ORACLE
Linux

SP 800-90B Non-Proprietary Public Use Document for Kernel CPU Time Jitter RNG Entropy Source

Entropy Source Version 2.2.0

Document Version 1.0

Last Update: 2023-05-22

Prepared for:

Oracle Corporation
World Headquarters
2300 Oracle Way
Austin, TX 78741
U.S.A.

Prepared by:

atsec information security Corp.
9130 Jollyville Road, Suite 260
Austin, TX 78759
USA

ORACLE
Linux

Table of Contents

1 Description

The Kernel CPU Time Jitter RNG version 2.2.0 is a non-physical entropy source. 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 environments and version.

Model	Operational Environment	Processor
Oracle Server X7-2	Oracle Linux 7.8	Intel® Xeon® Platinum 8167M
Oracle Server E1-2C	Oracle Linux 7.8	AMD EPYC™ 7551
Oracle Server A1	Oracle Linux 7.8	Ampere® Altra® Neoverse-N1
Marvell Liquid IO II	Oracle Linux 7.8	Marvell OCTEON III

2 Security Boundary

The boundary for this non-physical, software-based entropy source is the executable binary file. It is compiled from the C code that implements it. Figure 1 depicts the overall design of the entropy source and its core operations.

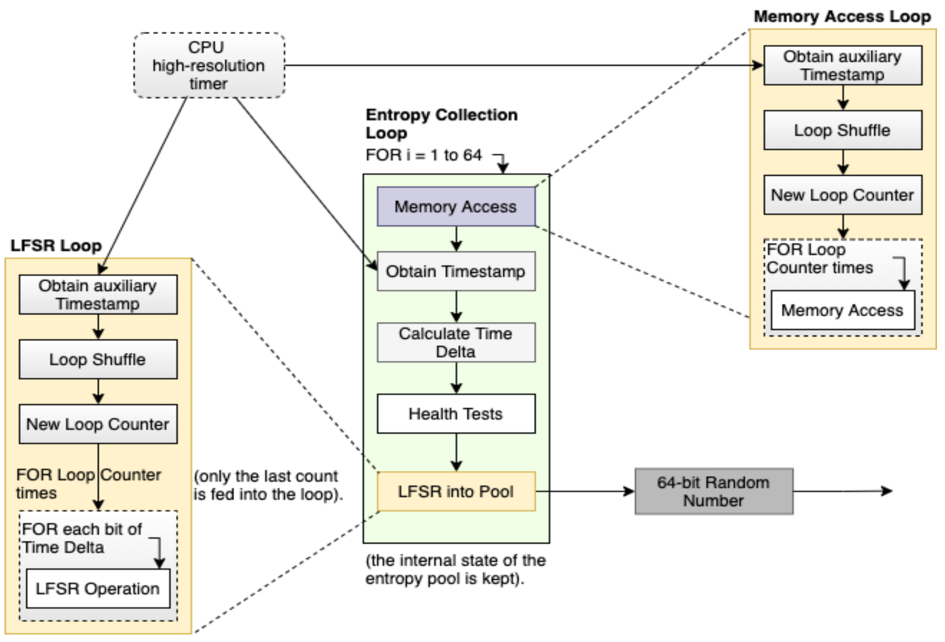

Figure 1: Security boundary of the entropy source.

The noise source is implemented by collecting and accumulating variances of the execution time of a defined set of instructions. The accumulation is done with the assistance of the LFSR function that also makes the bijective conditioning component of the entropy source, and an entropy pool that is used to accumulate the acquired entropy. The time jitter of the execution time is measured over the processing logic of the LFSR and functions that support the implementation.

If the Repetition Count Test (RCT) or the Adaptive Proportional 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 environments in which the entropy source is installed as shown in Table 1.

The operating conditions are expressed in Table 2.

Table 2: Operating conditions.

Processor	Temperature	Voltage	Clock Speed	Cache Size
Intel® Xeon® Platinum	65°C	0 - 2.15V	2.4 GHz	L1: 26x32 KB L2: 26x1 MB

Processor	Temperature	Voltage	Clock Speed	Cache Size
8167M	Junction temperature			L3: 35.75 MB
AMD EPYC™ 7551	0C to 100C Processor temperature control value	0V - 3.3V	3.0 GHz	L1: 32x64 KB L2: 32x512 KB L3: 64 MB
Ampere® Altra® Neoverse-N1	0C to 100C Junction temperature	0V - 1.1V	3.0 GHz	L1: 80x64 KB L2: 80x1 MB
Marvell OCTEON III	0C to 105C Junction temperature	0.82V - 0.88V	1.5 GHz	L1: 32x16 KB L2: 4096 KB

4 Configuration Settings

There are no specific configuration settings required for this entropy source. However, the following lists "#defines" in jitterentropy.c:

#define JENT_MEMORY_BLOCKS 64

#define JENT_MEMORY BLOCKSIZE 32

#define JENT_MEMORY_ACCESSLOOPS 128

None of these values should be altered. If any changes are made, a new ESV certificate will be required for the updated configuration of the entropy source.

5 Physical Security Mechanisms

The noise source is non-physical. The physical security mechanisms only apply to the hardware component of the operational environments 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:

- jent_read_entropy(): Obtains conditioned data for the caller. This is the main function from the entropy source, the one that shall be used to request output from entropy source. The entropy gathering logic creates 64 bits per invocation. This interface corresponds to the GetEntropy() conceptual interface from SP800-90B.
- jent_lfsr_time(): Obtains raw noise data for testing purposes. This interface corresponds to the GetNoise() conceptual interface from SP800-90B.

7 Min-Entropy Rate

The entropy source provides 57.52576 bits of entropy per each 64-bit output, that is an entropy rate of 0.899 bits/bit.

After the entropy source output is obtained by the consuming application (e.g. cryptographic module), additional entropy can be obtained by the concatenation of each 64-bit output. The entropy source does not truncate or partition the data before it is returned to the caller.

A module that utilizes the entropy source shall consider the amount of entropy available as described here and list such values in its Security Policy.

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 the LFSR. If any of these derivatives are zero, then the received time delta is considered stuck. In this case the LFSR is still updated, but the entropy value is not counted and the output of the LFSR is not used for insertion into the entropy pool. The stuck test then triggers the RCT for further processing. The second derivative is in fact the RCT itself.

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 complete successfully.

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

Table 3 defines the error codes that can be returned by jent_read_entropy():

Table 3: Error codes.

Error Code	Description
-1	entropy_collector is NULL
-2	RCT failed
-3	APT test failed

9 Maintenance

There are no maintenance requirements as the entropy source is software based.

10 Required Testing

To test the entropy source, raw data samples must be collected using a test harness that is capable of accessing the jent_lfsr_time() noise interface from the entropy source. The test harness and accessory kernel tools must be supplied by the vendor.

Raw noise data samples consisting of at least 1,000,000 bytes must be collected from the operational environments at its normal operating conditions and processed by the SP 800-90B entropy tool that is provided by NIST. The expected min-entropy rate must approach the one in Section 7.

Restart data must be collected at normal operating conditions through the jent_lfsr_time() interface following the restart procedure specified in SP 800-90B (i.e., 1,000 samples from 1,000 restarts each) and processed by the NIST SP 800-90B entropy tool. The minimum of the row-wise and column-wise entropy rate must be more than half that of the raw noise entropy rate.

If conditioned entropy data is to be tested, then the jent_read_entropy() interface shall be accessed instead of jent_lfsr_time() interface.