Cryptography using Evolutionary Computing

J. Blackledge[*], S. Bezobrazov[**], P. Tobin[†] and F. Zamora[††]

[*†] *School of Electrical and Electronic Engineering, Dublin Institute of Technology, Ireland*

[**] *Department of Computer Science, Brest State Technical University, Brest State Technical University, Belarus*

[††] *Department of Electronic Engineering, University of Baja California, Mexico*

E-mail: [*]jonathan.blackledge@dit.ie [**]bescase@gmail.com
[†]paul.tobin@dit.ie [††]fzamora@uabc.edu.mx

Abstract — **We present a method of generating encryptors, in particular, Pseudo Random Number Generators (PRNG), using evolutionary computing. Working with a system called *Eureqa*, designed by the Cornell Creative Machines Lab, we seed the system with natural noise sources obtained from data that can include atmospheric noise generated by radio emissions due to lightening, for example, radioactive decay, electronic noise and so on. The purpose of this is to 'force' the system to output a result (a non-linear function) that is an approximation to the input noise. This output is then treated as an iterated function which is subjected to a range of tests to check for potential cryptographic strength in terms of a positive Lyapunov exponent, maximum entropy, high cycle length, key diffusion characteristics etc. This approach provides the potential for generating an unlimited number of unique PRNG that can be used on a 1-to-1 basis. Typical applications include the encryption of data before it is uploaded onto the Cloud by a user that is provided with a personalised encryption algorithm rather than just a personal key using a 'known algorithm' that may be subject to a 'known algorithm attack' and/or is 'open' to the very authorities who are promoting its use.**

Keywords — **Coding and Encryption, Evolutionary Computing, Multiple Algorithms, Personalised Encryption Engines**

I Introduction

In Patrick Mahon's secret history of Hut 8 - the Naval Section at Bletchly Park (Station X) from 1941-1945 - it is stated that [1]: *The continuity of breaking Enigma ciphers was undoubtedly an essential factor in our success and it does appear to be true to say that if a key has been broken regularly for a long time in the past, it is likely to continue to be broken in the future, provided that no major change in the method of encryption takes place.* This statement relates to the famous Enigma encryptor used by German armed forces from the mid-1930s until 1945 and, to a lesser but equally important extent, the more advanced Lorenz encryptor used from mid-1942 onwards for high-level communications between the German High Command in Berlin and Army Commands throughout occupied Europe. The issue of the 'method of encryption' relates to the Kerckhoff-Shannon Principle, namely, *A crypto-system should be secure even if everything about the system, except the key, is public knowledge* [2] or as stated more succinctly by Claude Shannon *The enemy knows the system.* This paper shows how evolutionary computing could be the key to breaking with the Kerckhoff-Shannon principle. To this end, we provide a short back-ground to the case which contextualises the issue and then considers the use of a evolutionary computing system called *Eureqa* [3] for generating ciphers using input data streams

consisting of natural noise.

II The Kerckhoff-Shannon Principle

The Kerckhoff-Shannon Principle has been the foundation of cryptographic research for many decades and emphasis has and continues to be placed on the exchange of the keys (of increasing length) to operate (i.e. encrypt/decrypt data) specific symmetric and asymmetric algorithms that have proven cryptographic strength. However, it is well known that many cryptographically strong algorithms and/or the keys used to 'drive' them have been broken in practice. The reasons for this are as varied as the encryption methods used, at least, those that are known about.

New encryption algorithms and system are of course the subject of continuing research but, irrespective of this, there are a number of practical reasons for abiding by the Kerckhoff-Shannon Principle. These include the following: (i) the algorithm is a good one, e.g. it is robust and cryptographically strong; (ii) legacy code and the procedures and protocols associated with the use of an algorithm; (iii) the expense associated with changing the algorithm(s). However, there is another issue which we call the *Enigma Syndrome*. This relates to the concern that an encryption algorithm is often the product of the very authorities who want it to be used, so called because of the value that the stock pile of Enigma machines had after 1945 in terms of gaining intelligence from governments world-wide which, at the time (i.e. from the late 1940s and the early 1950s), were encouraged to use it and early derivative of it [4]. Form the late 1950s to date, issues of this type led to the development of new cipher bureaux's world wide whose focus was and continues to be to generate new and unique encryption algorithms for use by the governments they represent.

