High quality random number generator from constrained devices using post processing methods applied to a high level network

Guilherme Salgueiro[1], Telmo Monteiro[2], Diogo Costa[1], Miguel Teixeira[1,4], Armando N. Pinto[3], José P. Santos[1]

[1]Departamento De Engenharia Mecânica - Universidade de Aveiro (UA)
Caixa Postal 3810-193 – Universidade de Aveiro – Aveiro – Portugal

[2]Departamento de Física – Universidade de Aveiro (UA)
Aveiro, Portugal.

[3]Departamento de Electrónica, Telecomunicações e Informática – Universidade de Aveiro (UA)
Aveiro, Portugal.

[4]Renault Cacia
Aveiro - Portugal

{guilhermesalgueiro, telmo.monteiro, d.costa, miguel.s.teixeira, anp, jps}@ua.pt

Abstract. *Secure transmission of data in communications based on constrained devices is an understudied issue with severe consequences, as it may jeopardize the entirety of a network's security. Most cryptographic methods are based on random numbers, however, obtaining these proves challenging on devices with such low computational power. In this paper, we evaluate the use of post-processing techniques to increase the quality of randomly generated numbers in 32-bit RISC-V processors, within the context of a blockchain network. Through these methods an increase of entropy of 80% was achieved.*

Resumo. *O envio de dados através de comunicações baseadas em dispositivos de baixa capacidade é uma assunto pouco estudado com consequências graves, podendo colocar em risco a segurança de uma rede inteira. A maioria dos métodos criptográficos é baseada em números aleatórios, no entanto, obtê-los é um desafio em dispositivos com tão baixo poder computacional. Neste artigo, avaliamos o uso de técnicas de pós-processamento para aumento da qualidade de números aleatórios, provenientes de processadores RISC-V de 32 bits no contexto de uma rede blockchain. Através destes métodos alcançou-se um aumento de entropia de 80 %.*

1. Introduction

The problem of establishing encrypted communications with constrained devices comes from their low computational capacity. This leads to issues as in random number generation, typically through subpar methods, compromising security. Solving these problems can possibilitate the use of low level devices in higher level networks, such as blockchains, without introducing new points of failures. Therefore, achieving high quality encryption requires, at some stage, the use of randomly generated numbers. However, not all

random number generators (RNG) are truly random. These generators work based in entropy sources and can produce poor quality numbers, if the entropy is incorrectly harvested or the employed algorithm is too basic. If an encryption is based in repetitive numbers, it will lead to leaks of information and attacks of personification vulnerability. The normal approach to achieve good encryption with this type of processor is to generate the numbers in other equipment, however we try to solve this problem directly in the equipment that sends the information. In [Loza et al. 2015], [Ahmad et al. 2017] and [Li and Ge 2019] all achieved significant improvement to the entropy with hash functions. Despite these improvements not being applied to numbers generated by RISC-V CPUs, they ultimately result in a better entropy, statistical diffusion and high resistance to collision, highlighting hash functions as lightweight starting point for post processing of the numbers created using the RISC-V architecture. A hash function is a computationally efficient method of producing a short output from a variable-length string, called hash value, [Aumasson 2017]. These values can be used for efficient verification of information. Furthermore, hashing algorithms are deterministic, meaning that if the same entry is given, the same output will be obtained,[Aumasson 2017]. Moreover, they are resistant to pre-image, which means that through the hash value it is not possible to obtain the original text. At last, the hash functions are resistant to collision, which means two different texts always result in different output values, despite tending to be similar. The Secure Hash Algorithm 256 (SHA-256) is part of the SHA-2 family, developed by National Security Agency (NSA) and made public in 2001. Currently, the SHA-256 function is the most common choice, since it is a good commitment between computing cost and encryption, [Forouzan 2010]. In this paper, we will characterize the quality of the randomly generated numbers from the 32-bit RISC-V processors, as we have not found information about this generator, and present a post processing method to drastically improve these numbers.

