**# Title:**

Describe the DRBG algorithm used in the Arm RNDR instruction.

**# Status:**

Draft

**# Document:**

UEFI Specification 2.10

**# License:**

SPDX-License-Identifier: CC-BY-4.0

**# Submitter:**

* Arm (Sami Mujawar, Pierre Gondois, Jose Marinho, Samer El-Haj-Mahmoud)
* TianoCore Community (<https://www.tianocore.org>)

**# Summary of the change**

The Arm RNDR instruction outputs the result of a DRBG.  
The Arm Architecture states that the DRBG algorithm should be compliant with NIST SP800-90A, while not mandating a particular algorithm, so as to be inclusive of different geographies.  
  
This ECR introduces a default GUID that the EFI\_RNG\_PROTOCOL implementations, that directly expose the output of the RNDR via GetRNG, will adopt.  
The platform port should overwrite this default value to represent the concrete DRBG algorithm for the RNDR realization on that platform. The default value is provided as a fallback option in case the platform port is incomplete.  
This default value will still be used by OVMF.

**# Benefits of the change**

Create a meaningful fallback DRBG algorithm identification for the EFI\_RNG\_PROTOCOL implementations exposing the Arm RNDR instruction.

**# Impact of the change**

None as the EFI\_RNG\_PROTOCOL implementation exposing the Arm RNDR instruction has not been merged upstream in EDK2 yet.

**# Detailed description of the change [normative updates]**

* Insertions highlighted
* Removals in ~~red~~

### 37.5.4. EFI RNG Algorithm Definitions

**Summary**

This sub-section provides EFI\_GUID values for a selection of EFI\_RNG\_PROTOCOL algorithms. The algorithms listed are optional, not meant to be exhaustive and may be augmented by vendors or other industry standards.

The “raw” algorithm, when supported, is intended to provide entropy directly from the source, without it going through some deterministic random bit generator.

**Prototype**

*#define EFI\_RNG\_ALGORITHM\_SP800\_90\_HASH\_256\_GUID \*

{0xa7af67cb, 0x603b, 0x4d42,\

{0xba, 0x21, 0x70, 0xbf, 0xb6, 0x29, 0x3f, 0x96}}

*#define EFI\_RNG\_ALGORITHM\_SP800\_90\_HMAC\_256\_GUID \*

{0xc5149b43, 0xae85, 0x4f53,\

{0x99, 0x82, 0xb9, 0x43, 0x35, 0xd3, 0xa9, 0xe7}}

*#define EFI\_RNG\_ALGORITHM\_SP800\_90\_CTR\_256\_GUID \*

{0x44f0de6e, 0x4d8c, 0x4045, \

{0xa8, 0xc7, 0x4d, 0xd1, 0x68, 0x85, 0x6b, 0x9e}}

*#define EFI\_RNG\_ALGORITHM\_X9\_31\_3DES\_GUID \*

{0x63c4785a, 0xca34, 0x4012,\

{0xa3, 0xc8, 0x0b, 0x6a, 0x32, 0x4f, 0x55, 0x46}}

*#define EFI\_RNG\_ALGORITHM\_X9\_31\_AES\_GUID \*

{0xacd03321, 0x777e, 0x4d3d,\

{0xb1, 0xc8, 0x20, 0xcf, 0xd8, 0x88, 0x20, 0xc9}}

*#define EFI\_RNG\_ALGORITHM\_RAW \*

{0xe43176d7, 0xb6e8, 0x4827,\

{0xb7, 0x84, 0x7f, 0xfd, 0xc4, 0xb6, 0x85, 0x61}}

*#define EFI\_RNG\_ALGORITHM\_ARM\_RNDR \*

{0x43d2fde3, 0x9d4e, 0x4d79,\

{0x02, 0x96, 0xa8, 0x9b, 0xca, 0x78, 0x08, 0x41}}