Since the end of the cold war in the early 1990s, and, with the rapid development of computing and communications technology, many new companies have been established to either sell existing encryption systems and/or develop new approaches to data security. This led to the Regulation of Investigatory Powers (RIP) Act in the UK, for example, introduced in 2000 to regulate the powers of public bodies to carry out surveillance and investigation including the interception of communications, taking into account technological changes such as the growth of the internet and strong encryption introduced by the new generation of ciphers being developed at the time.

Although the RIP act was introduced for important and valid reasons, it highlights an issue that defines two principal landmarks in the history of Cryptography. If the years of 1900-1945 are taken to be the 'Battle of the Code Makers verses the Code Breakers' then from 1945-date we have and are continuing to witness the 'Battle between the Code Makers verses the Code Controllers'. In this context, and, with regard to the relatively recent introduction of Cloud computing, one of the principal issues of Cloud users is the security of the data that they uploaded onto the Cloud and whether standard commercially available encryption algorithms are secure enough for this purpose. Within the context of the *Cloud Security Alliance* [5] the following issues are becoming important: (i) the perception that many encryption schemes that abide by the Kerckhoff-Shannon principle are weaker than publicly acknowledged; (ii) the *Enigma Syndrome*. Even if point (i) above could be proved not to be an issue to the satisfaction of users, with regard to point (ii), a principal question has become: How can we trust the code controllers? One answer to this question is to 'generate our own codes'.

Irrespective of the technical challenges associated with users generating and/or using their own code (i.e. encryption algorithms), there is another over-riding factor that is important to understand and is compounded in the following quotation [6]: *Cryptology is like literacy in the Dark Ages. Infinitely potent, for good and ill, yet basically an intellectual construct, which by its nature will resist efforts to restrict it to bureaucrats and others who deem only themselves worthy of such privilege* . In this regard, we explore how Evolutionary Computing has the potential for 'democratising' data encryption by allowing individuals to acquire or even develop their own personalised encryption algorithms rather than relying on a personal (private) key alone to 'drive' a standardised algorithm open to public scrutiny. The context for attempting this is based on the following forecasts: (i) by 2016, annual global IP traffic is forecast to be 1.3 Zettabytes (1 Zettabyte = 1 Trillion Gigabytes); (ii) by 2016, there are expected to be 3.4 Billion Internet Users which amount to approximately 45% of the world's projected population. One of the consequences of these forecasts is the urgent need for ICT solutions to drive research priorities in the H2020 programme [7], for example, with regard to internet data security and, in particular, the security of personal data by application of unique encryption algorithms for encrypting data before it is uploaded into the Cloud through applications such as Dropbox and MS Office 365.

III Complexity, Randomness and Chaos

Algorithmic complexity and chaos underpin the development of modern encryption algorithms along with mathematically definable concepts such as unpredictability and entropy, for example. The design of any crypto system can be interpreted in

terms of designing a key-dependent bijective transformation that generates a data stream or 'string' which is bit-for-bit unpredictable to an observer with finite resources [8]. Crypto systems (which incorporate the design of Pseudo Random Number Generators (PRNG), the structure of an encryption algorithm that uses PSNG and key exchange protocols, for example) are predicated on the generation of time series or digital signals which are typically based on an intreated function (the algorithm). Upon the encryption of data using the algorithm and its transmission and/or storage, a cryptanalyst will be expected to have access to the time series and the algorithm under the Kerckhoff-Shannon Principle (i.e. the algorithm has been made publicly available for public scrutiny). The time series is not a compact subset of the trajectory (intermediate states are hidden) and the iterated function is taken to have a secret parameter (the key).

An 'algorithm' is designed to have a number of properties that provide cryptographic strength which, on a generic basis, are taken to include 'randomness', 'unpredictability' and 'complexity' [8]. More specifically, these properties include ensuring that the time series are uniformly distributed (with no bias toward any trajectory, thereby providing a maximum entropy cipher), a high positive Lyapunov exponent (ensuring that the trajectory becomes chaotic within a few iterations), a high cycle length and good diffusing properties so that different keys (involving a change of a single bit) produce different ciphers in which all bits of the bit stream have an equal likelihood of changing their state [9].