This work is intended for an enterprise blockchain, therefore a private typology was used. Several algorithms can be implemented to guarantee a good functioning of the chain, with Proof of Authority being used in this implementation, as it allows for more efficient operation, suitable to Private Blockchain's, [Zheng et al. 2018]. From a programming standpoint, it is possible to represent a Blockchain as a data structure, where entry (blocks) are stored and linked in a sequential order. The linkage between each one of the blocks is assured by storing the hash value of the previous block. A Blockchain can be built from decentralization (all the elements have the same amount of control in the chain), data tamper resistance and traceability of information, which stands for the possibility to track information from the elements involved in the transactions, [Zou et al. 2020].

2. Developed Architecture

This implementation, represented in the figure 1, is based in a Blockchain topology and feeded with data from the IoT devices. The Blockchain has two main functions: validation of data through the mining nodes and data storage. To keep the Blockchain lightweight, only the hash value that characterize each transaction is stored. To keep the chain running, the nodes fulfill two different tasks. The first one is based on administrative responsibilities, in which they have the responsibility of managing the message flow between the several layers and nodes, while validating front-end users' access to the network and regulating the access to information. The administrative nodes are also hold responsible of running the mining nodes and the gateways in the chain. The second task is performed

by the mining nodes, which can be deployed across multiple locations. As the number of mining nodes increases, so does the overall trust of the network, as more entities exist to validate and share the common ledger.

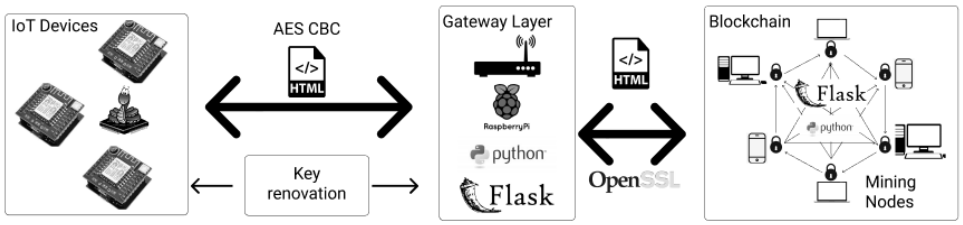

Figure 1. Proposed Architecture

The novelty of this paper lies in the IoT Devices. These layer is the lowest level of the architecture and is comprised by three ESP8266, equipped with a RISC-V 32-bits CPU. They are responsible for data collection from built-in light dependent resistors (LDR) and encryption, sending data to the gateway layer. To encrypt the information, Advanced Encryption Standard (AES) in mode Cipher Block Chaining (CBC) was used. This method requires a key, known by the gateway, an initialization vector (IV) and the data to send. The key was generated by an UUID, Universally Unique Identifier, according to the RFC 4122 norm. This key must be replaced in order to be less susceptible to attacks. After the keys replacement, it must be sent to the gateway in a secure channel. In this implementation, instead of the key, its hash value is used, calculated through SHA-256. The novelty of this work stands in the way that we post process the keys and the IV to guarantee a secure communication. Until now, it was not possible to use this devices in a secure way.

The Gateway Layer bridges communication between the sensing devices and the decentralized ledger. Transmitting information to every node in the blockchain network requires several security features, such as the capability to establish secure connections (e.g., TLS and Hypertext Transfer Protocol Secured (HTTPS)). To implement HTTPS connections between the gateway and the nodes, a self-signed certificate was implemented, with both server-side and client-side requiring a certificate. To achieve this, the certificates were made using OpenSSL[4](1.1.1j), ensuring network access, [The OpenSSL Project 2018].

3. Randomness Capability from IoT Devices

The message identification methods have as a principle the usage of an unique key that comes from a random key generator, perceived as a random number generator. In other words, the equipment that generated the key was based in random variables and is virtually impossible to obtain the same number or even similar randomization. However, this does not always happen. Every random number generator (RNG) has its foundation on entropy sources to generate its numbers. The entropy is defined as the measurement of disorder or randomness, being min-entropy the measure used by the National Institute of Standards and Technology (NIST), consisting in the unpredictability of a random variable, [Barker and Kelsey 2012]. This said, the min-entropy (in bits) of an independent discrete random variable X is the largest value m having the property that each observation of X provides at least m bits of information of X, [Barker and Kelsey 2012]. More accurately, the min-entropy of an independent discrete random variable X that takes values from the set $A = \{x_1, x_2, ..., x_k\}$ with probability $P_r(X = x_i) = p_i$ for $i = 1, ..., k$

