SP 800-90B Non-Proprietary Public Use Document

Trellix Intrusion Prevention System Sensors Jitter Entropy 3.4.0

Document Version 0.3
Trellix, 6000 Headquarters Drive, Suite 600, Plano, TX 75024
October 10, 2024

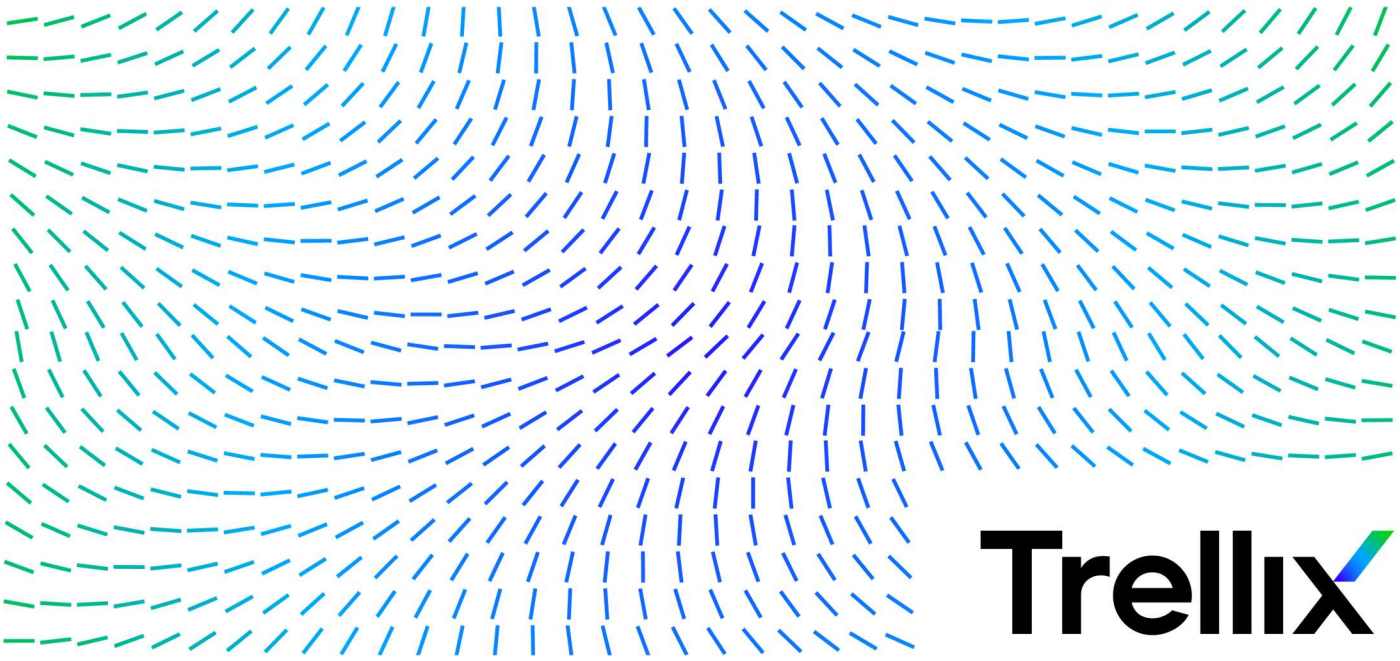


Table of Contents

Revision History 3

Description 4

Security Boundary 6

Operating Conditions 7

Configuration Settings 8

Physical Security Mechanisms 9

Conceptual Interfaces 10

Min-Entropy Rate 11

Health Tests 12

Maintenance 13

Required Testing 14

Revision History

Version	Change
0.1	Initial version
0.120221216	Intermediate update
0.2	Updates to OE table based on EAR
0.3	Formatting and editorial updates
0.4	Added NS9600 model

Description

CPU Jitter Entropy (JE) is a non-physical (NP) entropy source that is a non-IID software source incorporated into hardware FIPS 140-3 modules. The JE version is 3.4.0 and is compliant with SP 800-90B.

The basic concept of the CPU Jitter Entropy noise source is based on the unpredictable phenomenon of variances in the execution time of a given set of instructions which is collected and accumulated. The accumulation is implemented using SHA-3 256. The measurement of the execution time jitter is performed over the post-processing logic of the SHA-3 256 and supportive functions. I.e. the CPU Jitter RNG measures the execution time of the SHA-3 256 with its supporting functions (and the additional noise source of the memory access) where the execution time is then injected into the SHA-3-maintained entropy pool. The random number generator maintains a 256-bit entropy pool, that is indicated with the dark gray shaded boxes in Figure 1 which identify the entropy pool at two different times in the processing. The size is equal to the SHA-3 256 output size. The light gray shaded boxes indicate the noise sources of the random number generator is based on.

In a big picture, the random number generator implements an entropy collection loop that:

- invokes memory accesses to induce timing variations,
- fetches a time stamp to calculate a delta to the time stamp of the previous loop iteration,
- injects the time delta value into the entropy pool by hashing the content of the entropy pool with the new value to form a new state of the entropy pool, and
- verifies that basic properties of the time stamp are met by applying a health test.