A 'perfect PRNG' can be used to generate 'perfect security' if the cipher text is absolutely unpredictable to an external observer, i.e. all possible outcomes (states, sub-trajectories etc.) are equiprobable and do not depend on the previous states. In other words, the state sequence has a uniform probability distribution and no correlations (matching patterns). The concept of absolute unpredictability is equivalent to true randomness and related to 'white noise'. In practice, cryptographic systems only provide a certain level of data security that is usually much lower than that of a (theoretically) perfect system. This is due in part to the need to develop encryption algorithms that are practicably usable, primarily for reasons of cost effectiveness. In this context, it is necessary to deal with 'pseudo' concepts in which pseudo-random number sequences cannot be efficiently distinguished from uniform noise and where computationally unpredictable sequences cannot be predicted with available computer resources. This involves a range of concepts that need to be quantified to produce the theoretical framework for designing and assessing encryption algorithms and includes: (i) algorithmic complexity which considers the length of the shortest algorithm producing a cryptographic sequence where, on an intuitively basis at least, the internal complexity of the system provides (external) unpredictability; (ii) algorithmic randomness in which the output sequence is equal to the length of the sequence and is computationally incompressible containing no recognisable matching patterns or redundancies. A diagrammatic illustration of the relationship between these concepts is given in Figure 1.

With reference to Figure 1, we note that a purely random system is also algorithmically random. However, the concepts of pseudo and algorithmic randomness are different. A pseudo-random string is generated with a compact seed, but the external observer is not able (practically) to reconstruct the generator and predict the sequence. In other words, the string is highly compressible for authorised communication parties, but computationally incompressible for the adversary. In the general case, an algorithmic random string can be predicted by a probabilistic machine.

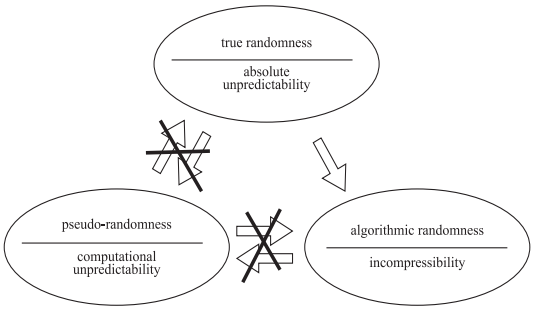

Fig. 1: Relationship between the concepts of real randomness, algorithmic randomness and pseudo randomness [9].

The randomness of a string can be 'measured' using properties such as the algorithmic complexity or the entropy which is taken to be a measure of the uncertainty about the exact state of any element of the string and where this measure is the same for all elements in the string. Quantitatively, the Shannon information entropy is in direct proportion to the algorithmic complexity where, in ergodic systems, the statistical properties of a single sequence coincides with that of all sequences, emitted by a PRNG. The randomness measure for chaos is the Kolmogorov-Sinai entropy [4] which is a multi-resolution integration of Lyapunov exponents. In practice, entropy is maximised if the cipher produces an output that is uniformly distributed or else can be post-processed to produce a new output that is uniformly distributed without significant data redundancy. In comparison

with a 'fully predictable system' where all states are known and with a complexity of 1, a 'fully unpredictable system' (i.e. 'delta uncorrelated white noise') is an infinite source of information, infinite complexity and entropy. These are the theoretical extremes and real cryptographic systems lie somewhere in-between: they are complex enough to be unpredictable by an external observer, but not too much to be reproducible. In this context there are two principal source types that can be considered as discussed in the following sections.

a) Natural Noise

Natural noise (e.g. environmental noise available from a range of sources) is highly dimensional with infinitely many states and independent variables. However, in generally, the entropy of such a system may not maximal because of its self-organisational properties and correlations. This is because many noise sources are random self-affine strings, the fractal geometric properties (in a statistical sense) being a well known and a fundamental property of many forms of natural noise. There are cryptographic applications using natural chaos; for example, Intel's hardware-based PRNG captures randomness from the thermal noise of the computer. Such random sequences are used only in key generation, not in encryption, because they are not reproducible.

