

SP 800-90B Non-Proprietary Public Use Document

Arista Entropy Source

Version 3.4.1

Arista Network, Inc.
5453 Great America Parkway
Santa Clara, CA 95054
USA

Document Version 1.1

June 2026

Document prepared by:



www.lightshipsec.com

Table of Contents

1 Description 3
2 Security Boundary 3
3 Operating Conditions 4
4 Configuration Settings 7
5 Physical Security Mechanisms 8
6 Conceptual Interfaces 8
7 Min-Entropy Rate 9
8 Health Tests 9
9 Maintenance 9
10 User Verification 9
10.1 Collecting samples 9
10.2 Raw Data Analysis 9
10.3 Entropy Source Version Verification 10
11 Vendor Permissions and Relationship 10

1 Description

This is the SP 800-90B non-proprietary Public Use Document for Arista Entropy Source (hereafter referred to as "the Entropy Source"). The Entropy Source is based on the jitterentropy-library v3.4.1.

The Entropy Source makes no IID claim and thus meets all the requirements for non-IID compliance. The tested platforms are described in Table 1.

The Entropy Source is compliant to NIST Special Publication 800-90B Recommendation for the Entropy Sources Used for Random Bit Generation (Last Modified Date: January 2018), IG D.J (Last Modified Date: November 5, 2021) and IG D.K (Last Modified Date: March 17, 2023).

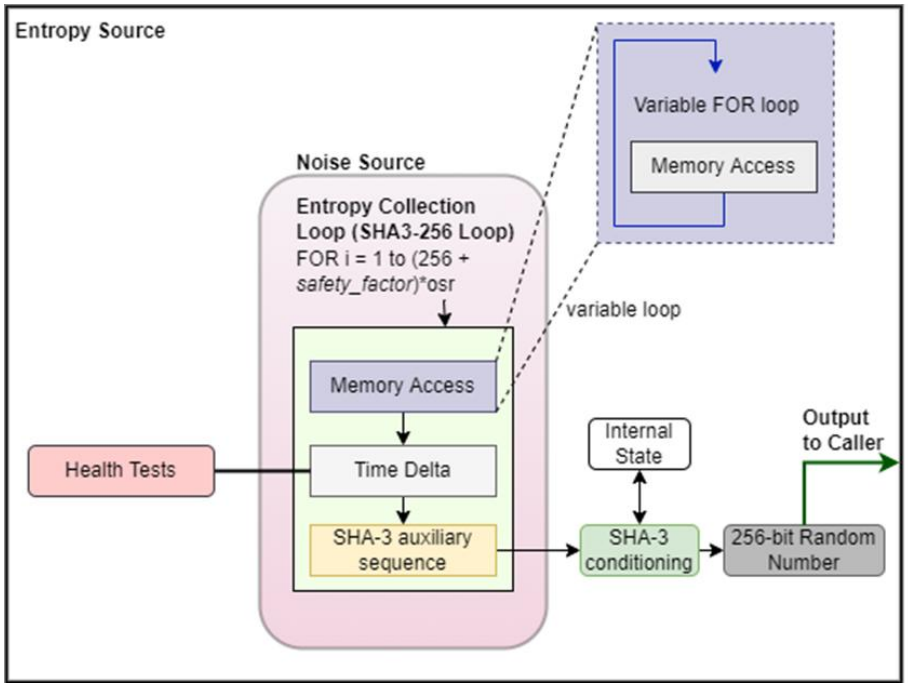
2 Security Boundary

The Entropy Source operates within a defined security boundary that includes both software and hardware components.

The software component is the JEnt library itself, which implements the entropy generation functions. The hardware component is the underlying computational environment required to execute the library, including the processor and system timer.

JEnt is based on the jitterentropy-library. Its design maps to the conceptual components described in NIST SP 800-90B and includes:

- A primary noise source (based on memory access and conditioning timing),
- Health tests to ensure the proper functioning of the entropy source
- A conditioning function implemented using SHA3-256.



