IBM QRadar Kernel CPU Time Jitter RNG Entropy Source version 2.2.0

SP 800-90B Non-Proprietary Public Use Document

Document Version: 1.0
Document Date: May 14, 2026

Prepared by

Gossamer Laboratories By SGS

Revision History

Version	Date	Change
V1.0	05/14/2026	Initial draft

Table of Contents

Table of Contents	3
Description	4
Security Boundary	5
Operating Conditions	5
Configuration Settings	6
Physical Security Mechanisms	6
Conceptual Interfaces	6
Min-Entropy Rate	6
Health Tests	7
Maintenance	8
User Verification	8
Vendor Permissions and Relationship	8

Description

The IBM QRadar Kernel CPU Time Jitter RNG version 2.2.0 is a non-physical entropy source, part of the kernel image, that feeds the kernel DRBG. 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.

IBM QRadar submitted this entropy source under the following versions of governing standards:

1. The January 2018 version of NIST SP 800-90B,
2. The November 5, 2021 version of FIPS 140-3 IG D.J, and
3. The March 17, 2023 version of FIPS 140-3 IG D.K.

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.

Model	Operational Environment and Version	Arch/ISA	Processor
R760	Red Hat Enterprise Linux 8	x86_64	Intel Xeon® Gold 6426Y (Sapphire Rapids/Golden Cove)

Table 1 – Operational environments and version

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 (i.e., kernel source code).

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

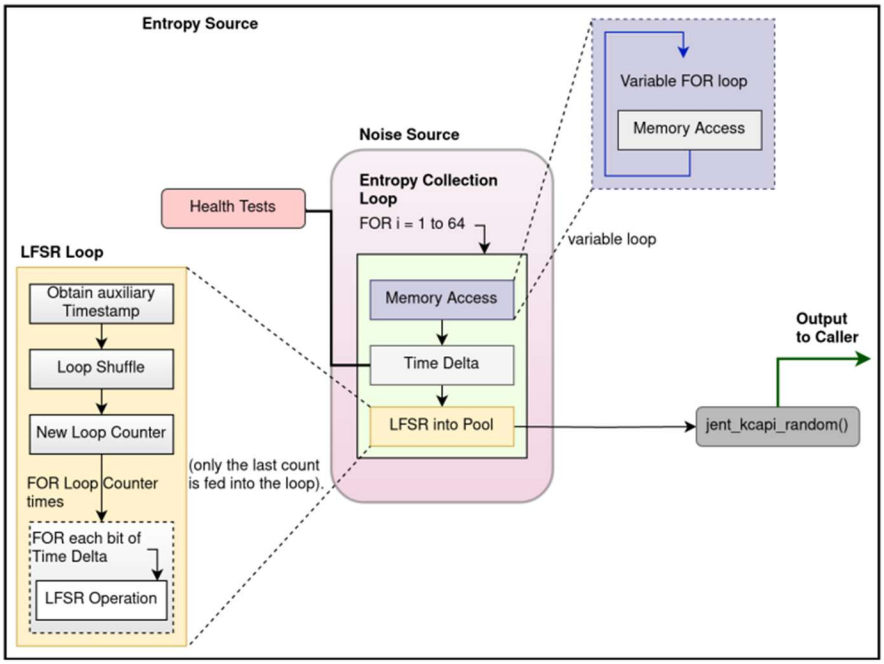


Figure 1 - Entropy Source Diagram

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 a Linear-Feedback Shift Register (LFSR) function that works as the first conditioning component of the entropy source, as well as 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. If the health test failure is permanent, the kernel which contains this entropy source will panic.

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.

Processor	Temperature	Voltage	Clock Speed	Cache (per core)

Intel Xeon® Gold 6426Y (Sapphire Rapids/Golden Cove)	-40°C - +85°C	unspecified	2.50 GHz	**L1 Cache:** 80 KB per core **L2 Cache:** 2 MB per core **L3 Cache:** 37.5 MB shared

Table 2 – Operating Environment and Conditions

Configuration Settings

Table 2 summarize the configuration settings used by with CPU Jitter. These settings must be preserved to comply with the CPU Jitter ESV certificate.