There is an important historical example of the use of natural noise. This relates to the SIGSALLY (Green Hornet) encryptor developed by AT & T Bell Labs used by Prime Minister Winston Churchill and President Franklin Roosevelt for 1-to-1 transatlantic communications from 1942-1946 [4]. The encryptor was based on the addition of noise to voice signals to produce a output (analogue) signal with minimal a signal-to-noise ratio, scrambling the speech signals over all frequency bands. The source of the noise was derived from a vacuum tube by recording the output of the tube (given no input signal) on a phonograph record. The result was a recording of electronic noise which was used to mask voice signals through addition of the recorded noise. The technology required to apply this approach in practice involved the time registration associated with the addition and subtraction of the noise source in a two-way sense. The distribution of these noise sources (i.e. the recorded media) was strictly controlled for obvious reasons. However, provided this 'control' was not comprised, the system represented a one-time pad and was effectively impossible to attack successfully. Even today, such a cipher would be difficult to attack using Bayesian strategies, for example, on the assumption that a statistical model for the Probability Density Function of the additive noise can be acquired and/or on assuming that the noise is fractal, thereby providing a model for the Power Spectral Density Function (PSDF) of the cipher.

b) Natural Chaos

Low-dimensional chaotic noise has infinitely many states but a small number of independent variables. Nevertheless, such systems cannot be applied directly to digital encryption because they cannot be implemented on a finite-state machine. It is only possible to apply an approximation to a chaotic system using a (typically nonlinear) iteration function working to finite floating point precision which yields a limited cycle length after which the string is repeated, a consequence that is common to all PRNG implemented on a digital computer. For applications to cryptography, the aim is therefore to find the best numerical implementation of a chaotic system which maximises the cryptographic strength of the cipher subject to minimum algorithmic complexity.

c) Pseudo Chaotic Encryption

The use of pseudo chaos for designing ciphers is now well known, acknowledged and widely used. The origins of this approach date back to the early 1950s when Claude Shannon explicitly mentions the basic stretch-and-fold mechanism now associated with chaos and as used in cryptology. There was then a 'silent period' until the late 1980s when emphasis was placed on implementing standardised symmetric and asymmetric encryption algorithms commercially such as the Digital Encryption Standard (DES which was later modified to the DES3 by encrypting with a key K_1, decrypting with another key K_2 and then encrypting again with K_1 in order to triple the length of the operational key without changing the algorithm) and the Rivest, Shamir & Adleman (RAS) algorithm (which requires a Public Key Infrastructure to be established for the generation, management and certification of the keys), respectively. Following the popularisation of chaos theory in the 1980s, it started to be applied to cryptography in the 1990s when some 30 publications appeared suggesting various ciphers but focusing on the application of analogue circuits for real time applications in spread-spectrum based military communications, for example. This included the use of Fractal Modulation [4], for example, used to hide the spread spectrum in natural RF noise. However, since 2000 the application of 'digital chaos' for encrypting data has grown exponentially with many chaotic maps being suggested by various authors and the development of multi-algorithmic systems to encrypt data on a randomised block-by-block basis [9].

There are many disadvantages in using chaos for cryptography especially with regard to the need to

compute the cipher to high floating point precision subject to the inclusion a partitioning strategy applied to the state space in order to provide a maximum entropy string (a necessary post-processing step which generates redundancy in the floating point input, thereby waisting CPU time). The principal value of chaos is the ability to create many different algorithms. This is of course possible with conventional random number generators (such as Knuth M-algorithm) but chaos provides greater diversity in terms of the functions available (other than the mod function, for example). The problem is that, to date, in order to produce a library of different algorithms, they have had to be designed 'by hand' often by modifying specific and well known chaotic iteration functions such the logistics map (modified by Matthews to produce the so called Matthews map which stretches the key space [9]) or by 'trial and error', i.e. 'inventing' nonlinear Iteration Function Systems and testing them for their properties with regard to cryptographic strength. The tests required for cryptographic strength which all 'modified' and/or 'invented' maps must pass, include the following: (i) large positive Lyapunov exponent relative to a known algorithm with accepted cryptographic strength, e.g the Advanced Encryption Standard (AES) [4]; (ii) the potential for generating a uniformly distributed cipher, i.e. can the output of a chaotic map be partitioned to produce a cipher with a completely uniform discrete Probability Density Function (without waisting too much data); (iii) the Power Spectral Density Function of the cipher is uniformly distributed thereby making a spectral attack redundant; (iv) the autocorrelation of the (digital) cipher is a (Kronecker) delta function indicating that there are no correlation's within the length of the cipher set to be used for encryption and thus, the cycle length is beyond the upper limit that has been set; (v) the CPU time required for the floating point computations (typically to double precession) is acceptable for the given hardware.