is defined as $H = \min_{1<i<k}(-\log_2 p_i) = -\log_2 \max_{1<i<k} p_i$. If *X* has min-entropy *H*, then the probability of observing any particular value for *X* is no greater than 2^{-H}. The maximum possible value for the min-entropy of a random variable with *k* distinct values is $\log_2 k$, which is attained when the random variable has a uniform probability distribution, [Barker and Kelsey 2012]. Another important concept is Independent and Identically Distributed (IID) data sets: each sample is independent from each other and uses the same distribution as the others. This means that an element, in a sequence, is independent of the random variables that came before it. The opposite, non-IID data, refers to a sequence where the probability distribution for the nth random variable consists in a function of previous random variables in the sequence, [Barker and Kelsey 2012]. To summarize, a random number generator of n-bit numbers will have its average min-entropy between 0 and n. A min-entropy value close to 0 reveals a weak performance, while a value close to n is considered an high quality value.

RNGs can be divided into two groups: True (based in physical processes), where the quantum and classical belong; and the Pseudorandom (based in algorithms). These two groups differ in the way they generate entropy. The classical is the most conventional, as it generates entropy by reading surrounding environment conditions, be it temperature, luminosity, humidity, etc, [Wilber 2013]. The quantum generators obtain their entropy based on quantum physics.Comparing the two types of generators, when it comes to quality, both shall present min-entropy values around or superior to 7.5. The inherent problematic to this comparison is focused on the entropy sources, given that in common generators the entropy source can be influenced and controlled to obtain a desired value. For example, if the entropy generators source reads the surrounding temperature, if we control this temperature, we would be able to control the generated entropy. Therefore, the quantum generators grant an higher security, given that its entropy is based on quantum physics, being extremely difficult to alter these entropy sources, [Wilber 2013].

4. Tests and Results

In order to develop a secure communication between the Raspberry Pi and the lower level IoT devices a study to the numbers from this generators was made, the question about the capacity of these modules to provide numbers that were in fact random surged. This is due to the fact that to randomly generate a number, the equipment needs to have quality entropy sources and to utilize them correctly, since there are norms the generator must obey, like the NIST SP800-90B, [Barker and Kelsey 2012]. The entropy used by the generator also depends on the operating system. In Windows for example, a larger variety of sources regarding entropy exists, since it derives from microprocessors that some computers have or from integrated processes inside the processors, [The OpenSSL Project 2018]. However, the ESP's entropy source is unknown when it is functioning with Micropython based firmware, as well as the quality of the random numbers produced by the Raspberry Pi. We proposed to apply the tests provided by NIST in their tool available at their github, [NIST, Joshua E. Hill, Morey Adam, Laufmann Stefan, Celi Chris 2015], to test if somehow we can produce a secure communication with this devices. The number collecting process and the NIST tests application is represented in figure 2. Analysing Table 1, it's visible that the entropy of the ESP is around 1.08, from 0 to 8, that means a grade of 13.55%. If we search the value of other good generators, we see that the medium value expected is higher than 7.5 and the Raspberry Pi achieved an average of 7.49 for 20 MB

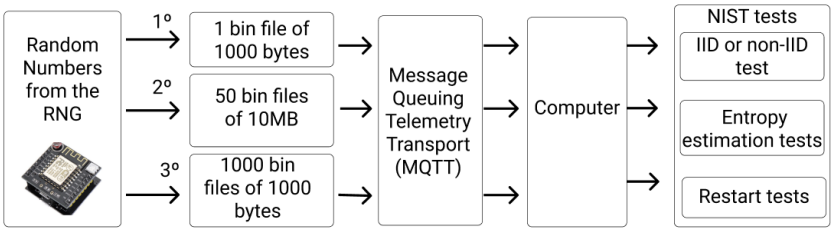

Figure 2. Process to apply NIST tests

files. Given this, it is possible to conclude that the generator used in ESP was of a poor quality, and the way that it generates the numbers compromised the IV from the AES CBC mode. If the generator repeats values or if it is possible to predict the next value, it is also possible to decode the information and the private key. Therefore, the implementation is vulnerable to attacks that intend to read information or, later, impersonate. The good results from the Raspberry Pi also explained why is so usual to found implementations where the generation of this numbers is not done in constrained devices.

