Table Of Contents

Booz | Allen | Hamilton

1.1 Introduction to Entropy

- The "Entropy Problem" is a serious problem for Vendors, Labs, NIST, NIAP and others
- Entropy is used to seed Deterministic Random Bit Generators (DRBGs)
- DRBGs are used to generate keys
- We need to have confidence keys are generated with the claimed (maximum) entropy
- You don't want to be the Vendor in the news because
  - Your keys can be trivially recovered on a laptop
- You don't want to be the Lab in the news because
  - You tested the product of the vendor in the news
- NIST and NIAP are probably protected from being in the news – lets go work for them ! ☺

1.2 Introduction to Entropy

- Define Shannon Entropy and Min Entropy
- Given a discrete random variable X with possible values $\{x_1, x_2 \ldots, x_n\}$ and with probabilities $p_1, p_2 \ldots p_n$ the entropy of X is defined as
- $\mathrm{H}(X) = -\sum_{i=1}^{n} p_i \log_2 p_i \geq 0$
- The Min-entropy is defined as
- $H_\infty = min\{-\log_2 p_1, -\log_2 p_2 \ldots, -\log_2 p_n\} = -\log_2 \max p_i$ (Note: -log is a decreasing function)
- Note: $0 \leq H_\infty \leq H(X)$

2.1 Unbiasing

- An entropy source may produce a biased stream.
- Given a biased input stream is it possible to derive an unbiased output stream with better entropy properties?
  - "Various techniques used in connection with random digits" (J. von Neumann, 1951)
  - "Independent unbiased coin flips from a correlated biased source: a finite state Markov chain" (M. Blum 1986)
  - "Iterating von Neumann's procedure for extracting random bits" (Y. Peres, 1992)
  - "Efficient Generation of Random Bits from Finite State Markov Chains" (H. Zhou and J. Bruck 2011)

2.2 Unbiasing
von Neumann Unbiasing

- Von Neumann unbiasing (1951)
- Simple scheme to generate a <u>shorter</u> unbiased stream from a biased one
- <u>Assume</u> random biased bits are produced <u>independently</u> of each other
- Prob (0) = p, Prob (1) = 1-p = q

Input String	Probability	Output String
00	p^2	Nothing
01	pq	0
10	qp	1
11	q^2	nothing

- Consider biased input string 00 01 10 11 01 11 11 10 => 0101 (unbiased output stream)

2.3 Unbiasing
Peres Unbiasing

- Note: In any unbiasing method length of output stream is limited by the entropy of the input stream.

- Problems with the von Neumann method:
  - Assumes input bits are produced independently
  - Output stream is considerably shorter than input (input length = n, output length~npq)
  - Wastes 00 and 11 bits

- Peres method (1992) iterates recursively the von Neumann method
  - Takes advantage of the 00 and 11 bits by forming new streams to generate extra bits
  - The rate at which unbiased bits are output ~ entropy of the input stream (Optimally Efficient)

2.4 Unbiasing
Peres Unbiasing

- Peres unbiasing example (ref: Markov Chains and Mixing Times, Levin, Peres, Wilmer, p312):

Input Bits	00	11	01	01	10	00	10	10	11	10	01	Extracted Bits
Extracted unbiased bits A	.	.	0	0	1	.	1	1	.	1	0	0011110
Discarded bits	0	1	.	.	.	0	.	.	1	.	.	
Extracted unbiased bits B	.	0	.	.	.	.	.	.	0	.	.	00
XORed bits	0	0	1	1	1	0	1	1	0	1	1	
Extracted unbiased bits C	.	.	.	.	.	1	.	.	.	0	.	10

- Extracted unbiased bit stream: 00111100010
- Problem: Assumes input bits are independent
- Can we do any better than this?

