OpenSSL CPU Time Jitter RNG Entropy Source version 3.4.0

SP 800-90B Non-Proprietary Public Use Document

Document Version: 1.1

Document Date: 2024-08-27

Prepared by:
atsec information security corporation
4516 Seton Center Parkway, Suite 250
Austin, TX 78759
www.atsec.com

Table of Contents

1 Description

The OpenSSL CPU Time Jitter RNG version 3.4.0 is a non-physical entropy source that is implemented in the Userspace Standalone CPU Time Jitter RNG shared library and the OpenSSL FIPS provider. Its purpose is to feed the secondary DRBGs implemented in the OpenSSL FIPS provider. The noise generation of this entropy source is based on the tiny variations in the execution time of the same piece of code. The execution time of this piece of code is made unpredictable by the complexity of the different hardware components that comprise modern CPUs and the different internal states that the operating system can have at a certain point in time.

The entropy source was tested on the operational environments listed in Table 1. The noise source was tested under the assumption that its output is non-IID.

Table 1: Operational environment and version.

Manufacturer	Model	Operational Environment and Version	Processor
ASUS	RS700-E11-RS4U	SUSE Linux Enterprise Server 15 SP6	Intel® Xeon® Gold 5416S
SuperMicro	SuperChassis 825BTQC-R1K23LPB and Motherboard H12DSi-NT6	SUSE Linux Enterprise Server 15 SP6	AMD EPYC™ 7343
GIGABYTE	R152-P30	SUSE Linux Enterprise Server 15 SP6	Ampere® Altra® Q80-30
IBM	z16 A01	SUSE Linux Enterprise Server 15 SP6	IBM® Telum™

2 Security Boundary

The boundary for this non-physical, software-based entropy source is the two executable binaries. It is compiled from the C code that implements it (i.e., combination of Userspace Standalone CPU Time Jitter RNG shared library and OpenSSL FIPS Provider C code). The noise source and SHA3-256 conditioning component are implemented as part of the Userspace Standalone CPU Time Jitter RNG shared library. The AES-256 CTR DRBG conditioning component is implemented in the OpenSSL FIPS provider.

Figure 1 depicts the overall design of the entropy source and its core operations.

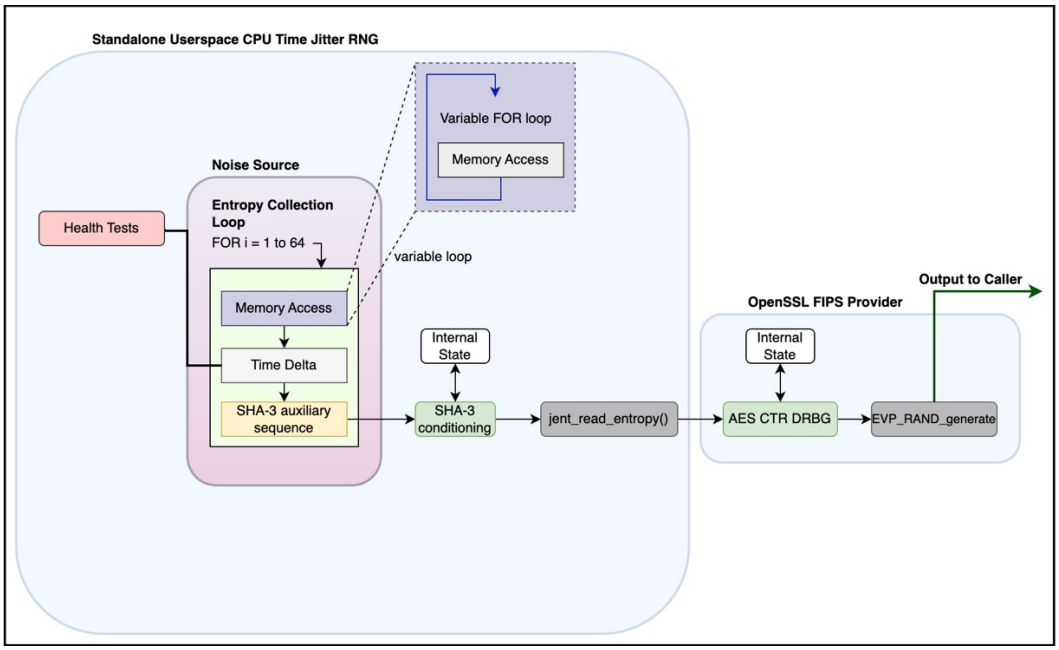

Figure 1: CPU Jitter 3.4.0 Design

The noise source is implemented by collecting and accumulating variances of the execution time of a defined set of instructions. The time jitter of the execution time is measured over time variances of variable memory accesses and variances in the execution time of a defined set of instructions, which includes an implementation of SHA3-256.

The noise source is processed through a chain of three conditioning components:

- a SHA3-256 function that works as the first conditioning component, using the noise source as input.
- a AES-256 CTR DRBG that works as the last conditioning component.

The noise source and the SHA3-256 conditioning components are implemented as part of the Userspace Standalone CPU Time Jitter RNG shared library. The AES-256 CTR DRBG conditioning component is implemented in the OpenSSL FIPS provider, which interfaces with the shared library relying on the jent_read_entropy() syscall.

If the Repetition Count Test (RCT) or the Adaptive Proportion Test (APT) health tests fail, the noise data is discarded, the entropy source halts without outputting any data, and a failure code is returned to the caller.

3 Operating Conditions

The noise source is non-physical, and thus the operating conditions are inherited from the operational environment in which the entropy source is installed, as shown in Table 2 below.

