Design and Analysis of a Noise-Based Random Oracle

Albert Wang, Jeff Gebert, and Paul Davis, *University of British Columbia*

***Abstract*—The design of many cryptographically secure systems require random numbers for use in protocol. Often, the implementation of the pseudorandom number generation component is critically flawed, and exploitation of this weakness by attackers can ultimately lead to the failure of the system. Truly random numbers can be efficiently distilled from naturally occurring entropic processes, providing ideal protection. An ideal source of random numbers is atmospheric radio frequency noise. A method of generating random numbers from such noise has been implemented. Statistical and probabilistic tests confirm the quality of randomness in the values produced. A method of implementing such a system commercially as a novel Random Oracle is proposed.**

***Index Terms*—computer security, cryptography, random oracle, RF noise, RNG, random number generator**

I. Introduction

Random number generation has been a requirement of secure cryptology since the advent of the one-time pad. With computers has come pseudo-random number generation, but many random number generators (RNGs) implemented electronically have been broken with careful cryptanalysis. A more secure source of random numbers is a chaotic physical process; atmospheric noise and nuclear decay are two obvious examples. A method of building a random number generator with atmospheric noise is discussed below. We include a discussion of using the RNG to create a cryptographically secure random oracle that might be easily implemented on a PC, for example.

II. Background Information

Random sequences are classified into two categories, based upon their origin: those created through *pseudorandom number generators* (PRNG), and those extracted from *truly random generators* (RNG).

Manuscript submitted December 2, 2005.

A. W. is with the University of British Columbia, BC, Canada. He is pursuing an undergraduate degree in Electrical Engineering with a Minor in Commerce (e-mail: awang@ ece.ubc.ca).

J. G. is with the University of British Columbia, BC, Canada. He is pursing an undergraduate degree in Engineering Physics (e-mail: gebert@interchange.ubc.ca).

P. D. is with the University of British Columbia, BC, Canada. He is pursuing an undergraduate degree in Engineering Physics (e-mail: pfd@interchange.ubc.ca).

PRNGs are algorithms that accept a *seed value*, or starting point for a method that will yield a fixed sized bit string. Strictly speaking, the algorithm takes the seed information and transforms it iteratively toward the intended output. Any pseudo-random generator, after a sufficient number of steps will return to a value, or state that has already been visited. This property defines a period for the algorithm.

Computers, as finite state machines, are completely deterministic; algorithms executed on computers will invariably possess the (in this case) undesired property of periodicity. While it is theoretically possible to design an algorithmic random number generator that did not exhibit periodicity, the memory requirements of such an algorithm would grow as it ran [1].

It is important to note that it is not difficult to design PRNGs with periods such that, given current computing paradigms, would take a time on the order of the lifetime of the universe to return to its original starting point [1]. However, this is often not the vulnerability exploited in the attack of secure systems.

Secure systems designers are under increasing pressure to develop systems that are computationally economical, temporally efficient, and transparent in operation. However, these goals are commonly at odds with robust design principles, and as a consequence, weak algorithms are used in places that are critical, in addition to seeds being drawn from a relatively limited seed space [2].

A pseudorandom number whose seed was drawn from a small population can be easily found by attackers. For instance, it is common practice to seed pseudorandom number algorithms with the system time of the computer at the time of instantiation. Given the generation algorithm, an attacker need only search the relatively small window of time to determine a list of possible outcomes. This has caused several system failures, including a rather public incident involving early versions of Netscape's Secure Socket Layer (SSL) encryption protocol [2].

III. Design of an RF-Based Random Oracle

A. RNG System Architecture

The entropy-based RNG we implemented relies on capturing ambient radio frequency (RF) signals as a physical source of noise and logging the data. Random numbers are extracted from the electromagnetic noise and RF signals in the atmosphere to provide a random oracle based on a physical source of noise. Previous research has indicated that

electromagnetic noise can be modeled as a stochastic process [3] - [7]; hence, RF noise is an excellent and practical choice for a physical source of randomness that can readily be found.

