Chainguard CPU Time Jitter RNG Entropy Source version 3.5.0

SP 800-90B Non-Proprietary Public Use Document

Document Version: 1.0

Document Date: 07-30-2024

Prepared by:

atsec information security corporation

4516 Seton Center Pkwy, Suite 250

Austin, TX 78759

www.atsec.com

Table of Contents

1 Description

The Chainguard CPU Time Jitter RNG version 3.5.0 is a non-physical, software-based entropy source in the form of a static library, libjitterentropy.a, that can be linked to applications to seed their DRBGs. 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
Amazon Web Services (AWS)	EC2 m7i.metal-24xl	Chainguard Image 20230214 on Amazon Linux 2023	Intel Xeon Platinum 8488C
Amazon Web Services (AWS)	EC2 m7g.metal	Chainguard Image 20230214 on Amazon Linux 2023	AWS Graviton3

Chainguard Image 20230214 is a Docker image.

2 Security Boundary

The boundary of the entropy source is the static library libjitterentropy.a.

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

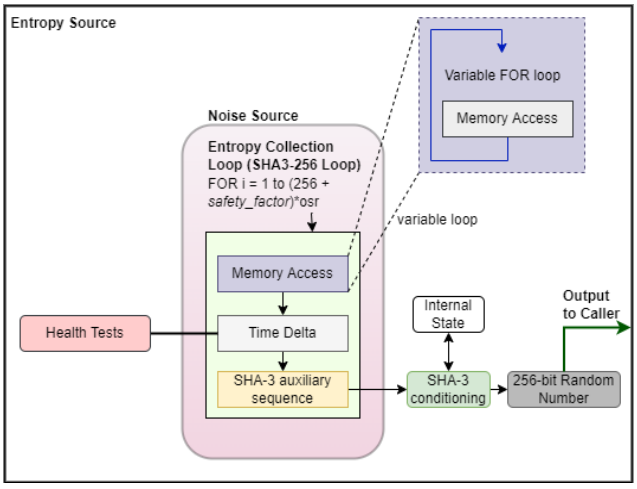

Figure 1: CPU Jitter 3.5.0 Design

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 output of the noise source is conditioned by a SHA3-256 conditioning component.

The noise source and the SHA3-256 conditioning component are implemented as part of the libjitterentropy.a static library.

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
EC2 m7i.metal-24xl	0°C-45°C	12V	60%	0.8–3.8 GHz	L1d: 2.3 MiB x 48 L1i: 1.5 MiB x 48 L2: 96 MiB x 48 L3: 105 MiB
EC2 m7g.metal	0°C-45°C	12V	60%	2.6 GHz	L1d: 4 MiB x 64 L1i: 4 MiB x 64 L2: 64 MiB x 64 L3: 32 MiB

4 Configuration Settings

The caller shall use the jent_read_entropy API to obtain entropy source output. The caller shall use the jent_entropy_collector_alloc API with the JENT_FORCE_FIPS flag.

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:

- jent_read_entropy(): 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.
- jent_measure_jitter(): 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 of $H_{submitter}$ = 1/3 bit per time delta.

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.

As a result, the entropy source provides 1 bit of entropy per bit of output (full entropy).

The oversampling rate (osr) and safety factor are fixed values defined in the implementation and cannot be altered.

8 Health Tests

The entropy source implements the following continuous health tests:

- Repetition Count Test conforming to SP 800-90B section 4.4.1.
  - $H = 1/3$ bit of entropy per 64-bit time delta
  - Intermittent failure alpha value $\alpha_i = 2^{-30}$
  - Permanent failure alpha value $\alpha_p = 2^{-60}$
- Adaptive Proportion test conforming to SP 800-90B section 4.4.2.
  - $W = 512$
  - $H = 1/3$ bit of entropy per 64-bit time delta
  - Intermittent failure alpha value $\alpha_i = 2^{-30}$
  - Permanent failure alpha value $\alpha_p = 2^{-60}$
- Stuck Test (Non-Permanent): The stuck test computes the first, second and third discrete derivatives of the time value that will be processed by the 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 (Non-Permanent): 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 $W = 131072$
  - Lag history size $L = 8$

As allowed by Section 4.3 of SP 800-90B, the entropy source defines two types of health test failures for the RCT and the APT: intermittent failures and permanent failures.

An intermittent failure is characterized by a false positive probability $\alpha_i = 2^{-30}$, which lies within the recommended range of $2^{-20} \leq \alpha \leq 2^{-40}$. When an intermittent failure is detected,

the CPU Jitter RNG is automatically reset (which clears the entropy pool), and the caller is notified of this intermittent failure.

Permanent failures are characterized by a false positive probability of $\alpha_p = 2^{-60}$. When a permanent failure is detected, the CPU Jitter RNG is automatically reset and the caller is notified of this permanent failure.

The continuous-health tests are applied to each new time delta obtained from the noise source.

Start-up tests conduct the same set and parameters of the continuous health tests on 1024 time deltas. The data is discarded after the start-up tests have completed successfully.

On-demand health tests of the noise source may be performed by deallocating the current Jitter RNG instance and reallocating a new instance (thus, restarting the entropy source). Similar to the start-up tests, 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.

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_measure_jitter() noise interface from the entropy source. The test harness must be supplied by the vendor.

Raw noise data consisting of at least 1,000,000 64-bit samples truncated to 8 bits must be collected from the operational environment 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_measure_jitter() 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.