2.5 Unbiasing
Markov Chain Unbiasing

- Zhou and Bruck reviewed the previous ideas of von Neumann, Blum and Peres
  - Create three new algorithms that improve upon previous ones
  - Processes correlated input streams from a Markov Chain
  - Produces an unbiased output stream
  - Optimal information efficiency
  - See their paper for the details

- Other ideas?
  - "A Comparison of Post-processing Techniques for Biased Random Number Generators" (Kwok, Ee, Chew, Zhenk, Khoo, Tan 2011) Compares Von Neumann method against using Linear Codes

3.1 Conditioning

- Conditioning is taking the raw entropy as an input to a function and using the output of the function as the conditioned source of entropy
- Properties:
  - Smooths out the roughness in the raw entropy source
  - Does not increase the entropy
- Example conditioning functions:
  - Hash Functions
  - Linear Feedback Shift Registers (LFSRs)

3.2 Conditioning – Hash Functions

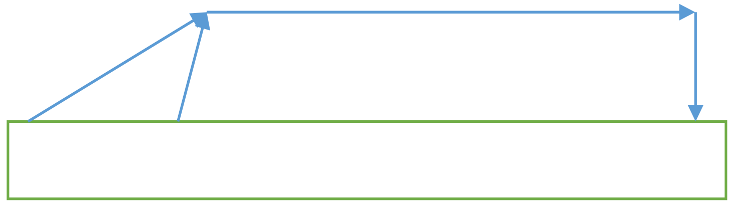
- A hash function is a random function
- What can we say about the relationship between the entropy of the input and the entropy of the output?
- Examples include SHA-256

Case 1 Input entropy is < 256 bits	output entropy is < 256 bits
Case 2 Input entropy input = 256 bits	output entropy is <= 256 bits
Case 2 Input entropy is > 256 bits	output entropy is <= 256 bits

- Can we say more than this? Open research problem.

3.3 Conditioning – LFSRs

- LFSRs are linear devices
- Consider a 32 bit LFSR
  - 32 bit non-zero initial fill
  - Primitive polynomial is used to step out the LFSR (Primitive means irreducible and maximal cycle)



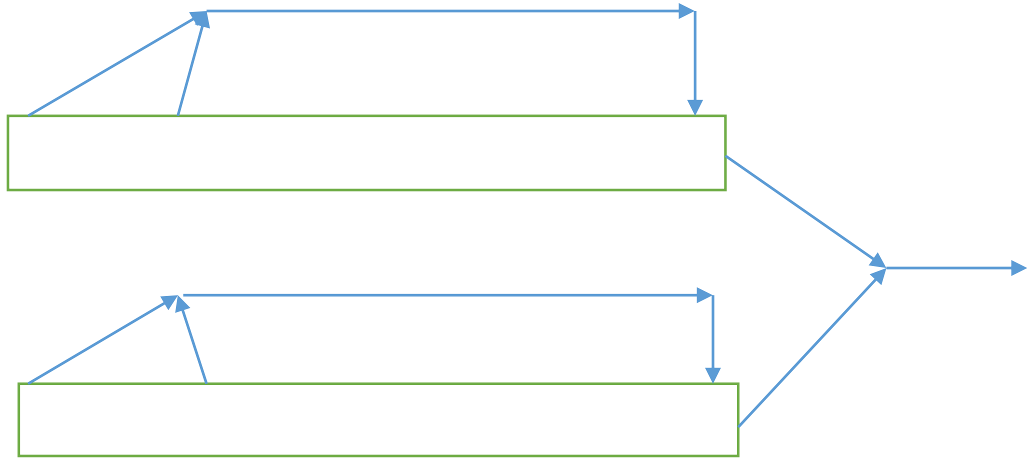
- e.g. $P(x)=x^0 + x^5 + x^{32}$
- (Note this example polynomial may not be irreducible or primitive)
- Maximal cycle length is $2^{32} - 1$

3.4 Conditioning – LFSRs

- LFSR example (contd)
- Suppose the initial fill has 32 bits of entropy taken from a raw entropy source
  - Step out the LFSR 1,000,000 steps
  - Read off 256 successive bits as the key
- How much entropy do these 256 bits have?
- Answer: 32
- The first 32 bits of the 256 bits can be used to generate the rest of the 256 bit key
- There are only $2\uparrow 32 - 1$ possible sets of 256 bits
- Not a very good idea!