The principle of building RF-based RNGs is readily transferable to personal computers because computer circuit boards themselves generate a considerable amount of electromagnetic interference (EMI) [8]. This noise, which is nondeterministic in nature, adds a favorable amount of entropy to the measurements.

Our RF-based random number generator was designed to include a reception antenna, a data acquisition hardware module, processing software, and data logs. The reception antenna and data acquisition hardware is used as the basis of the system to acquire raw data from a stochastic process.

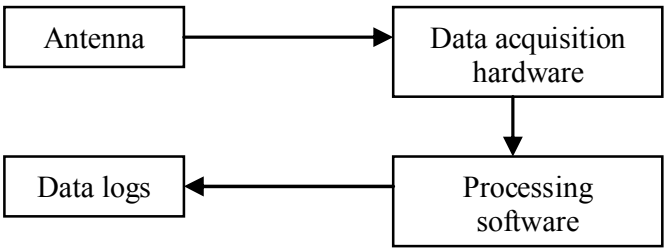

Fig. 1. Block diagram of the designed RF-detection RNG random oracle

A Cisco antenna was used for initial trials, and was later augmented with a propagation channel configured on a Spirent TAS4500 channel emulator to simulate an RF broadcast attack. A hypothetical attack in which a malicious party broadcasts a strong signal on the frequency the data acquisition hardware is scanning in an attempt to influence the randomness of the result was investigated in this manner.

Data was captured through either an Anritsu MS2721A spectrum analyzer or an Agilent E8362B Professional Network Analyzer (PNA). The MS2721A spectrum analyzer was used for its portability to dtermine the environmental differences in performance, while the E8362B PNA was used for its ability to measure dual-channel data and phase information. Using the portable spectrum analyzer, initial tests were conducted to check if there was a noticeable difference in the output magnitude measured in dBm. The dual-channel PNA was used primarily to compare the properties of the phase information vis-à-vis the magnitude of the signals measured. It was also used to verify that the signal from a transmitter would still have a random phase at the receiver due to scattering [8]. The protocol for data acquisition from the Agilent PNA is represented in Figure 1.

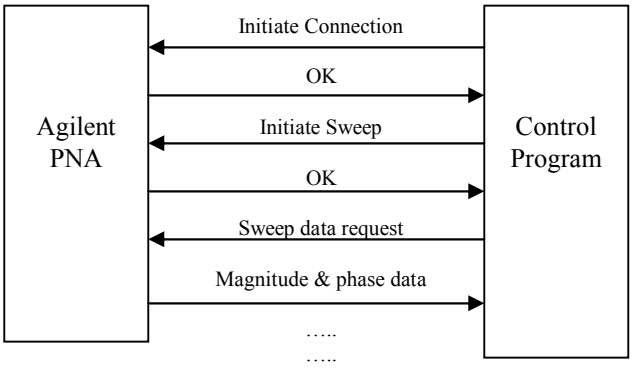

Fig. 2. Data acquisition protocol between PNA and controlling computer

Data acquisition and processing software was written in Microsoft Visual Studio using C++ to interface with the RF measurement equipment. Sweeps were taken at 800 and 900 MHz because they are noisy commercial and public safety channels used for voice and data communications [9]. The processing software translated data downloaded from the RF measurement devices into an ASCII file of 501 to 16,001 data points and associated the data with a time stamp at the time of measurement.

The data logs consist of a collection of sweep data, taken at various times and time-stamped. If necessary, the logs can be searched; a process facilitated by the time stamps. Ordered by their time stamp, the data logs can easily be sorted through in O(log N) time with a binary search [10].

B. System Testing and Verification

The system was tested thoroughly to check that requirements for a secure system were met. Measurement data was manipulated with post-processing software in MATLAB to obtain suitable output for our random oracle. A simulated RF transmission attack was modeled using a Spirent TAS4500 channel emulator.