Parameter	Value	Location
JENT_MEMORY_BLOCKS	64	jitterentropy.c
JENT_MEMORY_BLOCKSIZE	32	jitterentropy.c
JENT_MEMORY_ACCESSLOOPS	128	jitterentropy.c
JENT_RCT_CUTOFF	30	jitterentropy.c
JENT_APT_CUTOFF	325	jitterentropy.c
JENT_RCT_CUTOFF_PERMANENT	60	jitterentropy.c
JENT_APT_CUTOFF_PERMANENT	355	jitterentropy.c
JENT_APT_WINDOW_SIZE	512	jitterentropy.c

Table 3 - Configuration Paramteres

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.

Conceptual Interfaces

The entropy source provides the following interfaces:

- jent_kcapi_random(): 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 SP 800-90B.
- jent_lfsr_time(): Obtains raw noise data for testing purposes. This interface corresponds to the GetNoise() conceptual interface from SP 800-90B.

Min-Entropy Rate

The noise source provides an entropy rate of $h_{submitter} = 1$ bit per 64-bit time delta. The entropy source collects 64 time deltas of 64 bits each (4096 bits) as input to the LFSR conditioning component. This corresponds to 64 bits of entropy. The output entropy rate of the LFSR is assessed to be 57 bits of entropy per 64 bit output.

As a result, the entropy source provides 57 bits of entropy per 64 bit output.

Health Tests

The IBM QRadar Kernel CPU Jitter RNG implements the following health tests:

- Stuck Test
- Repetition Count Test
- Adaptive Proportion Test

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 $2^{-30} \leq \alpha \leq 2^{-40}$. When an intermittent failure is detected, the CPU Jitter RNG is automatically reset (which includes clearing the entropy pool and resetting the DRBG conditioning component), and the caller is notified of this failure. The only exception to this rule is during the start-up tests, where intermittent failures will be treated as permanent.

Permanent failures are characterized by a false positive probability of $\alpha_p = 2^{-6}$. When a permanent failure is detected, the CPU Jitter RNG is also reset, but the Linux kernel that contains this entropy source immediately enters the error state. In practice, this results in a kernel panic.

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

The stuck test is considered non-permanent, as positive stuck tests will be registered but will not immediately halt the entropy source.

Start-up tests conduct the same set and parameters of the continuous health tests on 1024 samples of noise data. The data is discarded after the start-up tests have completed successfully. Any health test failure during the start-up tests will always be treated as a permanent failure, which results in the permanent shutdown of the entropy source.

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. Similar to the start-up tests, the data used for the on-demand health tests are discarded after successful completion.

When entropy is requested from through the jent_kcapi_random() function, the return code indicates the status of the entropy source. A return code of -EAGAIN indicates an intermittent health test failure, whereas a return code of -EFAULT indicates a permanent health test failure.

The following error codes can occur:

-1: Entropy collector is NULL. The reference to the rand_data structure is invalid; the entropy collector must be allocated.
-2: RCT failure. The Repetition Count Test (and/or stuck test) has failed. The entropy source will remain in a failure state until it is successfully re-initialized.
-3: APT failure. The Adaptive Proportion Test has failed. The entropy source will remain in a failure state until it is successfully re-initialized.

Maintenance

There are no maintenance requirements for the IBM QRadar Kernel CPU Jitter RNG module.

User Verification

IBM QRadar developed and maintains their Kernel CPU Jitter RNG Module for their own use and thus do not provide any user or module developer guidance.

Vendor Permissions and Relationship

IBM QRadar restricts reuse and other vendors can only use the certificate with written and signed permission from the IBM QRadar's point of contact (as listed on the ESV certificate).