3.5 Conditioning – LFSRs

- Next consider two LFSRs whose output is added (XOR) together.
  - LFSR1 is driven by $P1(x)$
  - LFSR2 is driven by $P2(x)$
- What polynomial $Q(x)$ does the output stream satisfy? Answer: $\mathrm{Q}(\mathrm{x})=LCM(P1(x), P2(x))$
- If $P1(x)=P2(x)$ then the output stream is simply satisfied by $\mathrm{Q}(x)=P1(x)$
- If $P1(x)\ and\ P2(x)$ are both primitive and different then $Q(x)=P1(x)*P2(x)$



3.6 Conditioning – LFSRs

- LFSR example (contd)
- Suppose each initial fill has 32 bits of entropy taken from a raw entropy source
  - Step out each LFSR 1,000,000 steps
  - Read off 256 successive bits of the summation stream as the key
- How much entropy do these 256 bits have?
- Answer: close to 64 bits
- The first 64 bits of the 256 bits can be used to generate the rest of the 256 bit key using $Q(x)=LCM(P1(x),P2(x))$
- As can be seen you need 8 32-bit LFSRs to get close to 256 bits of entropy. You must use at least 8 different primitive polynomials.

4.1 What is the Entropy Estimation Problem?

- Given an entropy source with discrete random variable X with possible values $\{x_1, x_2 \ldots, x_n\}$ and with probabilities $p_1, p_2 \ldots p_n$ the Entropy Estimation Problem is estimate the entropy $H(X)$ or min-entropy $H_\infty(X)$ from a data sample $s_1\ s_2 \ldots s_T$

- Given $s_1\ s_2 \ldots s_T$ we can calculate estimated probabilities $\{p_1, p_2, \ldots, p_n\}$ which are the maximum likelihood estimators (MLEs) also known as plug-in estimators.

- Then the entropy estimate is given by:

- $H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$

- The Min-entropy estimate is given by:

- H_∞ (X)$= min\{-\log_2 p_1, -\log_2 p_2 \ldots, -\log_2 p_n\}$

Booz | Allen | Hamilton

4.2 What is the Entropy Estimation Problem?

- Example:
- Random Variable X takes $n=2$ values {0,1} with probabilities $p↓1 =$0.25, $p↓2 =$0.75 respectively then
- $H(X)=0.8113$
- $H↓\infty (X)=0.4150$
- Let $T=10$
- Data sample =1101110111 then $p ↓1 =2/10$ $p ↓2 =8/10$ are the MLEs or plug-in values
- $H (X)=0.7219$
- $H ↓\infty (X)=0.3219$

4.3 What is the Entropy Estimation Problem? Properties of $\boldsymbol{H}\,(\boldsymbol{X})$

- Good Properties of estimators $S\,(X)$ (aka statistics)

  Unbiased: $E(S\,(X))$=Average value of $S=S(X)$

  e.g. repeat the entropy estimation experiment many times and compute the average.

- Is $H\,(X)$ unbiased? Answer=No!

  E($H\,(X)) \leq H(X)$

- The entropy estimator using plug-in values under-estimates the true entropy value

- In fact:

  $H \downarrow MM\,(X) = H\,(X) + (\mathrm{n}-1)/2\mathrm{T}$ is a better estimator of the entropy (MM=Miller-Madow)

