Entropy Sources and You:

An Overview of SP 800-90B

John Kelsey, NIST

RNG Workshop--Dec 2012

The Big Picture

- Random bits are needed for everything in cryptography.
- Deterministic mechanisms for generating pseudorandom bits are useful, but need some truly unguessable seed.
- Getting this seed is the point of SP 800-90B

SP 800-90 is About Random Bits

- 800-90A is about deterministic algorithms
  - DRBG = Cryptographic PRNG
- -90B is about getting unpredictable inputs for DRBGs
- -90C is about putting pieces together

800-90B: Entropy Sources

- ES yields bit strings with an entropy assessment
- This allows seeding of DRBG, and generation of full-entropy outputs
- Requirement: We must be able to trust entropy assessments, and to add them together safely.
  - *To seed a DRBG that needs 192 bits entropy, we can concatenate 32 entropy source outputs, each assessed at 6 bits*.

Full Entropy Source

- Idea: Entropy source whose outputs can be considered indistinguishable from random
- Useful for making NRBGS (aka true random generators,like/dev/random is often supposed to work.)
- Two paths:
  - Cryptographic conditioning
  - Raw bits indistinguishable from random

GOAL: Good entropy sources in widespread use

- Design guidance
- Meaningful validation testing by labs
- Health tests to detect failures

This is currently a problem in real-world systems!

Entropy Source =

- Noise source = where the raw, somewhat unpredictable bits come from
- Conditioning Component = optional stuff done to outputs to improve their statistical properties
- Health Tests = ongoing simple tests of noise source outputs to detect major failures in the field

What do we mean by "entropy?"

- Entropy is how we quantify unpredictability.
- Min-entropy used throughout 800-90
  - -lg(P[max]) = base 2 log of maximum probability
  - Intuitively: *What is the highest probability anyone might have of predicting this output, given all other outputs?*
    - *Most likely output has prob 1/4: 2 bits of min-entropy*

How will labs assess an entropy source?

- *Noise source* needs some justification
- Use *statistical tests* to assess entropy per sample
  - *Requires access to raw noise source bits*
- Requirements on *conditioning component*
- Ongoing *health tests* on device

Noise Source

- Some unpredictable process
- Measurements and digitization
- This is where the unpredictability comes from
- Example: Ring oscillator
- Requirement: Justify unpredictability

Assessing entropy

- Draw lots of raw samples from noise source
- EITHER:
  - iid tests -> assessed entropy per sample
  - non-iid tests -> assessed entropy per sample
- THEN:
  - Sanity checks against assessed entropy per sample

Access To Raw Bits

- Tests have to be run on lots of unprocessed bits (a million or so)
- Need *unprocessed* bits
  - Some fuzziness in that definition
  - Example: XORed together Ring Oscillator
  - Are there simple ways of processing bits that should be allowed? (Ex: Von Neumann Unbiasing?)
- Access to raw bits can be problematic
  - Sparse, intermittent, etc.

IID Tests:Overview

- Several tests to try to disprove claim that source is iid
  - Shuffling tests
  - Chi-Square tests of stability and independence
- If source appears to be iid, we can assess entropy by counting most common sample value.

Shuffling Tests

- If source is iid,then a shuffled (permuted) version of the sequence of samples was just as likely as original sequence of samples

- Shuffling tests compute several statistics on original sequence, and then on 1,000 shuffled versions of sequence.

- Question: Is original sequence an outlier?

Non-iid test overview

- Most sources aren't iid
  - Though they may pass iid tests when sparsely sampled
- Try many tests.
  - Each yields a fairly conservative entropy estimate
  - Lowest entropy estimate wins

Sanity Checks

- iid and non-iid tests provide an entropy estimate
- Sanity checks make sure something hasn't gone horribly wrong
    - Compression test
    - Collision test
- When sanity checks fail, source fails
    - Our methods for estimating entropy have failed here!