The most important point in the list above are points (ii), (iii) and (iv). This is because, an infinite and truly random string has no statistical bias, a delta autocorrelated function and an infinite and uniform power spectrum (white noise). With regard to point (ii), for example, assuming the existence of one-way 1:1 functions, there can exist probability distributions, which are not uniform and are not even statistically close to a uniform distribution, but are, nevertheless, computationally indistinguishable from a uniform distribution [10]. Hence, checking for equal probability of the states is fundamental. A high (but strictly positive) Lyapunov exponent is preferable because the iteration function it is taken to characterise will generate chaotic trajectories within a few iterations. However, these tests do not guarantee the diffusive properties of a cipher, namely, that the PRNG is 'Structurally Stable'. Ideally, we require an algorithm that has (almost) the same cycle length and Lyapunov exponent for all initial conditions. Most of the known pseudo-chaotic systems do not possess this property and there is no rigorous analytical method, as yet known, for assessing this property. This is an important problem because without solving it, it is not possible to guarantee that a crypto system based on a deterministic chaotic algorithm or set of algorithms will always produce uncorrelated strings for any and all keys. Another issue is that of algorithmic complexity which cannot be commuted, i.e. there is no universal solution for simplifying programs and for proving that the length is minimal. We cannot apply this definition directly to compare the complexity of cryptographic sequences or algorithms. Nevertheless the theoretical applications are very important. In particular, the Kolmogorov complexity provides a unified approach to the problem of data compressibility [4].

Subject to these important and, as yet, unresolved issues, although the applications of 'digital chaos' has yielded commercially realisable products it is not scalable. In this paper we explore the use of evolutionary computing to scale up the process by using natural noise as the input to an evolutionary process. In this sense, we explore a way of automating and diversifying the approach to produce a potentially unlimited number of one-time-pads using a range of noise sources in analogy with the Green Hornet transatlantic scrambling principle of 1942-46, discussed earlier.

IV Evolutionary Computing

Evolutionary Computing is associated with the field of Computational Intelligence, and like Artificial Intelligence, involves the process of continuous optimisation. Artificial Intelligence aims, through iterative processes, to compute a set of optimal weights that determine the flow of information (the amplitude of a signal at a give node) through a network that simulates a simple output subject to a complex input. In this sense, an Artificial Neural Network (ANN) simulates a high entropy input with the aim of transforming the result into a low entropy output. However, this process can be reversed to generate a high entropy output from a low entropy input. In this sense, a ANN can be used to generate a cipher by simulating natural noise once it has been trained to do so. To use a ANN in this way, the cryptographer requires knowledge of the ANN algorithm and the weights that have been generated through the training process (i.e. the input of the noise sources used to

generate the weights). Figure 2 shows an example of the input noise (obtained from recordings of atmospheric noise provide by [11]) and the ANN simulated output. The type of ANN that is used for this process is crucial and it has been found that a 'Radial Basis ANN' is best suited for the purpose, details of which lie beyond the scope of this paper. Given this statement, the precise ANN algorithm becomes analogous to a PRNG in conventional cryptography and the weights are equivalent to the key.

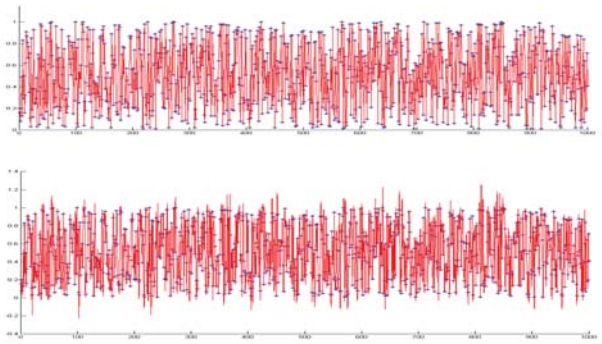

Fig. 2: Example of training a ANN to simulate a genuine random number stream: Original noise (above), and ANN approximation (below).