  No unbiased estimator of entropy exists for discrete case

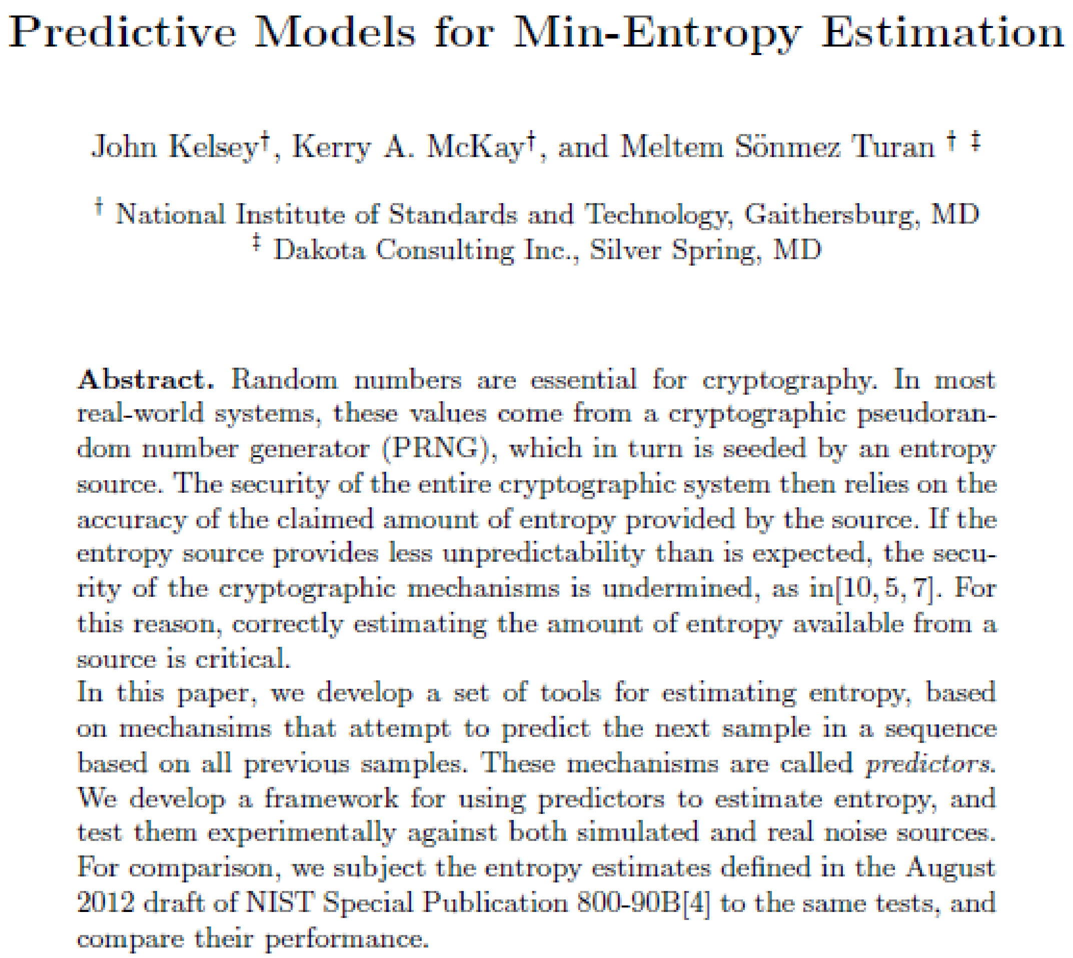
- Ref: “Estimation of Entropy and Mutual Information”, Liam Paninski (2003)

4.4 What is the Entropy Estimation Problem? Properties of $\boldsymbol{H \downarrow \infty \, (X)}$

- Is $H \downarrow \infty \, (X)$ unbiased?
- Answer – Open Research Problem

5. Other Research

- IACR- CHES 2015



Predictive Models for Min-Entropy Estimation

John Kelsey[†], Kerry A. McKay[†], and Meltem Sönmez Turan [†] [‡]

[†] National Institute of Standards and Technology, Gaithersburg, MD
[‡] Dakota Consulting Inc., Silver Spring, MD

Abstract. Random numbers are essential for cryptography. In most real-world systems, these values come from a cryptographic pseudorandom number generator (PRNG), which in turn is seeded by an entropy source. The security of the entire cryptographic system then relies on the accuracy of the claimed amount of entropy provided by the source. If the entropy source provides less unpredictability than is expected, the security of the cryptographic mechanisms is undermined, as in[10, 5, 7]. For this reason, correctly estimating the amount of entropy available from a source is critical.
In this paper, we develop a set of tools for estimating entropy, based on mechansims that attempt to predict the next sample in a sequence based on all previous samples. These mechanisms are called *predictors*. We develop a framework for using predictors to estimate entropy, and test them experimentally against both simulated and real noise sources. For comparison, we subject the entropy estimates defined in the August 2012 draft of NIST Special Publication 800-90B[4] to the same tests, and compare their performance.