Initially, the logarithmic magnitude data was recorded directly from the spectrum analyzer and examined for randomness. The results were poor due to sustained signals in the channel from personal communications devices (e.g. a 900 MHz cordless phone conversation).

An attack was simulated with a strong signal sent to the RF-receiver over a propagation channel exhibiting standard delay and fading characteristics [11]. Due to unpredictable phase variations of the received signal, no unusual levels of correlation were observed. This strongly suggests that an attacker cannot decrease the randomness of the unit by introducing a strong jamming signal into the environment because the phase of the signal cannot be controlled on the receiver side over an open propagation channel. Consequently, multi-path propagation effects and delay spread between two different paths will make attempts to physically influence the phase measurement impossible [12].

IV. Approaches to Assessing Randomness

Determining a definitive measure of randomness is a somewhat difficult process as the concept of randomness is defined in several contexts. However, for the purposes of cryptography, there are three generally accepted properties [13]:

- Output indiscernible from white noise
- Unpredictability
- Forward Unpredictability

The first property defines qualitatively what a random stream of bits should resemble – a concept familiar to most practitioners of communication systems. However, this definition does not easily lend itself to systematic approaches of assessment.

The latter two properties can be framed mathematically, in both statistics and information theory.

A. Statistical approach

The National Institute for Standards and Technology has published a statistical test suite for applications related to cryptography [14]. This set of tests forms is rather standard and correlates well with those performed by other researchers [15, 16]. Implemented in C++, these tests accept data files of zeros and ones, and perform up to 16 different methods of analysis.

Before any testing could take place, a preprocessing step was necessary to convert the ASCII data files into an acceptable format. To do this, the data was parsed digit by digit. Digits whose values were between 0 and 7 were converted to their binary equivalent in three significant figures. Since this was phase data stored in terms of degrees, it ranged between -180º and +180º. To avoid a skewed output, only the digits to the right of the decimal were recorded.

Five of the NIST tests were conducted on several data files. The other 11 tests were implemented with an unchangeable lower limit on the number of bits required. These lower limits prevented us from obtaining results for these tests.

The first test that was run was the "Frequency (Monobit) Test". The purpose of this test is to determine the closeness of the fraction of ones to ½. This test was critical, as all subsequent tests are dependent on its success. The second test, a slight variant to the first, is called the "Frequency Test within a Block". This test focuses on the proportion of ones within M-bit blocks.

The "Cumulative Sums Test" measures the maximal excursion (from zero) of the 1D random walk taken by the bits, if the bits are adjusted to (-1,+1). In general, non-random sequence will stray from the origin, while random sequences will tend to stay near the origin.

The "Non-overlapping Template Matching Test," and the "Discrete Fast Fourier Transform Test" are tests that detect periodicity in the output stream. As the name might suggest, the non-overlapping template test compares a pre-specified bit pattern to the test data. The template is incremented in position in a non-overlapping fashion. The Fourier transform test measures the peaks of the spectra of the test data, and determines its periodicity.

The output of the RF-noise tests is summarized in Table 1. In this battery of tests, the data was submitted as 200 partitioned bit streams of 1000 bits. The P-Value, or "tail probability" is defined as the probability that the chosen test statistic will assume values that are equal to or worse than the observed value for the particular test [14]. The proportion column denotes the proportion of bit streams that passes the NIST default requirements for randomness. In this configuration, the minimum proportion was 0.9685.

Table 1: P-Values and the Proportion of Passing Sequences, RF-RNG

P-VALUE	PROPORTION	STATISTICAL TEST
0.249991	0.9730	frequency
0.264458	0.9892	block-frequency
0.109149	0.9838	cumulative-sums
n/a	0.9722	fft
n/a	0.9892	nonperiodic-templates

Table 2: P-Values and the Proportion of Passing Sequences, DRBG Using SHA-1

P-VALUE	PROPORTION	STATISTICAL TEST
0.435390	0.9871	frequency
0.194562	0.9834	block-frequency
0.103183	0.9759	cumulative-sums
n/a	0.9990	fft
n/a	0.9831	nonperiodic-templates

Table 2 summarizes the same battery of tests conducted for the pseudorandom generation method "DRBG Using SHA-1", an algorithm included as reference in the test suite. As can be seen, both sequences pass the NIST requirements.