Table 2: Operating Conditions for each Operational Environment

Manufacturer / Model	Temperature	Voltage	Humidity	Clock Speed	Cache Sizes
ASUS RS700-E11-RS4U	10C° – 35C°	+100-240V	20 to 90%	2 GHz	L1i: 16 x 32 KB L1d: 16 x 48 KB L2: 16 x 2 MB L3: 30 MB
SuperMicro SuperChassis 825BTQC-R1K23LPB and Motherboard H12DSi-NT6	10C° – 35C°	+100-240V	8 to 80%	3.2 GHz	L1i: 16 x 32 KB L1d: 16 x 32 KB L2: 16 x 512 KB L3: 128 MB
GIGABYTE R152-P30	10C° – 35C°	+100-240V	8 to 80%	3.0 GHz	L1: 64K Instruction and 64K Data per Core L2: 1 MB per Core
IBM z16 A01	5C° – 40C°	+200-240V / 380-415V / 480V	8 to 85 %	5.2 GHz	L1: 128 KB Instructions and 128 KB Data L2: 32 MB per Core

4 Configuration Settings

OpenSSL allows configuration of the following primary DRBG values:

- random
- cipher
- digest
- properties
- seed
- seed_properties

None of these values should be altered. If any changes are made, this would invalidate the ESV entropy certificate.

In addition to that, EVP_RAND_* APIs allow to specify seed sources which may not be ESV validated. Any external seed source passed as input parameter to EVP_RAND_* APIs will not result in the usage of the entropy source in scope for this ESV validation.

Lastly, the primary DRBG is only used as a (stateful) conditioning component: it is only used to seed the secondary DRBGs and must not be used directly.

5 Physical Security Mechanisms

The noise source is non-physical. The physical security mechanisms only apply to the hardware component of the operational environment in which the entropy source is installed, and thus the entropy source inherits those mechanisms.

6 Conceptual Interfaces

The entropy source provides the following interfaces:

- EVP_RAND_generate(): Obtains conditioned entropy for the caller. This is the main function of the entropy source, the one that shall be used to request entropy data. This interface corresponds to the GetEntropy() conceptual interface from SP800-90B. This function shall be used with the primary DRBG context as the input.
- jent_hash_time(): Obtains raw noise data for testing purposes. This interface corresponds to the GetNoise() conceptual interface from SP800-90B.

7 Min-Entropy Rate

The noise source provides an entropy rate for each time delta $H_{submitter}$ = 1/3 bits/bit.

The entropy source collects 960 time deltas of 64 bits each (61440 bits) as input to the SHA3-256 conditioning component. This corresponds to 320 bits of entropy. The output entropy rate of the SHA3-256 is assessed to be 1 bit/bit.

Then, one 256-bit output block of the SHA3-256 is used to seed the AES-256 CTR DRBG conditioning component, which corresponds to 256 bits of entropy. This DRBG conditioning component outputs a 256 bit block, which is assessed to contain 256 bits of entropy. The AES-256 CTR DRBG employs prediction resistance and derivation function.

As a result, the entropy source provides 1 bit/bit of entropy rate at its output.

8 Health Tests

The entropy source implements the following continuous health tests:

- Repetition Count Test conforming to SP 800-90B section 4.4.1.
  - $H = 1$ bit of entropy per 64-bit sample.
  - alpha value of $\alpha = 2^{-30}$.
  - Cutoff value $C = 31$.
- Adaptive Proportion test conforming to SP 800-90B section 4.4.2.
  - $W = 512$
  - $H = 1$ bit of entropy per 64-bit sample
  - alpha value of $\alpha = 2^{-30}$.
  - Cutoff value $C = 325$.
- Stuck (Non-Permanent) Test: The stuck test computes the first, second and third discrete derivatives of the time value that will be processed by SHA3-256. If any of these derivatives are zero, then the received time delta is considered stuck. In this case the input state to SHA3-256 is not updated, and the entropy value is not counted. The stuck test then triggers the RCT for further processing. The second derivative is in fact the RCT itself.

- Lag Predictor Test: The goal of this test is to detect a failure mode in which the outputs may become mostly deterministic. In essence, this test constructs a scoreboard and tracks the number of times that a subpredictor was correct. The subpredictor that scored the most correct predictions is used to predict the next value of a series. The lag predictor test is configured in this entropy source with the following parameters:
    - $\alpha = 2^{-22}$
    - Window size: $window\_size = 131072$
    - Lag history size: $lag\_history\_size = 8$
    - Global cutoff = $InverseBinomialCDF = CRITBINOM(n = window_{size} - lag_{history_{size}}; p = 2^{-\frac{1}{osr}}; 1 - \alpha)$
    - Local cutoff = 111

The continuous-health tests are applied to each new sample obtained from the noise source. Whenever a failure is detected during the health testing specifically for the RCT and APT, entropy data is not returned to the caller; instead, a failure code is returned to enable the caller to acknowledge the failure. The entropy source then halts and will refuse new requests for entropy. Upon return of the failure code, the caller shall attempt to reset or reboot the entropy source or return an error to its own operator. The stuck test is considered non-permanent, as positive stuck tests will be registered but will not immediately halt the entropy source.

Startup tests conduct the same set and parameters of the continuous health tests on 1024 samples of noise data. The data is discarded after the startup tests have completed successfully.

On-demand health tests of the noise source may be performed by rebooting the operational environment, which results in the immediate execution of the start-up tests. Typically, this entropy source designed for user space cannot be reloaded without restarting the executable. Similarly, the data used for the on-demand health tests are discarded after successful completion.

9 Maintenance

There are no maintenance requirements as this is a software-based entropy source.