6.1 NIST Entropy Tool

- NIST Entropy tool
- Estimates the min entropy of a data stream
- Case 1 tests for iid and then performs entropy estimation calculations
- otherwise
- Case 2 if data is non-iid performs other entropy estimation calculations

6.2 NIST Entropy Tool - iid

- Python iid_main.py [-h] [-v] datafile bits_per_symbol number_of_shuffles
- IId (Independent Identically Distributed) calculations
  - Compression Test
  - Over / Under Test
  - Excursion Test
  - Directional Runs Test
  - Covariance Test
  - Collision Test
  - iid shuffle test
  - Chi square independence test
  - Chi square stability test
- If the data is IID then calculates the Min Entropy value of the data stream
- Performs some sanity tests: Compression and Collision Tests

6.3 NIST Entropy Tool – non-iid

- Python noniid_main.py [-h] [-u usebits] [-v] datafile bits_per_symbol
- Non-iid calculations
  - Collision Test
  - Partial Collection Test
  - Markov Test
  - Compression Test
  - Frequency Test
- Estimates Min Entropy of the data sample
- Performs a sanity check
  - Compression Test
  - Collision Test

7.1 Example of Test Results for Raw Entropy Samples

- Note: The following results are purely scientific results. Booz Allen Hamilton is not endorsing these entropy sources.
- We ran the NIST entropy tool against the following three data samples
- 1. NIST Random Beacon
- 2. ID Quantique
  - Quantum Random Number Generator (QRNG) called Quantis
  - Quantum optical process based upon photons being randomly reflected or transmitted
- 3. Quintessence Labs
  - Quantum Random Number Generator (QRNG) called qStream
  - Quantum properties of light

7.2 Test Results of Data Sample 1

- Data Sample 1 = Seed data from NIST Random Beacon
- Seed data is the XOR of the output of two commercial entropy sources
- NIST Random Beacon produces a series of 512 bit seed data samples
- 512 bits seed data samples were concatenated to form 1,000,000 byte input steam to NIST entropy tool
- Data was treated as 8 bit bytes
- Results:
- Data was confirmed as IID
- Min Entropy Estimate = 7.8852 bits (out of 8)
- NIST is currently developing a quantum random source

Booz | Allen | Hamilton

7.3 Test Results of Data Sample 2

- ID Quantique
- Company provided a conditioned entropy source produced by properties of quantum mechanics - Quantis
- Data Sample 2
- 1,000,000 byte input steam to NIST entropy tool
- Data was treated as 8 bit bytes
- Results:
- Data confirmed to be IID
- Min Entropy Estimate = 7.8943 bits (out of 8)

7.4 Test Results of Data Sample 3

- Quintessence Labs
- Company provided a conditioned entropy source produced by properties of quantum mechanics - qStream
- Data Sample 3
- 1,000,000 byte input steam to NIST entropy tool
- Data was treated as 8 bit bytes
- Results:
- Data confirmed to be IID
- Min Entropy Estimate = 7.8957 bits (out of 8)

8. Improvements to the NIST Entropy Scheme

- 1. Big Data (fast) entropy estimation – no limits on how much data is sampled – special purpose devices for fast calculations or on the symbol size
- 2. All entropy source designs should be available in the public domain and be non-proprietary, like cryptographic algorithm designs
- 3. There should be an international competition to design and develop the best entropy source for commercial cryptographic applications like there was for AES and SHA-3 algorithms.

9.1 References

- Company Websites:
- http://www.nist.gov/itl/csd/ct/nist_beacon.cfm
- http://www.quintessencelabs.com/
- http://www.idquantique.com/
- Papers:
  - “Various techniques used in connection with random digits” (J. von Neumann, 1951)
  - “Independent unbiased coin flips from a correlated biased source: a finite state Markov chain” (M. Blum 1986)
  - “Iterating von Neumann’s procedure for extracting random bits” (Y. Peres, 1992)
  - “Efficient Generation of Random Bits from Finite State Markov Chains” (H. Zhou and J. Bruck 2011)
  - “A Comparison of Post-processing Techniques for Biased Random Number Generators” (Kwok, Ee, Chew, Zhenk, Khoo, Tan 2011) Compares Von Neumann method against using Linear Codes

9.2 References

- “Estimation of Entropy and Mutual Information”, Liam Paninski (2003)
- NIST entropy tool https://github.com/usnistgov/SP800-90B_EntropyAssessment
- “Predictive Models for Min-Entropy Estimation”, Kelsey, McKay, Turan (2015)
- “Entropy Bounds and Statistical Tests”, Hagerty and Draper (2012)

10. Questions?

- David Cornwell, PhD
- Booz Allen Hamilton
- Cornwell_david@bah.com
- 410 684 6579

Booz | Allen | Hamilton