Vendor CHTs

David Johnston
dj.johnston@intel.com

For : NIST RBG Workshop. May 31st – June 1st 2023
<table>
<thead>
<tr>
<th>SCC</th>
<th>+ &gt; 0.62</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min Entropy &lt; 0.3</td>
<td></td>
</tr>
<tr>
<td>Fail point for 4x</td>
<td></td>
</tr>
<tr>
<td>Extraction ratio</td>
<td></td>
</tr>
</tbody>
</table>

Vendor CHT

Pass Region

Pass/Fail Transition Region
Minimum Input Entropy to Conditioner
6X Extraction Ratio

CHT Pass/Fail Threshold
Chosen for 100% fail at 0.2

Zone of Janky Silicon

Non-IID Assessed Entropy

Actual Entropy
4.5  

1. The continuous tests **shall** include either:
   
a. The **approved** continuous health tests, described in Section 4.4, or
   
b. Some developer-defined tests that meet the requirements for a substitution of those approved tests, as described in Section 4.5. If developer-defined health tests are used in place of any of the approved health tests, the tester **shall** verify that the implemented tests detect the failure conditions detected by the approved continuous health tests, as described in Section 4.4. The need to use the two approved continuous health tests can be avoided by providing convincing evidence that the failure being considered will be reliably detected by the developer-defined continuous tests. This evidence may be a proof or the results of statistical simulations.
   
c. The continuous tests **may include** additional tests defined by the developer.

- **RCT**  
- **APT**  

**Inadequate for negative SCC based failure**

- **vCHT**  

**Vendor CHT must catch what the RCT and APT Cannot**
+ also meet 4.5a & 4.5b

- **vCHT**  
- **RCT**  
- **APT**  

**Vendor CHT must catch what the RCT and APT Cannot**
+ also meet 4.5a & 4.5b
+ also catch what RCT and APT catch
4.5 Developer-Defined Alternatives to the Continuous Health Tests

a. If a single value appears more than $\lceil 100/H \rceil$ consecutive times in a row in the sequence of noise source samples, the test shall detect this with a probability of at least 99%.

- If the CHTs are required to detect all failures the RCT would detect why is this requirement on a vendor CHT?
- A vendor CHT is there to catch the cases that the RCT and APT cannot catch and replace the RCT and APT if not present. Thus the RCT capture set is mandatory.
- The capturing runs of $\lceil 100/H \rceil$ bits is like an RCT with $C=\lceil 100/H \rceil$. The RCT’s $C$ is always smaller and is going to detect this condition 100% of the time.

<table>
<thead>
<tr>
<th>$H$</th>
<th>RCT $C$</th>
<th>$\lceil 100/H \rceil$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>68</td>
<td>1000</td>
</tr>
<tr>
<td>0.2</td>
<td>35</td>
<td>500</td>
</tr>
<tr>
<td>0.3</td>
<td>24</td>
<td>334</td>
</tr>
<tr>
<td>0.4</td>
<td>18</td>
<td>250</td>
</tr>
<tr>
<td>0.5</td>
<td>15</td>
<td>200</td>
</tr>
<tr>
<td>0.6</td>
<td>13</td>
<td>167</td>
</tr>
<tr>
<td>0.7</td>
<td>11</td>
<td>143</td>
</tr>
<tr>
<td>0.8</td>
<td>10</td>
<td>125</td>
</tr>
<tr>
<td>0.9</td>
<td>9</td>
<td>112</td>
</tr>
</tbody>
</table>
Proposal for 4.5 part a

• (1) Instead of requiring a vendor CHT to do it, amend the RCT definition to require C is at least small enough to detect a repeated sequence of $\lceil 100/H \rceil$ symbols. This will be a no-op for any binary RCT since it already meets this requirement.

• (2) Remove 4.5 part a.

• (3) or just do (2) since (1) is moot
4.4.1 Repetition Count Test

Given the assessed min-entropy \( H \) of a noise source,

4.5 Developer-Defined Alternatives to the Continuous Health Tests

b. Let \( P = 2^{-H} \). If the noise source's behavior changes so that the probability of observing a specific sample value increases to at least \( P^* = 2^{-H/2} \), then the test **shall** detect this change with a probability of at least 50% when examining 50,000 consecutive samples from this degraded source.
Proposal for 4.5 part b

• The current rule make a threshold that is a function of how good the noise source is, rather than a function of the failure point. Instead make it a statement that the test will almost always fail at the lowest tolerable entropy rate and make it valid over the number of bits that need to be tested before they go into the conditioner. I.E. n_in.