3 Operating Conditions

The Entropy Source was tested in the following operational environments:

Table 1. Tested Operational Environments

Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 17 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE1	AMD EPYC™ Embedded 3251 (Zen) CPU Clock Speed: 2.50 GHz L1 Cache: 32 KB L2 Cache: 512 KB L3 Cache: 16 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 3 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE2	AMD G-Series GX-224PC (Puma) CPU Clock Speed: 2.40 GHz L1 Cache: 32 KB L2 Cache: 1 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 15 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE3	AMD R-Series RX-421ND (Excavator) CPU Clock Speed: 2.10 GHz L1 Cache: 32 KB L2 Cache: 2 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1		

Oversampling rate: osr = 4 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE4	Intel Atom® Processor C2558 (Silvermont) CPU Clock Speed: 2.40 GHz L1 Cache: 24 KB L2 Cache: 1 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 7 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE5	Intel Atom® Processor C3558R (Goldmont) CPU Clock Speed: 2.40 GHz L1 Cache: 24 KB L2 Cache: 2 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 20 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE6	Intel® Xeon® D-2798NX Processor (Sunny Cove) CPU Clock Speed: 2.10 GHz L1 Cache: 48 KB L2 Cache: 1.25 MB L3 Cache: 30 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 12 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**

OE7	Intel® Xeon® Processor D-1548 (Broadwell) CPU Clock Speed: 2.00 GHz L1 Cache: 32 KB L2 Cache: 256 KB L3 Cache: 12 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 12 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE8	Intel Atom® Processor P5721 (Tremont) CPU Clock Speed: 2.20 GHz L1 Cache: 32 KB L2 Cache: 4.5 MB L3 Cache: 7.5 MB	EOSv4
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 19 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE9	Intel® Xeon® Silver 4316 (Sunny Cove) CPU Clock Speed: 2.30 GHz L1 Cache: 48 KB L2 Cache: 1.25 MB L3 Cache: 30 MB	CloudVision Appliance v8 on VMware vSphere ESXi 9.0
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 41 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE10	Intel® Xeon® Gold 6534 (Emerald Rapids) CPU Clock Speed: 3.90 GHz L1 Cache: 48 KB L2 Cache: 2 MB	CloudEOSv4 on VMware vSphere ESXi 9.0

	L3 cache: 22.5 MB	
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 41 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE11	Intel® Xeon® Gold 6534 (Emerald Rapids) CPU Clock Speed: 3.90 GHz L1 Cache: 48 KB L2 Cache: 2 MB L3 cache: 22.5 MB	CloudEOSv4 on Hyper-V in Windows Server 2025
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 41 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE12	Intel® Xeon® Gold 6534 (Emerald Rapids) CPU Clock Speed: 3.90 GHz L1 Cache: 48 KB L2 Cache: 2 MB L3 cache: 22.5 MB	CloudEOSv4 on KVM in Ubuntu 24.04
Operating System and Software CPU Jitter RNG version 3.4.1 Oversampling rate: osr = 28 Health Testing reporting and use of safety factor: fips_enabled		
Label	**CPU**	**Operating System**
OE13	AMD Ryzen™ Embedded V3C48 (Zen 3) CPU Clock Speed: 3.30 GHz L1 Cache: 32 KB L2 Cache: 512 KB L3 Cache: 16 MB	EOSv4

4 Configuration Settings

The tables below list the Entropy Source's configuration parameters. These settings must be preserved to comply with the CPU Jitter ESV certificate.

Table 2. CPU Jitter Configuration Parameters

Parameter	Default Value	Location
JENT_CONF_DISABLE_LOOP_SHUFFLE	Defined	jitterentropy.h
JENT_HEALTH_LAG_PREDICTOR	Defined	jitterentropy.h
JENT_RANDOM_MEMACCESS	Defined	jitterentropy.h
ENTROPY_SAFETY_FACTOR	64	jitterentropy.h
JENT_MIN_OSR	41	jitterentropy.h
JENT_MEMORY_SIZE	2^{17}	jitterentropy.h
JENT_MEMORY_BITS	17	jitterentropy.h
JENT_MEMORY_BLOCKS	512	jitterentropy.h
JENT_MEMORY_BLOCKSIZE	128	jitterentropy.h
JENT_MEMORY_ACCESSLOOPS	128	jitterentropy.h
JENT_CONF_ENABLE_INTERNAL_TIMER	Defined (or not)	Makefile
JENT_APT_WINDOW_SIZE	512	jitterentropy.h
JENT_LAG_WINDOW_SIZE	2^{17}	jitterentropy.h
JENT_LAG_HISTORY_SIZE	8	jitterentropy.h
JENT_POWERUP_TESTLOOPCOUNT	1024	jitterentropy-base.c
CLEARCACHE	100	jitterentropy-base.c
MIN_HASH_LOOP	0	jitterentropy-noise.c
MIN_ACC_LOOP_BIT	0	jitterentropy-noise.c

