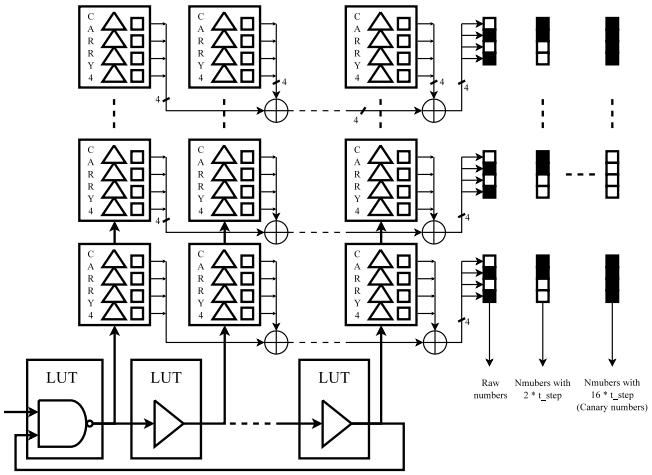


Fig. 5: Architecture of the carry-chain based TRNG.

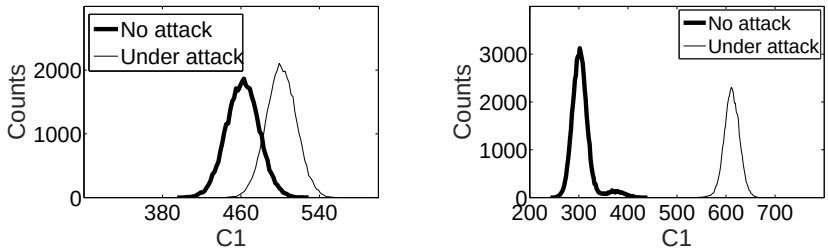
5 Case Study 2: A Carry-chain based TRNG

In this section we illustrate the canary extraction based architecture on the example of a carry-chain based TRNG [7].

The architecture of this generator is shown in Figure 5. A free running ring oscillator is used as the noise source. The phase of the ring-oscillator is affected by white noise. The signal of the ring oscillator is digitized using high-resolution tapped delay lines. On FPGA, these tapped delay lines are implemented using carry-chain primitive which is designed to be ultra-fast and highly precise. At the sampling moment, the position of the signal edge is captured by the tapped delay lines. The accumulated jitter causes the unpredictability of this position. The sampled data from the tapped delay lines are post-processed using an xor operation and a priority encoder.

The entropy of this TRNG depends on the jitter accumulation time t_A and the step size of the time-to-digital conversion performed in the tapped delay lines t_{step}. The canary numbers for this TRNG are designed by increasing the value of this parameter. This is achieved by down-sampling the tapped delay lines. Different priority encoders are used to detect the edge of the signal in different resolutions, which are corresponding to different $n \cdot t_{step}$. To find the most useful canary numbers, down-sampling factors $n = \{2, 4, 8, 16\}$ are examined. $n = 16$ is selected in the end for generating the canary numbers.

In order to verify the proposed design of canary numbers, auto-correlation is used to analyze the bit dependence of raw numbers and canary numbers respectively. For every experiment, the auto-correlation value C_1 is computed on a sequence of 1024 bits. The experiment is repeated 100000 times at 30^oC (no



(a) Auto-correlation distributions of the raw bits under normal operating conditions and under attack.

(b) Auto-correlation distributions of the canary bits under normal operating conditions and under attack.

Fig. 6: Test results of the carry-chain based TRNG.

attack) and at 0^oC (attack). Histograms of obtained data are shown in Figure 6. The auto-correlation distributions shown in Figure 6a are located very close to each other. Even though these distributions are clearly different, it is not possible to design a test to determine the origin of data based on a single C_1 value. On the other hand, as shown in Figure 6b, the auto-correlation distribution of the canary numbers under normal working conditions and under attack are much more distinguishable. One single threshold value can be used to detect the origin of the data. The canary numbers clearly manifest a detectable defect at the early attack stage while the raw numbers are still not significantly compromised.

6 Conclusions

We proposed a health testing design methodology based on the canary numbers. The intention of this methodology is to detect attacks at an early stage, before the quality of the raw bits significantly decreases. This is especially important for use cases where a conditioning component has fixed compression rate. Initial experiment results indicate that canary-based testing is a promising approach.

Our recommendation is to modify the current version of SP 800-90B to foresee the possibility of designing an entropy source for improved testability.

References

1. E. Barker and J. Kelsey. Recommendation for the entropy sources used for random bitgeneration. NIST DRAFT Special Publication 800-90B, 2012.
2. M. Baudet, D. Lubicz, J. Micolod, and A. Tassiaux. On the security of oscillator-based random number generators. *Journal of Cryptology*, 24(2):398–425, 2011.
3. D. J. Bernstein, Y. Chang, C. Cheng, L. Chou, N. Heninger, T. Lange, and N. van Someren. Factoring RSA keys from certified smart cards: Coppersmith in the wild. In *Advances in Cryptology - ASIACRYPT*, pages 341–360, 2013.

4. N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman. Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices. In *Proceedings of the 21th USENIX Security Symposium*, pages 205–220, 2012.
5. W. Killmann and W. Schindler. A proposal for: Functionality classes for random number generators. BDI, Bonn, 2011.
6. A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter. Public keys. In *Advances in Cryptology - CRYPTO*, pages 626–642, 2012.
7. V. Rožić, B. Yang, W. Dehaene, and I. Verbauwhede. Highly efficient entropy extraction for true random number generators on FPGAs. In *Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015*, pages 116:1–116:6, 2015.
8. M. Wachs and D. Ip. Design and integration challenges of building security hardware IP. In *Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015*, pages 177:1–177:6, 2015.
9. B. Yang, V. Rožić, N. Mentens, W. Dehaene, and I. Verbauwhede. TOTAL: TRNG On-the-fly Testing for Attack detection using Lightweight hardware. In *Proceedings of the 2016 Design, Automation & Test in Europe DATE*, 2016.