B. Probabilistic Approach

In addition to the statistical approach to quantifying correlation in the signal, a probabilistic analysis of the signal was undertaken. In communication research, noise is generally modeled as a random process. Specifically, a Markov process is the standard description of channel noise in most applications. A Markov process is generally defined as a process whose present state is dependent only on the previous states occupied by the system.

In the case at hand, the signal is analyzed as a first-order Markov process, which is simply the assumption of a stochastic process whose present state depends only on its most recent previous state. Mathematically, we can express the probability of a given present bit value (say, zero) as

$$P(c_n=0) = P(c_n=0 \mid c_{n-1}).$$

The transition probability in a Markov chain is defined as the likelihood that a given bit will be followed by another given bit – for example, a_{01} will represent the probability that a zero in the sequence is followed by a one. Although these probabilities cannot be calculated in the present system, they have been estimated numerically by normalizing the number of transitions from one specific state to another and normalizing

to the total number of transitions possible. This method produces the following transition probability matrix:

$$A = \begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix} = \begin{matrix} 0.5077 & 0.4923 \\ 0.5007 & 0.4993 \end{matrix}$$

Further, the entropy of an information signal is defined as a measure of the information content of a source X, H(X) [18]. From the transition matrix, it is clear that within our numerical accuracy, both bit values are equally likely. Therefore, we can treat the source as a random binary source of bit probability *p*, with entropy

$$H(X) = -p \log_2 (p) - (1-p) \log_2 (1-p)$$

This function is plotted in Figure 3. Since for our system *p* is very close to 0.5, H(X) ≈ 1.0 and entropy is optimized (very near the peak of the curve). This essentially means that each bit value is varying independently and the signal is as random as possible.

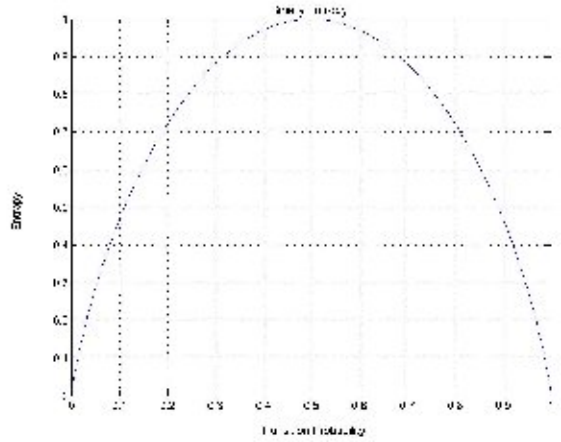

Figure 3. The Binary Entropy Function

V. Discussion

A. Applications

The weaknesses of pseudo-random number generators have been examined above. Several applications in particular are significantly weakened by poor RNGs, and thus could benefit from a strong method of random number generation. In symmetric encryption schemes, information security is accomplished via secret key pairs and sometimes nonces. If these keys are successfully attacked, the information being transmitted can be easily read. Hence, a truly random number source could provide significant additional security for session key generation.

In a similar way, the security of password generation and one-time pads can be greatly increased with a completely unpredictable RNG. In other fields like gambling or Monte Carlo simulations, it is similarly critical to employ highly uncorrelated random values. Thus, there is considerable demand for a method capable of supplying those values in a useful way

B. Random Oracle Framework

We propose building on the existing knowledge of chaotic physical systems to encapsulate an RNG as a Random Oracle [19]. The Random Oracle model is an abstraction in cryptographic theory which is usually implemented using hash functions as pseudo-random number generators, leading to some insecurities[20]. It has two requirements:

i. If a new query *x* has been received by the oracle, a random value is return
ii. If the query *x* has already been received by the oracle, the same value is returned as was previously returned.