• E.G. “If the source entropy rate falls below the minimum required entropy rate into the conditioning chain to achieve full entropy at the conditioning chain output, the vendor defined test shall detect this with probability greater than 1-(10E-6) over n_in bits to the conditioning chain.”

• This means that the output of the conditioner chain will be full entropy and the vendor can use an honest H_submitter value rather than an artificially low number.
Proposal for RCT 4.4.1 & APT 4.4.2

Given the assessed min-entropy $H$ of a noise source, the probability $p$ of that source generating $n$ identical samples consecutively is at most $2^{-H(n-1)}$. The test declares an error if a sample is repeated $C$ or more times. The cutoff value $C$ is determined by the acceptable false-positive probability $\alpha$ and the entropy estimate $\hat{H}$ using the following formula:

- The RCT text calls for $H$ to be set to the assessed entropy.
- The assessed entropy is the wrong place to set a test threshold.
- The specification is ambiguous as to whether the $H$ used in the RCT is the same $H$ used in the APT and in vendor defined CHTs.
- $H$ should be set per test based on the properties of the test and the optimum cutoff points for a test given the expected distribution of entropy rate from a functioning source and the lowest tolerable entropy rate.
- Change “the assessed min-entropy $H$” to “a chosen test cutoff entropy $H_{rct}$” and similarly use $H_{apt}$ in 4.4.2.
- The false positive error rate should be close to 100% at the minimum tolerable entropy rate.
• Since we have to use the results of ea_non_iid in our CHT parameters (E.G. in 4.4.1) we care that these results are realistic. In many cases, they are not.

• In this synthetic test 7800 samples of entropy data were input to the test at entropy levels from 0.025 to 0.975 in 0.025 increments. The actual entropy, the bias, scc and H_original, along with the estimation error and the test with the lowest estimate were recorded for each data point. 200 runs per entropy level. 39 entropy levels.

• ea_non_iid -i -v -t -l 0,1000000 <filename> 1 was used.

• The next 4 plots give the set of points where a specific test gave the lowest estimate.
• The lonely MultiMWC Estimate
• 1 case in 7800
• The tTuple estimate only gives the lowest estimate when entropy is below 0.2ish
• 339 cases in 7800
• If you are claiming entropy of 0.1 you can be judged as having entropy of 0.01 by the tTuple test.
• The collision test contributes the lowest estimate in about 18% of cases, when entropy > 0.1
• The underestimation is largest around entropy from 0.6 to 0.8
• 1434 out of 7800
• Winner Winner! Chicken Dinner!
• The tCompression test contributes the lowest estimate for 77.2% of cases across the full entropy range.
• The underestimation is largest at entropy 0.65
• 6026 out of 7800
<table>
<thead>
<tr>
<th>alg_idx</th>
<th>Count</th>
<th>Test Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>Most Common Value</td>
</tr>
<tr>
<td>2</td>
<td>1434</td>
<td>Collision Test Estimate</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>Markov Test Estimate</td>
</tr>
<tr>
<td>4</td>
<td>6026</td>
<td>tCompression Test Estimate</td>
</tr>
<tr>
<td>5</td>
<td>339</td>
<td>T-Tuple Test Estimate</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>LRS Test Estimate</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Multi Most Common in Window (MultiMWC) Prediction Test Estimate</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>Lag Prediction Estimate</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>Multi Markov Model with Counting (MultiMMC) Prediction Test Estimate</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>LZ78Y Prediction Test Estimate</td>
</tr>
</tbody>
</table>

**Colors**

- Collision: Black
- TCompression: Green
- T-Tuple: Yellow
- MultiMWC: Red
Average Err: -0.15339
largest Err: -0.44592
Smallest Err: -0.00334
median Err: -0.15689

tCompression total: -911.636
tCompression count: 6026

tCompression Av err: -0.15128
smallest tC err: -0.00352
largest tC Err: -0.44592

not tCompression total: -238.804
not tCompression count: 1774

not Tc Av Err: -0.13461
smallest not Tc Err: -0.00334
largest not Tc Err: -0.27637