Lots of discussion later on validation testing

- What important failures are we missing?
- What good sources will be hard to validate?
- Restart testing
- Data collection issues
- Future changes to tests

Conditioning

- Conditioning is an attempt to improve the statistical properties of output bits
  - Example: Collecting 32 low-quality bits from ring oscillator, condensing into one high-quality bit.
  - Example: Feeding samples into SHA256 and taking the output of the hash.

Entropy Arithmetic

- Basic insight: Conditioning can only keep or lose entropy, never gain it.
- Entropy in output = MIN of:
  - Entropy in input minus any losses from collisions
  - Output size and internal state size of conditioner
  - Assessed entropy in output
- Our math in SP 800-90B is messed up
  - We'll discuss how to fix it later today.

Full Entropy Sources

- For full entropy sources, a common rule of thumb is that 2n bits of entropy in input gives an n-bit output indistinguishable from random
- Unless we can see a better way to do things, we will stick with this rule for full entropy sources

Approved Conditioning Functions

- **Approved hash function**
  - Used directly
  - HMAC
    - Hash-df from SP 800-90 A
- **Approved block cipher** used with either:
  - CMAC
  - CBC-Mac as specified in document
  - BC-df from SP 800-90A

Why use an approved function?

- No further testing of outputs needed -- just entropy arithmetic
- Allowable for use in building Full Entropy Source
- Giving people good schemes may keep everyone from rolling his own

Unapproved Conditioning Function

- Anything not on our list
- Requirement: outputs must pass iid tests
- Entropy assessment = MIN of:
  - Assessed input entropy less losses from collisions
  - Output and internal state size
  - Assessed output entropy from iid tests and sanity checks.

Should we have another category?

- XOR sum of many outputs
- Von Neumann unbiasing
  - Generalization for nonbinary values?
- Outputs could be considered raw bits in validation testing
- These could be allowed for full entropy sources

Health Tests

- Noise sources seem inherently fragile
- Continuous health tests provide a "sanity check" on noise source during operation
  - Tests are run continuously, on bits that are output and used.
  - This introduces a very small bias (some sequences will cause test to fail)
  - Can't tolerate high false positive rate, so tests aren't too powerful.

Current Scheme

- Two required tests
  - Repetition count test
  - Adaptive proportion test
- Alternative: Demonstrate equivalent functionality

Repetition Count Test

- Sample assessed at H bits of min-entropy
- Most probable value for next sample value is no more than:
  - P[max] = 2^{-H}
- So, choose smallest integer C s.t.
  - P[max]^{C-1} < 2^{-30}
- If we see C repetitions of same value, we flag a failure.

Adaptive Proportion Test

- P[max] gives us a bound B on how many times any sample value should appear in a window of N samples.
- Every N+1 samples:
  - Take next sample as thing to be counted
  - Count how often it occurs in next N samples
  - If count > B then flag an error

Memory requirements

- Adaptive proportion test requires only three variables:
    - Where we are in N+1 samples
    - What current value being counted is
    - Current count
- Recall: this runs continuously

We will discuss these tests later today

- How well do they deal with likely failures?
- How hard are they to implement?
- What about grandfathering existing designs?

Equivalent functionality

- Some designs have their own continuous tests
- Probably much better for their source--designers know how their source is likely to fail.
- Equivalent functionality testing:
  - Allow designer to show that their continuous health tests will detect the same problems required tests detect.
  - If so, they don't have to implement ours.

Startup Testing

- FIPS 140 has a history of startup testing requirements
  - Known answer tests included.
  - Not too clear what the benefit of these is.
- Statistical startup tests on noise source seem more valuable
  - Requirement: Run continuous tests on output bits for at least one window of adaptive proportion test before using bits.
  - Discuss performance implications of this later.

Summary

- Entropy Source =
  - Noise Source
  - (optional) Conditioning Component
  - Health Tests
- Each part has design and documentation requirements
- Each part also subject to lab tests for validation.