This framework is well-suited to the system, since the first requirement is easily met by pulling a value from the random bit stream. In order to accomplish the second objective, an auxiliary memory device must be incorporated into the system. Given the availability of fast, cheap and high-capacity memory systems, this is a feasible implementation. The oracle, above satisfying the above requirements, must be demonstrably secure to provide assurance as well as effectiveness to potential users.

C. Security Concerns

If the RNG in question is to be used in critical security applications, it must be highly secure itself. By using a different approach to producing random numbers, the design also opens doors to new attacks. The most dangerous and likely attacks have been anticipated and countermeasures included in the proposed design. These attacks include most prominently an information attack, a hardware-based data interception and a software-based data interception.
First, in an information-based attack, an attacker would attempt to bombard the system with a deterministic RF signal in the hopes that the RNG sensor would acquire the malicious signal, rather than random noise. In this way, the attacker could control the output of the RNG.

Although this is a serious security concern, the attack can be combated in several ways. Most importantly, it is important to note that a carefully designed receiver and processing system can be made to ignore most RF information signals in its vicinity. The design discussed above demonstrated the ability to disregard qualitative attempts to affect its measurement with deterministic signals. Further, we propose implementing an evaluation module that mimics much of the functionality of the statistical and probabilistic tests already mentioned. For example, Markov estimates and an FFT computation could be built in hardware as an auditing unit. Then, if non-random signatures are detected, countermeasures, channel switching, notification, shut-down or various other actions could be taken at the discretion of the user.

Finally, it is important to emphasize the capability of frequency scanning; dynamic spectral behavior could be used to thwart an attacker beaming a signal modulated at a specific frequency or small bandwidth.

It is also critical to consider attempts to intercept information after it leaves the RNG. To prevent hardware tampering, the most effective solution is careful physical

security. The analogous attack in software is unauthorized interception of random data or query of the oracle. This can be defended against by careful design of software interfacing. Most notably, enforcing least-privilege and complete mediation in software, as well as careful system access controls, will lead to secure information.

These principles of security are illustrated schematically in Figure 4.

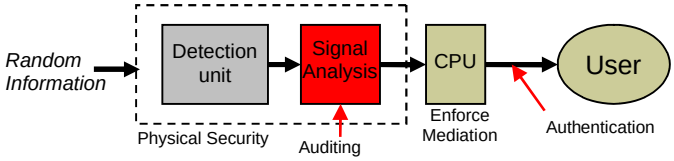

Figure 4. Random Oracle security Layers

D. Commercial Viability

Clearly, the design implemented for this analysis is only a proof-of-principle, as it is large, expensive and difficult to operate. A commercially viable design must be cheap, portable and easy to install and use. Although security performance is important, it is often only achievable if ease of use is first achieved.

However, the step from the present system to a commercial one is not prohibitively difficult. Integrated circuits exist which could cheaply approximate the function of a receiver. Likewise, simple analysis algorithms like Fast Fourier Transforms (FFT) can be simply performed in digital hardware. With careful software development and system integration, it is not difficult to imagine a unit capable of interfacing with a common PC being developed for less than $100. It is worth noting that poor reception components could actually help the system as they contribute to noise.

Although some attempts have been made to other physical processes as a chip-based random number generator, there has only been limited success, particularly commercially [21]. By developing not just a chip but a secure system engineered from the ground up, we believe a much better performing unit may be realized. Moreover, by keeping security as a primary design motivation and performing rigorous analysis on output randomness, we believe assurance in the system can only benefit. This, in turn, can lead to more field implementations and securer systems.

VI. CONCLUSIONS

A method of harnessing the randomness of atmospheric electromagnetic noise has been implemented as a proof-of-principle. Statistical and probabilistic analysis has confirmed a high degree of randomness in the output produced. Moreover, a careful implementation as a random oracle has been examined with critical security methods suggested. Such a system, as demonstrated, holds the potential for marked security improvements from current practice.