Over all 7800 tests

Over tests where tCompression gave the lowest estimate

Over tests where any other test gave the lowest estimate
Maurer’s Universal Statistical Test Abstract:
“A new statistical test for random bit generators is presented which, in contrast to presently used statistical tests, is universal in the sense that it can detect any significant deviation of a device's output statistics from the statistics of a truly random bit source when the device can be modeled as an ergodic stationary source with finite memory but arbitrary (unknown) state transition probabilities. The test parameter is closely related to the device's per-bit entropy which is shown to be the correct quality measure for a secret-key source in a cryptographic application. The test hence measures the cryptographic badness of a device's possible defect. The test is easy to implement and very fast and thus well suited for practical applications. A sample program listing is provided.”

The assumption of ergodicity and stationarity does not hold for physical sources.

The Maurer’s Universal Statistic is questionable for physical sources.

Also, the lower bound adjustment is based on unsound assumptions:

Comments on Cryptographic Entropy Measurement
Anna M. Johnston
Juniper Networks
amj@juniper.net
October 30, 2019

“The central limit theorem does not hold and confidence intervals are not well defined for $H_\infty$.”

Proposal: Remove SP800-90B 6.3.4 (Compression Estimate)
Some Statistics!
Simplifying The SCC Equation for Binary Data

Start with

\[
SCC = \frac{n \left( \sum_{i=0}^{n-1} x_i x_{(i+1) \mod n} \right) - \left( \sum_{i=0}^{n-1} x_i \right)^2}{n \left( \sum_{i=0}^{n-1} x_i^2 \right) - \left( \sum_{i=0}^{n-1} x_i \right)^2}
\]

\(n\) is the number of bits. We can fix this for the test HW.

\[
\left( \sum_{i=0}^{n-1} x_i x_{(i+1) \mod n} \right)
\]

0*0=0
0*1=0
1*0=0
1*1=1

So just count occurrences of 11

Reduce degrees of freedom by 1 to eliminate the mod \(n\).

End With

\[
sc = \frac{(n - 1) \text{count11} - \text{count1}^2}{(n - 1) \text{count1} - \text{count1}^2}
\]

Count occurrences of 1

When \(x_i = 1\) or 0, \(x_i = (x_i)^2\)

So is identical to above count.
\[ scc = \frac{(n - 1) \text{count11} - \text{count1}^2}{(n - 1) \text{count1} - \text{count1}^2} \]

Rearrange for 11s and 1s

Test puts bounds on 11s and 1s counts

What entropy is the test testing for?
• The mean and SCC are separate things that can be tested for (efficiently in hardware)

• But they are related in ways that make swoopy lines when plotted against actual entropy levels as a function of mean and SCC making it hard to test for entropy levels

• Let’s use a really simple Markov models to link the two....
A two state Markov model has two degrees of freedom.

- \( P_{01} + P_{00} = 1 \)
- \( P_{11} + P_{10} = 1 \)
- As a generator, the \( P_{01}, P_{10} \) pair can be set to generate data with any combination of bias and SCC using the equations below

\[
\mu = \frac{P_{01}}{P_{10} + P_{01}} \\
SCC = 1 - P_{10} - P_{01} \\
P_{10} = (1 - \mu)(1 - SCC) \\
P_{01} = \mu(1 - SCC)
\]
Onto Entropy!
Unexpected MCVs turned out to be real

Each point is the most common 4 bit symbol in 1MiB of data generated from the Markov model

E.G. 101 is the most probable two step path. But $P_{11} > P_{10}$

$(P_{10} \times P_{01}) \times P_{11}$
The Markov model can be used to generate data with known entropy. By fitting to the model, the min entropy can be measured.

djent will give you Markov parameters from data
markov2p.py will compute entropy from Markov parameters

\[ H_{\infty}(X) = -\log_2(P(MCV)) \]
$ djenrandom -m markov_2_param --entropy=0.6 -k 1024 -b -s > file.bin

$ djent -b file.bin
opening file.bin as binary
Symbol Size(bits) = 1
   Min Entropy (by max occurrence of symbol 0) = 0.601632
   Analysing 8388600 1-bit symbols
   Shannon IID Entropy = 0.925765 bits per symbol
   Optimal compression would compress by 7.423505 percent
   Chi square: symbol count=8388601, distribution=848372.77, randomly exceeds 0.00 percent of the time
   Mean = 0.340992
   Monte Carlo value for Pi is 3.798721 (error 20.92 percent).
   Serial Correlation = 0.002939
   Longest Run Symbol = 0. Run Length = 38
   Probability of longest run being <= 38 = 0.999985
   Position of Longest Run = 892812 (0xd9f8c). Byte position 111601 (0x1b3f1)
   A 2 state Markov generator with transition probabilities P01=0.339990, P10=0.657071 would generate data with the same mean and serial correlation