Table 1. ESP8266 and Raspberry Pi min-entropy and standard deviation

	10 MB	20 MB	
ESP8266	1,084	1,071	Average
	0,0003	0,080	Standard deviation
Raspberry	7,438	7,494	Average
	0,172	0,200	Standard deviation

To improve the communications quality, the numbers can be generated in the Raspberry with Python and sent to the IoT layer. However, this higher complexity means more information transactions, leading to bigger exposure. An alternative to this is post-processing the numbers from the ESP, so a study was executed on the commonly used alternative proposed by NIST. Said alternative was executed and based on hash function applying on the numbers generated through the ESP. To test this option, 512 bits were generated and post-processed with SHA-256, getting 32 bytes of output. The same tests were applied to these numbers and the improvement was notorious, achieving an average of 7.494 for 20MB files. This post-processing method results in a increase from 13,38% to 93.68%, close results to the values obtained using more complex generators, including quantum ones or a Raspberry Pi.

5. Conclusions

This work has as its main objective to achieve a secure encryption from constrained devices, making possible information transmission to feed a blockchain without information leakage. If we analyze the computational time that would be required for this implementation, the option of generating the numbers in ESP without post processing takes less than 0.5 milliseconds. However, if we post process the number it leads to 1 millisecond. At last, if we simulate generating the numbers in the Raspberry Pi and send it to the ESP, it will lead to times around 300 milliseconds. Although this value is drastically higher, this does not compromise its usage. However, the complexity of the implementation is

relatively high, so the post process stands as the one with the best cost-benefit. This study was done with the purpose of checking the large usage and quality of the hash functions and, in this case, they became a very powerful post processing method. Generating numbers with such an high entropy is still a current challenge, but data encrypted with these numbers are only vulnerable to brute-force methods, requiring more than several decades to decrypt it. In future works, the implementation of a secure channel with constrained devices to transmit the quantum numbers is required, executing a study on quality of the numbers when entropy sources are controlled, in alternative a study to implement several entropy sources, including quantum ones, to reach similar results, however this may be compromised by the computational capacity of this devices.

6. Acknowledgements

This work is supported by Fundação para a Ciência e a Tecnologia (FCT) through national funds, by the European Regional Development Fund (FEDER), through the Competitiveness and Internationalization Operational Programme (COMPETE 2020), under the project QuantumPrime reference: PTDC/EEI-TEL/8017/2020. This work is funded by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/50008/2020-UIDP/50008/2020 (action QuRUNNER).

References

Ahmad, M., Khurana, S., Singh, S., and AlSharari, H. D. (2017). A Simple Secure Hash Function Scheme Using Multiple Chaotic Maps. *3D Research*, 8(2):1–15.

Aumasson, J.-P. (2017). *Serious Cryptography*. No Starch Press, Inc.

Barker, E. and Kelsey, J. (2012). Recommendation for the Entropy Sources Used for Random Bit Generation, NIST Special Publication 800-90B.

Forouzan, B. A. (2010). *TCP/IP Protocol Suite*. Higher Education, 4 edition.

Li, Y. and Ge, G. (2019). Cryptographic and parallel hash function based on cross coupled map lattices suitable for multimedia communication security. *Multimedia Tools and Applications*, 78(13):17973–17994.

Loza, S., Matuszewski, L., and Jessa, M. (2015). A Random Number Generator Using Ring Oscillators and SHA-256 as Post-Processing. *International Journal of Electronics and Telecommunications*, 61(2):199–204.

NIST, Joshua E. Hill, Morey Adam, Laufmann Stefan, Celi Chris, K. M. (2015). Nist GitHub.

The OpenSSL Project (2018). OpenSSL: The open source toolkit for SSL/TLS. `www.openssl.org`.

Wilber, S. A. (2013). Pure Quantum True Random Number Generators. Technical report, The Quantum World Corporation.

Zheng, Z., Xie, S., Dai, H.-N., Chen, X., and Wang, H. (2018). Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14:352.

Zou, Y., Meng, T., Zhang, P., Zhang, W., and Li, H. (2020). Focus on Blockchain: A Comprehensive Survey on Academic and Application. *IEEE Access*, 8:187182–187201.