ACKNOWLEDGMENT

The authors would like to thank Dr. Dave Michelson of the UBC Radio Science Lab for permission to use his RF test and measurement equipment and his guidance in this project.

REFERENCES

[1] Hall, C., Kelsey, J., Schneier, B., Wagner, Da., “Cryptanalytic Attacks on Pseudorandom Number Generators,” Counterpane Systems publication.
[2] ???
[3] Goldberg, I. Wagner, D., “Randomness and the Netscape Browser,” Dr. Dobb's Journal, January 1996, pp 66–70.
[4] V.G. Spitsyn, “Method of numerical analysis of interaction electromagnetic wave with random active media,” IEEE AP-S Internutional Symposium Digest, Atlanta, Georgia, June 2 1-26, 1998, Vol. 1,pp. 112-115.
[5] V.G. Spitsyn, “Development of a numerical method of electromagnetic wave propagation analysis in the three – dimensional random discrete media,” IEEE AP-S Internutiond Symposium Digest, San Antonio, Texas, June 16-21,2002, Vol. 4, pp. 288-291.
[6] V.G. Spitsyn, “Method of computation of nonlinear electromagnetic wave interaction with stratified absorbing media,” IEEE AP-S International Symposium Digest, Columbus, Ohio, June 22-27, 2003, Vol. 4., pp. 398-401.
[7] V.G. Spitsyn, “Modeling of radiowave scattering on the ionospheric plasma disturbances, created of space vehicle,” Tomsk Publishing House “STT”, 2002.
[8] Parhami, P., Spencer, M.E., Robinson, J.T., Rynne, T.M., “Stochastic process test techniques,” 1993 IEEE International Symposium on Electromagnetic Compatibility.
[9] Shahparnia, S., Ramahi, O.M., “Electromagnetic interference (EMI) reduction from printed circuit boards (PCB) using electromagnetic bandgap structures,” IEEE Transactions on Electromagnetic Compatibility, Nov 2003, Vol. 46, Issue 4, pp. 580 – 587.
[10] Federal Communications Commission Spectrum Policy Task Force, “Report of the Interference Protection Working Group,” Nov 15, 2002. Available: http://www.fcc.gov/sptf/files/IPWGFinalReport.pdf
[11] National Institute of Standards and Technology, “Binary Search,” October 27, 2005. Available: http://www.nist.gov/dads/HTML/binarySearch.html.
[12] Kim, Young Yong, Li, San-qi, “Capturing important statistics of a fading/shadowing channel for network performance analysis,” IEEE Journal on Selected Areas in Communications, May 1999 Volume: 17, Issue: 5, pp. 888 – 901.
[13] Horikoshi, S., Fuji, M., Itami, M., Itoh, K., “A study on multipath propagation modeling in millimeter wave IV,” The 5th International Symposium on Wireless Personal Multimedia Communications, 2002, pp 286-290.
[14] Drake, Christopher, Nicholson, Anthony, “A survey of pseudo-random number generators,”
[15] Ruhkin, A. et al. “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications,” NIST Special Publication 800-22, May 2001.
[16] Tezuka, Shu, Uniform Random Numbers: Theory and Practice, Boston: Kluwer Academic Publishers, 1995.
[17] Engel, E, A Road to Randomness in Physical Systems, Berlin: Springer-Verlag, 1992.
[18] Proakis, Fundamentals of Communication Systems, New Jersey: Prentice Hall, 2005, p. 707
[19] Anderson, Ross, Security Engineering, A Guide to Building Dependable Distributed Systems, New York: Wiley Computer Publishing, 2001
[20] Canetti et al, “The Random Oracle Methodology, Revisited”, Journal of the ACM, Vol. 52, Issue, 4, 2004
[21] Yaseka, et al, “Physical Random Number Generator Based on MOS Structure After Soft Breakdown,” IEEE Journal of Solid State Circuits, Vol. 39, No. 8, August 2004