Formalities

Preliminary Title:
Improved Security Through Randomness Testing

Name:	**CSC Supervisor:**
Joel Gärtner	Douglas Wikström
Email:	**Principal:**
jgartner@kth.se	Omegapoint
Date:	**Principal Supervisor:**
26 januari 2017	Hannes Salin

Background and Objective

Several cryptographic protocols require some unpredictable input in order to provide security. Furthermore, any keys used in the protocols must also be generated in an unpredictable manner since these are crucial for the level of security, thus it must be hard for any adversary to guess or compute such keys. In order to get a source of unpredictability, random generators are used. These generators can potentially be truly random and generate numbers based on some phenomenon which is deemed unpredictable. The speed of these generators is however often limited and because of this it is more common that generators used are pseudo random, i.e. based upon deterministic algorithms, which when given a seed for the generator, produces an arbitrarily long sequence which appears random. Depending on the application, different types of pseudo random generators are needed. For simulations and similar, the most important properties is that it is fast to generate numbers and that they behave statistically in a way that seems random. For cryptographic purposes the generators also have to be unpredictable, meaning that an adversary who sees part of the generated sequence won't be able to efficiently guess the next bit of the sequence with a probability significantly higher than if the sequence was truly random.

These cryptographically secure pseudo random generators will thus expand a seed into a sequence which cannot be efficiently distinguished from a truly random sequence without knowledge of the seed. If the seed is known however, the whole sequence is easily reproducible as the algorithm generating the sequence is deterministic. As such that the pseudo random generator is cryptographically secure will not provide any unpredictability if the seed is predictable. Therefore it is of great importance for a process to select a seed of unpredictable bits which can then be expanded into arbitrarily much seemingly random data using a cryptographically secure pseudo random generator. In order to select this seed there is a requirement for a source of unpredictability. This can for example be an actual true random generator or another process on the generating machine which behaves in an unpredictable manner. It is however not obvious how much data is needed before enough unpredictability has been gathered to the seed. A measure of this unpredictability is the entropy of the sample received from the source. Entropy is a concept taken from thermodynamics which was first used in information theory by Shannon with Shannon entropy [11]. A sequence of bits from a source will have an entropy of 1 per bit if it is completely unpredictable and an entropy of 0 per bit if it is completely predictable. It is easy to estimate the entropy of data if the data consists of samples with n bits where each such sample is generated completely independent from any of the other samples. If this is the case then the unpredictability and thus the entropy only depends on the probability distributions of the single samples x and thus it is sufficient to estimate this distribution to get an estimation of the entropy. In case the samples are not independent identically distributed then it is a lot harder to estimate the entropy.

Several test suites exists which try to determine when data does not behave randomly [5] [6] [10]. These suites can to some extent identify sequences where the samples do not behave like identically distributed random variables. Most such tests do however only identify when there is incorrect behaviour in the sequence and does not give an estimate of the actual unpredictability of the sequence. As the sequence can somehow be distinguished from truly random data, it could be a potential weakness if used in cryptographic applications by itself. The sequence does however probably still contain some amount of unpredictability and this could potentially be extracted by transforming the sequence so that it statistically behaves more like uniformly distributed random data. This can be done with multiple sources of entropy which can be combined in order to create a source of random data which behaves randomly and is unpredictable. This source could then be used to get a seed for a cryptographically secure random generator which could expand this source of unpredictable into as much random data as necessary.

An estimate of the entropy of data is thus useful in order to determine how much data is necessary to get enough unpredictability for random generators. A recent approach to estimating the entropy of data is to use predictors [9]. These estimators take the approach of trying to predict the data which is fed into them as good as possible. The entropy of the data can then be approximated via the success rate of the predictors. Several predictors were constructed in order to give good predictions and thus good estimations of the entropy. All entropy estimations will however have the problem that they only estimate the entropy of some specific types of distributions of data while other distributions will get wrong estimations. The constructed predictors were meant to be general and work in general but more specific estimators will work better for some types of entropy sources. As such there is the possibility of more estimators which better models more types of data sources.