Table 3. CPU Jitter Configuration Flags (for jent_entropy_init_ex and jent_entropy_collector_alloc)

Parameter	Status	Argument Name	Meaning
JENT_DISABLE_MEMORY_ACCESS	Prohibited	flags	Disable the memory access timing.
JENT_FORCE_INTERNAL_TIMER	Allowed	flags	Force use of pthreads-based internal timer, rather than the underlying hardware timer.
JENT_DISABLE_INTERNAL_TIMER	Allowed	flags	Prevent the use of the pthreads-based internal timer.
JENT_FORCE_FIPS	Recommended	flags	Force fips_enabled to be set (enabling health test reporting and use of the safety factor).
JENT_MAX_MEMSIZE_*	Allowed	flags	Allows for run-time selection of the memory region size.
osr	Suggested	osr	Sets the internal osr rate to max(osr, JENT_MIN_OSR).

5 Physical Security Mechanisms

N/A – The Entropy Source is non-physical.

6 Conceptual Interfaces

The entropy source provides the following interfaces:

- **jent_read_entropy():** Obtains entropy for the caller. The entropy gathering logic creates 256 bits per invocation. This interface corresponds to the GetEntropy() conceptual interface from SP800-90B.
- **jent_entropy_init():** Initializes the Entropy Source and performs statistical tests to verify that the underlying system offers the properties needed for measuring and collecting entropy. This interface corresponds to the HealthTest() conceptual interface from SP800-90B.

7 Min-Entropy Rate

The Entropy Source provides 256 bits of assessed min-entropy per 256-bit output sample, or full entropy.

8 Health Tests

The Entropy Source implements the following health tests:

- Stuck Test
- Repetition Count Test
- Adaptive Proportion Test
- Lag Predictor Test

These tests are run when the Entropy Source starts and continuously while it is operating. All tests check for persistent failures based on their respective "cutoff" values, which represent expected error thresholds. All these tests are performed on raw data before any conditioning is done.

All health test failures are fatal. Once triggered, the Entropy Source will transition into an error state and will no longer provide entropy. To clear the error state, the calling application must unload the failed Entropy Source and instantiate a new instance.

9 Maintenance

N/A – The Entropy Source does not require maintenance.

10 User Verification

To confirm that the Entropy Source has been configured properly, the developer shall run the 90B Entropy Assessment tools on the noise source outputs in the same manner that was done during the Entropy Source Validation exercise. The steps involved are described below.

10.1 Collecting samples

The developer shall run the jitterentropy-hashtime tool included in the Entropy Source package. This script gathers 1,000,000 continuous samples of raw noise data. It also gathers 1,000,000 samples of restart data which comprised of 1000 files each containing the first 1000 samples generated after the Entropy Source has been restarted.

10.2 Raw Data Analysis

The developer shall run the noise source samples through NIST's SP 800-90B Entropy Assessment tools. The Entropy Source is configured properly if the assessment yields a min entropy rate of at least 1/OSR bits per sample.

10.3 Entropy Source Version Verification

The JENT library version can be verified using the jent_version() API. For OEs using the OpenSSL entropy provider, the version can be verified using the following command:

- openssl list -providers

The expected output includes the Arista CPU Jitter Entropy Provider and its version, for example:

- jitterentropy_fips
  - name: Arista CPU Jitter Entropy Provider
  - version: v3.4.1-1
  - status: active

The version format is v{X.Y.Z}-{A}, where X.Y.Z represents the underlying jitterentropy-library major, minor, and patch version obtained using the jent_version() API, and A represents the Arista entropy provider revision.

11 Vendor Permissions and Relationship

The reuse of this ESV requires written and signed permission from Arista Networks, Inc.