The loop is executed exactly 256 times multiplied by the oversampling rate (osr) as each loop iteration is intended to provide at least 1/osr bit of entropy added to the entropy pool. This implies that after the completion of all rounds, at least 256 bits of entropy are collected and compressed into the 256-bit entropy pool – i.e. the entropy pool contains full entropy. The following subsection discusses every step in detail. When considering that the time delta is always calculated by calculating the delta to the previous loop iteration, and the fact that the majority of the execution time is spend in the SHA-3 hashing operation, the central idea of the CPU Jitter Random Number Generator is to measure the execution time jitter over the execution of the SHA-3 hashing operation as well as the memory access.

Source: https://www.chronox.de/jent/CPU-Jitter-NPTRNG.pdf

The entropy source has been tested on the following hardware platforms:

OE Name	Processor	Mircoarchitec-ture	H/W Version
Ubuntu 20.04 on Intel(R) Atom(R)C Series (C2538, Rangeley)	Atom C2538	Rangeley	1.00
Ubuntu 20.04 on Intel(R) Xeon(R) E5 v3 Family (E5-2620 V3, Has-well)	Xeon E5-2620 V3	Haswell	1.10
Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6130, Skylake)	Xeon Gold 6130	Skylake	1.00
Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 5218N, Cascade Lake)	Xeon Gold 5218N	Cascade Lake	1.00
Ubuntu 20.04 on Intel(R) Xeon(R) E5 Family (E5-4610, Sandy Bridge)	Xeon E5-4610	Sandy Bridge	1.30

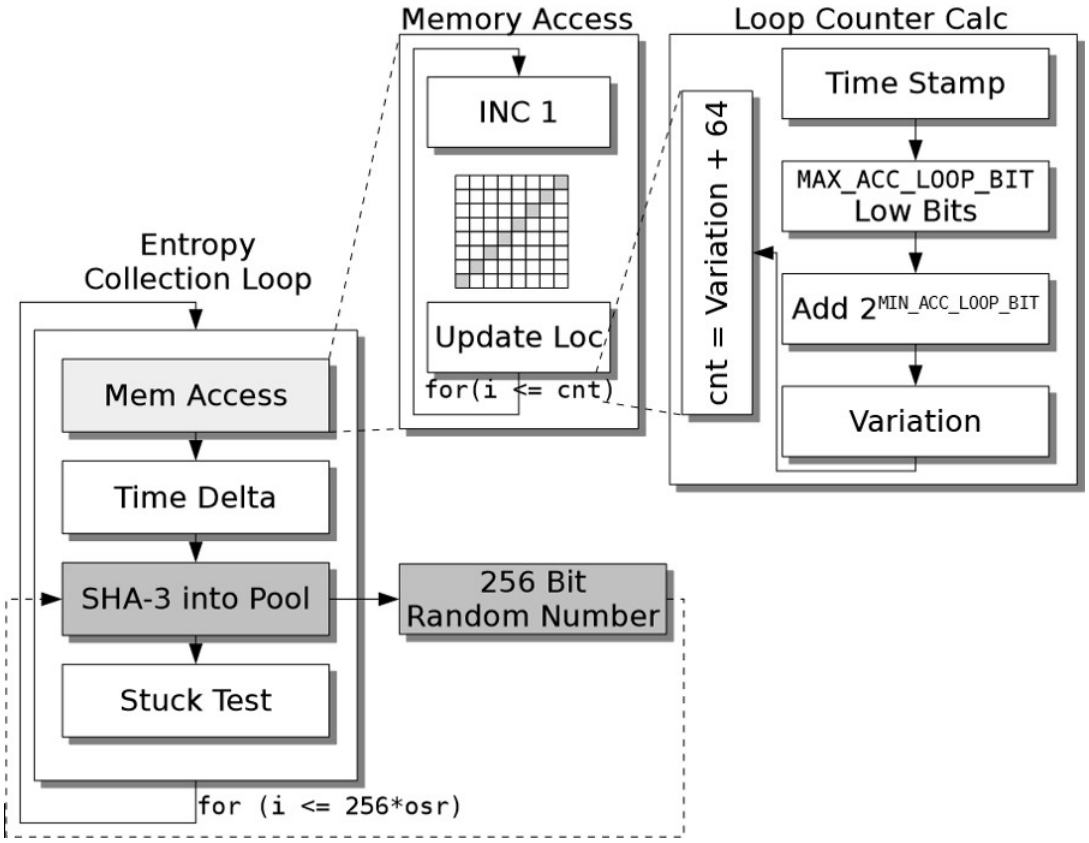
OE Name	Processor	Mircoarchitec-ture	H/W Version
Ubuntu 20.04 on Intel(R) Xeon(R) E5 Family (E5-4640, Sandy Bridge)	Xeon E5-4640	Sandy Bridge	1.30
Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6138, Skylake)	Xeon Gold 6138	Skylake	1.00
Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6230, Cascade Lake)	Xeon Gold 6230	Cascade Lake	1.10
Ubuntu 20.04 on Intel(R) Xeon(R) D Processors (D-1734NT, Ice Lake)	Xeon D-1734NT	Ice Lake	1.00
Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (SILVER 4416+, Sapphire Rapids)	Xeon Silver 4416+	Sapphire Rapids	1.00
Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 5520+, Emerald Rapids)	Xeon Gold 5520+	Emerald Rapids	1.00