A specific application where there is a need for an entropy estimator is for the random number generator /dev/random on Linux systems. Here data is gathered in a pool and the amount of entropy in the data is estimated and added to an estimate of the entropy in the pool [1]. The random number generator then only returns a number when it deems there to be enough entropy in the pool. Other random number generators which depend on outside entropy have similar systems in order to avoid giving numbers to the users of the generators when the numbers are deemed too predictable. The estimates of the available entropy is of interest as a too high estimate of the entropy could be a potential security vulnerability. There have already been several examples of security vulnerabilities which were the result of a lack of entropy when generating random numbers [7] [12].

Research Questions and Method

Question

Can new types of entropy estimators give better estimates than existing ones on given types of data.

Problem Definition

Investigating existing entropy sources such as /dev/random and seeing how they get their entropy and how they estimate the amount they have. Compare the way these estimates are performed with existing general tests which are meant to estimate entropy of general data. Then produce estimators which try to give better entropy estimates given the data source. Further tests with these estimators will also be performed on data which is produced to follow specific distributions in order to get values to compare the results to.

Examination Method

Existing algorithms for entropy estimation to be tested includes the ones implemented by NIST special publication 800-90b [4] and the ones in Predictive models for Min Entropy [9] as well as the estimates used in entropy sources such as /dev/random. These will be compared with the developed algorithms, which will include algorithms which estimate the entropy in the way done by predictors as well as potentially other type of estimators.

Preliminary the data to be tested is the output of different entropy sources, before they are mixed in an irreversible way with other data. Furthermore tests will be performed on data which have been generated to have a specific distribution and thus have a known entropy, given that the random number generator used for the data is a good one. Potentially some study could be performed on estimating the entropy of output from random number generators, although only on weak such, which are not meant to be cryptographically secure.

Expected Scientific Results

The hypothesis is that new entropy estimators may complement existing estimators by providing better estimates for at least some type of data. This is being tested by producing random data which have a known entropy, with varying distributions, which the estimators can try to give good estimates for. The results will hopefully be useful for better estimations of entropy for some entropy sources which given better entropy estimations can be either safer to use or alternatively more efficient while providing the same level of security.

Evaluation and News Value

Evaluation

The produced estimators will be compared with existing entropy tests on different types of data. In order to evaluate the produced estimators the tests on data which follows known distributions are probably of the most interest as these have a known correct value on the entropy. This gives a clear indication on when the newly created estimators produce better values than existing tests. Any large deviations from the results of existing estimators with the new ones on real world data may then potentially give information about the properties of the real world data or alternatively that the produced estimators are not good matches for this type of data.

News Value

New estimators could be useful for producer of new entropy sources in order to give better estimates of the produced entropy. Existing entropy sources could also potentially benefit from new entropy estimators to give a better estimate and thus potentially eliminate existing weaknesses or give a more efficient stream of random data.

Pilot Study

Some study of the implementations of pseudo random number generators and cryptographically secure pseudo random number generators to get a better understanding of the context. Furthermore study of the Yarrow and Fortuna algorithms and their implementations may be of interest as these use entropy accumulators as integral parts of their algorithm to secure the algorithm even in cases where part of the state becomes known to an attacker.

There has also been guidelines produced by NIST related to random number generators and entropy sources. These documents NIST SP 800-90(A,B,C) [3] [4] [2] are guidelines related to how secure random generators should perform and how sources of entropy sources should behave and how they can be tested. Furthermore researchers at NIST have also produced the paper *Predictive Models for Min-entropy Estimation* [9] related to using predictors to producing better estimates for entropy. Other sources related to how entropy and randomness testing is performed is also of interest, including for example how suites such as TestU01 [10] evaluates data.

Furthermore it will need to be investigated what sources of entropy exist today and how they work. This includes built in entropy generators in operating systems but also specific generators built for that specific purpose. Sites such as *random.org* [8] which is a service that provide random data will also be of interest.