While an ANN approach to generating ciphers is of value in special cases, it does not provide the same flexibility in terms of using a formulaic approach to designing PRNG using iterated (nonlinear) functions. To do this an evolutionary algorithms approach is required in which a population-based, stochastic search engine is required that mimics natural selection. Due to their ability to find excellent solutions for conventionally difficult and dynamic problems within acceptable time, evolutionary algorithms have attracted interest from many areas of science and engineering. The application of evolutionary algorithms to cryptology as presented in this paper is, to the best of the authors knowledge, an original concept. Full details of the approach used and the results obtained to date lie beyond the scope of this paper and will be published elsewhere. However, in the following section we present an example result based on the processing steps quantified in the schematic shown in Diagram 1.

Noise Source: (e.g. Atmospheric Noise)
↓
Evolutionary Computing System
↓
Function ↔ Approximation to Noise Source
↓
Post Processing (of Iterated Function)
↓

Cipher ⟶
- Positive Lyapunov Exponent ✓
- Uniformly Distributed Statistics ✓
- Uniformly Distributed PSDF ✓
- Acceptable Cycle Length ✓
- Acceptable CPU Time ✓
- ⋮

Diag. 1: Schematic of the processes for evolving a cipher.

V Example Result

We report on one of a growing database of ciphers being generated using *Eureqa* developed at the Cornell Creative Machines Lab (Cornell University, USA). The system iteratively develops a nonlinear function to described complex input signals usually associated with experimental data on a chaotic system. If genuine random (delta uncorrelated) noise is input into the system, then from a theoretical point of view, no nonlinear function should be found on an evolutionary basis. Thus, inputting natural noise is a way of 'cheating' the system to 'force' it to provide a result that may be suitable (on an iterative basis) as a PRNG (subject to the tests outlined in Diagram 1). The input used can be obtained from any available source, online or otherwise.

For this study, we use the data available from RANDOM.ORG which, to date, has generated 1.28 trillion random bits for the Internet community [11]. Figure 3 shows an example screen shot of the *Eureqa* system used to generate the following iteration functions for cipher generation

$$\begin{aligned} c_{i+1} = 129.68 + 68.41 \sin(c_i \sin(\cos(\cos(1.54 + c_i))) \\ + \sin(\sin(2.54 + c_i + 85.75 c_i^{-1}) \\ - \cos(2.23 \cos(0.63 c_i)))) \end{aligned} \quad (1)$$

The highly non-linear iteration function given by equation (1) is the result of *Eureqa* undertaking over 100 iterations (using 250 noise samples randomly selected from the data bases available at RANDOM.ORG) to evolve the result, taking approximately 23 hours using a Intel - Xeon 2.40 GHz Processor to do so. While equation (1) provides a valuable iterator (subject to normalisation so that $c(i) \in (0, 1] \forall i$ and post-processing based on the tests described in Diagram 2), it can not provable that this equation is structurally stable, i.e. that a cryptographically strong cipher is guaranteed for any floating point value of c_0 between 0 and 1, say, irrespective of the precision of c_0. Thus is important because c_0 (which seeds and thereby initiates the cipher stream) could, for example, be generated by a Hash function from a low bit private key and possibly fail at some point in the future for lack of structural stability. However, this is in keeping with many other PRNG.

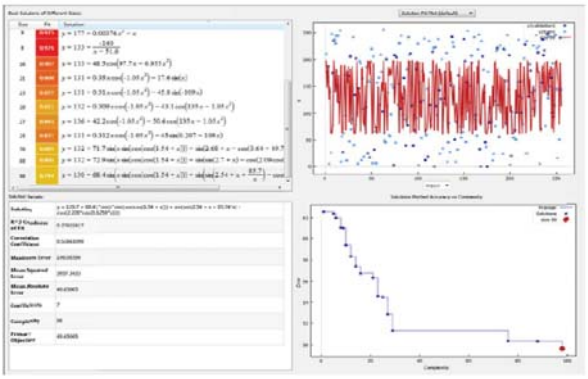

Fig. 3: Screen shot of Eureqa used for evolving nonlinear functions suitable for cipher generation.