$ python3
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 26 2018, 23:26:24)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import markov2p
>>> er,_,mcv = markov2p.p_to_entropy(p01=0.339990, p10=0.657071, bitwidth=8)
>>> er
0.5997142477854777
>>> mcv
0
• Entropy vs $p_{10}, p_{01}$

• Swoopy lines. Not great for making online entropy tests
• Min Entropy vs mean,scc

• Again, swoopy lines, OK for testing with bounds on P1,P11 as long as mean is close to 0.5 but loses selectivity if mean varies more.
• Plotting entropy against $p_1, p_{11}$ normalized to (0,1) interval shows mostly straight lines. The lower edge of each iso-entropy line is convex, meaning when approximated to a straight line, it is conservative, excluding good points, rather than including bad points.
Test For Points Within an Entropy Contour

- Red region covers set of points with entropy rate 0.4 or higher.
Straight Edges Allow Linear Comparison Tests

\[ \begin{align*}
AB & : \quad pc11 < \frac{4}{3}pc1 - \frac{4}{15} \\
BC & : \quad pc11 < \frac{2}{3}pc1 - \frac{1}{15} \\
DC & : \quad pc11 > 2pc1 - 1 \\
ED & : \quad pc11 > pc1 - 0.4
\end{align*} \]
Make The Coefficients Integer and Scale by $n-1$

\begin{align*}
\text{AB} & : 15pc11 < 15pc1 - 4 \\
\text{BC} & : 15pc11 < 10pc1 + 1 \\
\text{DC} & : pc11 > 2pc1 - 1 \\
\text{ED} & : 5pc11 > 5pc1 - 2 \\
\text{AB} & : 15\text{count}11 < 15\text{count}1 - 9212 \\
\text{BC} & : 15\text{count}11 < 10\text{count}1 + 2303 \\
\text{DC} & : \text{count}11 > 2\text{count}1 - 2303 \\
\text{ED} & : 5\text{count}11 > 5\text{count}1 - 4606
\end{align*}

- Now we have a test for entropy $> 0.4$, as a function of count1 and count11
- This is the Polygon Test
Efficient in Hardware – 17 bit signed numbers

```
<<4

a-b

16c11

<<3

a+b

<<1

15c11

a

b

+2303

-9212

10c1plus2303

a

b

a < b

bc_good

<<1

a+b

<<1

20c1

-2303

20c1minus2303

a

b

a > b

dc_good

<<2

a+b

a

b

-4606

5c1

a+b

5c1

5c1

a > b

ed_good
```
Polygon Test Summary

• If mean and SCC describe the statistics of your noise source well
  • Hint: If you use feedback to maintain a random variable in a goodly random place, mean and SCC probably do describe the statistics of your noise source well

• Then it is possible to implement real time, online tests based on pattern counts, that have a tight pass/fail threshold at some entropy level.

• This is an improvement over tests aimed at distinguishing a failed from a functional source, which inevitably lead to high false positive and/or high false negative errors because they depend on splitting a uniform universe of symbols into a bad set and a good set.

• For obvious reasons, I dubbed this instance of such a test as the “polygon test” and it will feature in upcoming Intel RNGs as the vendor defined CHT.
Summary

• Vendor CHTs are hard because
  • The non-iid tests are unreliable and can give massive underestimation. The resulting number is references in the parameter calculations for tests that work against actual entropy, not the lower bound estimate we get from the non-iid tests
  • 4.5a and 4.5b requirements on vendor CHTs do not help. A test needs to be designed to catch failure modes. Re-creating what the RCT does and adding entropy bounds that are not real is a distraction from what is important in the test

• With 4.5a and 4.5b out the way
  • Given a free hand to design a test, tests that are not looking to meet 4.5a and 4.5b (although they might) can be designed with very sharp entropy cut-offs that are simple to implement and can run at full speed, while the RCT and APT contribute nothing to the detection capability of the CHT system.