It may also be useful to study the different cryptographic protocols where randomness is useful, in order to see what kind of impact too low entropy may have in relation to security. In combination with this it will also be of interest to investigate the weaknesses which have been discovered that were the result of lacking entropy when generating random numbers for cryptographic protocols.

Conditions and Schedule

The plan is to work more or less full time during the entire thesis project beginning 17 January until it is finished.

Resources

There should not be a big demand on resources or equipment. Potentially output from a hardware source of entropy could be of interest, as it could be another source of entropy to analyse.

Limitations

The project will deal mainly with entropy estimation. There will thus not be a lot of focus on what is done with the entropy or how to produce entropy. How entropy is produced in current implementations is however of interest as this may give insights into how to better estimate the produced entropy.

Collaboration with the principal

The principal will be available for discussion regarding content and direction of the project. He will also be able to proof read any material and give response on what is written. In case of any needed resources he will also be able to give some help in acquiring these.

Schedule

Specification

Begin 17-January 2017

End Week 5

Comment
This specification will hopefully be done and approved by week 5.

Pilot Study

Begin 17-January 2017

End Week 9

Comment
The work with the pilot study was initiated during the work on the specification and will continue when the specification is done for approximately 4 more weeks.

Implementation

Begin Week 8

Working Prototype Week 12

End Week 15

Comment
A working prototype which is relatively close to the final implementation is planned to be finished by Week 12. After this testing and fine tuning of the prototype may be performed in order to get good data for the report which will be written in parallel.

Report

Begin Week 13

Finished first draft Week 17

End Week 23

Comment
There will be work on the report during the other parts of the thesis, but during the indicated time the main focus will be on the report. A first draft of the report should be finished early to give plenty of time for adjustments and comments.

Presentation

Begin Week 16

Finished first draft Week 18

End Week 21

Comment The work on the presentation should be done in parallel with the fine tuning of the report after the first draft has been completed. A finished first draft of the presentation should be done relatively early to give plenty of time to practice the presentation and fine tune it.

Litteraturförteckning

[1] *RANDOM(4) Linux Programmer's Manual.*

[2] E Barker and J Kelsey. Recommendation for random bit generator (rbg) constructions (draft nist special publication 800-90c). *National Institute of Standards and Technology*, 2012.

[3] Elaine Barker and John Kelsey. Recommendation for random number generation using deterministic random bit generators (revised). *NIST Special Publication*, pages 800–90, 2011.

[4] Elaine B Barker and John Michael Kelsey. *(Second DRAFT) Recommendation for the Entropy Sources Used for Random Bit Generator.* US Department of Commerce, National Institute of Standards and Technology, 2016.

[5] Lawrence E. Bassham, III, Andrew L. Rukhin, Juan Soto, James R. Nechvatal, Miles E. Smid, Elaine B. Barker, Stefan D. Leigh, Mark Levenson, Mark Vangel, David L. Banks, Nathanael Alan Heckert, James F. Dray, and San Vo. Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, Gaithersburg, MD, United States, 2010.

[6] Robert G Brown, Dirk Eddelbuettel, and David Bauer. Dieharder: A random number test suite. *Open Source software library, under development*, 2017.

[7] Ian Goldberg and David Wagner. Randomness and the netscape browser. *Dr Dobb's Journal-Software Tools for the Professional Programmer*, 21(1):66–71, 1996.

[8] Mads Haahr. Random. org: True random number service. *School of Computer Science and Statistics, Trinity College, Dublin, Ireland. Website (http://www. random. org). Accessed*, 1, 2017.

[9] John Kelsey, Kerry A. McKay, and Meltem Sönmez Turan. Predictive models for min-entropy estimation. *IACR Cryptology ePrint Archive*, 2015:600, 2015.

[10] Pierre L'Ecuyer and Richard Simard. Testu01: A c library for empirical testing of random number generators. *ACM Trans. Math. Softw.*, 33(4):22:1–22:40, August 2007.

[11] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, January 2001.

[12] Scott Yilek, Eric Rescorla, Hovav Shacham, Brandon Enright, and Stefan Savage. When private keys are public: results from the 2008 debian openssl vulnerability. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 15–27. ACM, 2009.