VI Conclusions

Practical cryptography is based on passing known statistical tests, e.g. [12] which is designed to ensure the pseudo-random property of a generator, pseudo-random sequences being taken to be used instead of truly random sequences in most cryptographic applications. This paper introduces a way of designing algorithms for generated pseudo random (chaotic) sequences using truly random strings to evolve an iterator that is taken to be an approximation to these sequences. This approach pays no attention to the algorithmic complexity of the iterator which is one of the main problems in the application of chaos to cryptography. Neither does it consider the structural stability of the iterator or its algorithmic complexity. However, it does provide a practical solution to the problem of developing a large database of PRNG for the application of personalising encryption algorithms for strictly 1-to-1 communications or '1-to-Cloud' (encrypted) data storage. By using evolutionary computing systems such as *Eureqa* seeded with noise, it is possible to generate a nonlinear function f with appropriate control parameters. Using this function in an iterative form with an additional transformation g say, and a partition function σ, a PRNG suitable for encrypting data can be constructed. The combined effect of g and σ is that of a hard-core predicate. However, the one-step unpredictability does not guarantee that the output sequence will be unpredictable when an adversary has access to a sufficiently long sequence. In other words, the vast number of samples can, on a theoretically basis at least lead to the predictability. With these provisos, the work reported in this paper demonstrates that evolutionary computing provides the potential for generating an unlimited number of ciphers which can be personalised for users to secure their 'Data on the Cloud'. Algorithms can be published so that the approach conforms to the Kerckhoff-Shannon Principle as in the example provided, i.e. equation (1), in the knowledge that a new set of evolutionary computed algorithms can be developed. Since 2012 over 300 ciphers have been produced in this way, and, in summary, the technique may present a technical solution to the 'democratisation of the cipher bureaux'.

Acknowledgments

Jonathan Blackledge is supported by the Science Foundation Ireland Stokes Professorship Programme. Segei Bezobrazov is funded by the Erasmus Mundus Action II co-operation and mobility programme EWENT (East-West European Network on Higher Technical education) managed by Warsaw University of Technology, Poland. The authors are grateful to Dr Marek Rebow at Dublin Institute of Technology for arranging the authors collaborative research programme.

References

[1] O. Hoare, *Enigma: Code Breaking and the Second World War - The True Story through Contemporary Documents*, Introduced and Selected by Oliver Hoare, UK Public Records Office, Richmond, Surrey, 2002.

[2] A. Kerckhoff, "La cryptographie militaire", Journal des Sciences Militaires, Vol. IX, pp. 583, January 1883, pp. 161191, February 1883.

[3] Eureqa, "A software tool for detecting equations and hidden mathematical relationships in your data", Cornell Creative Machine Lab, USA, 2013, `http://creativemachines.cornell.edu/eureqa`

[4] J. M. Blackledge, *Cryptography and Steganography: New Algorithms and Applications*, Centre for Advanced Studies Text-books, Warsaw University of Technology, ISBN: 978-83-61993-05-6, 2012.

[5] The Cloud Security Alliance `https://cloudsecurityalliance.org/`, 2013.

[6] Cloud Risk, `http://www.securesql.info/?tag=quotes`, *A Thinking Man's Creed for Crypto*, Vin McLellan, 2013.

[7] The Horizon 2020 Programme `http://ec.europa.eu/research/horizon2020/index_en.cfm?pg=h2020`, 2013.

[8] N Ptitsyn, *Cryptography using Deterministic Chaos*, De Montfort University, 2004

[9] J. M. Blackledge and N Ptitsyn, "On the Applications of Deterministic Chaos for Encrypting Data on the Cloud, Third Interna-

tional Conference on Evolving Internet, INTERNET 2011, 19-24 June, IARIA, Luxembourg, ISBN: 978-1-61208-008-6, 78-87, 2011.

[10] L. Kocarev. *Chaos and cryptography*, 2001, http://rfic.ucsd.edu/chaos/ws2001/kocarev.pdf.

[11] RANDOM.ORG: *True Random Number Service*, 2013, http://www.random.org

[12] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, and S. Vo "A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications". NIST, 2001. http://csrc.nist.gov/rng/rng2.html.