A mapping of the Sensor Models to Operational Environments is as follows:

Sensor Model	ESV CERT OE NAME
IPS-NS9100	Ubuntu 20.04 on Intel(R) Xeon(R) E5 Family (E5-4610, Sandy Bridge)
IPS-NS9200	Ubuntu 20.04 on Intel(R) Xeon(R) E5 Family (E5-4640, Sandy Bridge)
IPS-NS9300 P	Ubuntu 20.04 on Intel(R) Xeon(R) E5 Family (E5-4640, Sandy Bridge)
IPS-NS9300 S	Ubuntu 20.04 on Intel(R) Xeon(R) E5 Family (E5-4640, Sandy Bridge)
IPS-NS7150	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6130, Skylake)
IPS-NS7250	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6130, Skylake)
IPS-NS7350	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6130, Skylake)
IPS-NS5200	Ubuntu 20.04 on Intel(R) Xeon(R) E5 v3 Family (E5-2620 V3, Haswell)
IPS-NS5100	Ubuntu 20.04 on Intel(R) Xeon(R) E5 v3 Family (E5-2620 V3, Haswell)
IPS-NS3200	Ubuntu 20.04 on Intel(R) Atom(R)C Series (C2538, Rangeley)
IPS-NS3100	Ubuntu 20.04 on Intel(R) Atom(R)C Series (C2538, Rangeley)
IPS-NS9500 1.00	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6138, Skylake)
IPS-NS9500 1.10	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 6230, Cascade Lake)
IPS-NS3600	Ubuntu 20.04 on Intel(R) Xeon(R) D Processors (D-1734NT, Ice Lake)
IPS-NS7600	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (SILVER 4416+, Sapphire Rapids)
IPS-NS7500	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 5218N, Cascade Lake)
IPS-NS9600	Ubuntu 20.04 on Intel(R) Xeon(R) Scalable Processors (GOLD 5520+, Emerald Rapids)

Security Boundary

The following figure depicts the entropy source's components. The entropy source's security boundary is the operational environment of the containing cryptographic module. The entropy source gathers its entropy from the time deltas between memory access operations and is used to seed a SP 800-90A DRBG.



Operating Conditions

The operating conditions under which this entropy source is claimed to operate correctly is for input power to the hardware platforms of 100 – 240 volts alternating-current (VAC) (50/60Hz) over a temperature range of 0 degrees C to 35 degrees C. The testing was performed at 220 VAC 50Hz nominal input voltage and frequency.

Configuration Settings

Because this entropy source is incorporated into FIPS 140-3 hardware modules, there are no configuration settings available to the user. The default settings are used and are listed in the table as follows.

Configuration item	Setting	Purpose
JENT_CONF_DISABLE_LOOP_SHUFFLE	Defined/enabled	Only execute one hash loop
JENT_MEMORY_ACCESSLOOPS	128	Memory access loops
JENT_MEMORY_BLOCKS	512	Number of memory blocks
JENT_MEMORY_BLOCKSIZE	128	Memory block size
JENT_MIN_OSR	3	Oversampling rate
JENT_CONF_ENABLE_INTERNAL_TIMER	Not defined	Not defined so hardware fine-grained timer is used

Physical Security Mechanisms

The entropy source inherits the security mechanisms of the cryptographic module in which it is implemented.

Conceptual Interfaces

This entropy source is incorporated into FIPS 140-3 hardware modules. Therefore there are no conceptual interfaces available to the user. The following are conceptual interfaces per SP 800-90B, and the equivalents provided by the entropy source:

Conceptual Interface	Description
GetEntropy	jent_read_entropy
GetNoise	the script invoke_testing.sh is used to gather the raw entropy for testing
HealthTest	health tests are performed at startup of JE and are also run continuously

Min-Entropy Rate

This entropy source provides a min-entropy rate of 1 bit per bit.

Health Tests

SP 800-90B health tests are performed using the Adaptive Proportion Test (APT) and Repetition Count Test (RCT) and are run at startup and continuously under the normal operating conditions listed previously. Any errors detected are returned for handling by the caller. There are no known suspected noise source failure modes.

Maintenance

There are no maintenance requirements for this entropy source.

Required Testing

The target platform testing included gathering 1,000,000 raw data samples as well as 1,000 samples of raw data after each of 1,000 entropy source restarts. No additional testing is required. To verify that the entropy source is configured and operating correctly, the health tests can be initiated on demand.

COPYRIGHT

2024 © Musarubra US LLC

Trellix and FireEye are the trademarks or registered trademarks of Musarubra US LLC, FireEye Security Holdings US LLC, and their affiliates in the US and /or other countries. Other names and brands are the property of these companies or may be claimed as the property of others.

