Broadcom

Marvell

SUSE

NVME Host Parameters for Fibre Channel Adapters

Version 0.80

4/26/2021

Version History

Version	Date	Notes
0.30	August 10, 2020	First public draft.
0.40	February 24, 2021	Added the following variables: NvmeHostID, NvmeHostNQN. Updated Variable GUID section. Added System Variables vs Port Variables section. Added Default Host ID and Host NQN section.
0.50	April 2, 2021	Added creation and direction info for each variable. Updated Default Host ID and Host NQN section to state that most systems will use the default Host ID and Host NQN. Updated Variable Initialization section. Updated Variable Attributes section. Fixed formatting error.
0.60	April 8, 2021	Updated Variable Initialization section. Added more detail and examples to Default Host ID and Host NQN section.
0.70	April 10, 2021	Fixed a typo.
0.80	April 26, 2021	Rewrote Overview section. Moved Default Host NQN and Host ID section to beginning of doc. Added section numbers. Added UEFI Variable based Host NQN and Host ID section. Updated Title page.

1 Overview

This document describes how the NVME Host NQN and Host ID are defined for FC adapters. If the OS is booted from an FC NVME storage device, the UEFI drivers and the OS should use the same Host NQN and Host ID. The Host NQN and Host ID can be defined using one of the following methods:
-Default values based on the system UUID.
-UEFI variables created by the UEFI driver or the OS.

It is expected that most systems will use the default values based on the system UUID method. The default method "works" without any additional interaction or knowledge between the UEFI drivers and the OS. It also creates a minimalistic and singular system which eases migration if a future standardized solution comes into existence.

In the absence of other documented methods, the methods described in this document will be the preferred way to define the Host NQN and Host ID.

2 Default Host NQN and Host ID

If NVME UEFI variables are not defined, the system UUID will be used to create the default Host ID and Host NQN. The system UUID is a 16 byte value. The system UUID is normally found in the SMBIOS table. Examples of the default Host ID and Host NQN are shown below:

Section 7.2.1 in DSP0134_3.4.0 gives the following example to understand byte order of the UUID structure:

> The UUID {00112233-4455-6677-8899-aabbccddeeff} would thus be represented as:
>
> > 33 22 11 00 55 44 77 66 88 99 aa bb cc dd ee ff
>
> If the value is all FFh, the ID is not currently present in the system, but it can be set. If the value is all 00h, the ID is not present in the system

Using the DSP0134 example, the variables would be set as:

Host ID: 33 22 11 00 55 44 77 66 88 99 aa bb cc dd ee ff

> e.g. The UUID and the Host ID are treated as byte streams with each UUID byte copied 1:1 at the same byte offset to the Host ID field.

Host NQN: nqn.2014-08.org.nvmexpress:uuid:00112233-4455-6677-8899-aabbccddeeff

> e.g. The UUID structure per RFC 4122 is parsed by element, elements separated by dashes, multibyte values read per field endianness and converted to ASCII hex.

If the SMBIOS field indicates that the ID is not currently present in the system, the UEFI driver should revert to adapter port-specific variables.

It is expected that most systems will support the SMBIOS field and use the default Host ID and Host NQN. Non-default Host IDs and Host NQNs are only required if the NVME storage array is configured for a previously defined Host ID or Host NQN or the platform does not supply a SMBIOS UUID value.

3 UEFI Variable based Host NQN and Host ID

In some cases, it might be necessary to pass Host NQN and Host ID information between the UEFI drivers and the OS. The user might define a port specific Host NQN using a UEFI configuration tool. The OS installer might want to use a custom Host NQN.

UEFI variables are name/value pairs that are used in the UEFI environment. UEFI variables can be volatile or persistent. UEFI variables can be used to pass information between the UEFI environment and the OS.

3.1 Variable Definitions

NvmeHostID

This variable contains the NVME Host ID of the system. The variable name is case sensitive. This variable can be used by FC, Ethernet, and other transports. This variable will be used by the OS to pass system Host ID info to the UEFI driver.

Type: Array of bytes
Size: 16 bytes
Persistent: Yes
Name Example: NvmeHostID
Value Example: 00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff

NvmeHostID:ss:bb:dd:ff

This variable contains the NVME Host ID of the adapter port. The variable name is case sensitive. The variable name contains the PCIe Segment, Bus, Device and Function numbers, in hex. Hex digits ‘a’ – ‘f’ are lower case. There can be multiple instances of this variable (one per adapter port). This variable can be used by FC, Ethernet, and other transports. This variable will be used by the UEFI driver to pass port specific Host ID info to the OS.

Type: Array of bytes
Size: 16 bytes
Persistent: No
Name Example: NvmeHostID:00:a0:10:00
Value Example: 00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff
PCIe Address Info:
ss: Segment
bb: Bus
dd: Device
ff: Function

NvmeHostNQN

This variable contains the NVME Host NQN of the system. The variable name is case sensitive. This variable can be used by FC, Ethernet, and other transports. This variable will be used by the OS to pass system Host NQN info to the UEFI driver.

Type: ASCII string, Null terminated
Size: Variable
Persistent: Yes
Name Example: NvmeHostNQN
Value Example: nqn.2014-08.org.nvmexpress:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

NvmeHostNQN:ss:bb:dd:ff

This variable contains the NVME Host NQN of the adapter port. The variable name is case sensitive. The variable name contains the PCIe Segment, Bus, Device and Function numbers, in hex. Hex digits ‘a’ – ‘f’ are lower case. There can be multiple instances of this variable (one per adapter port). This variable can be used by FC, Ethernet, and other transports. This variable will be used by the UEFI driver to pass port specific Host NQN info to the OS.

Type: ASCII string, Null terminated
Size: Variable
Persistent: No
Name Example: NvmeHostNQN:00:a0:10:00
Value Example: nqn.2014-08.org.nvmexpress:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
PCIe Address Info:
ss: Segment
bb: Bus
dd: Device
ff: Function

3.2 System Variables vs Port Variables

The NVME spec. supports multiple Host NQNs and Host IDs per system. Some implementations might not support multiple Host NQNs and Host IDs. To handle all cases, the following logic is used to find the Host NQN and Host ID:

If (port variables defined)

    Use port variables: NvmeHostID:ss:bb:dd:ff, NvmeHostNQN:ss:bb:dd:ff

Else If (system variables defined)

    Use system variables: NvmeHostID, NvmeHostNQN

Else

    // NVME variables not defined
    Use system UUID to create default Host ID and Host NQN.

3.3 Variable Attributes

UEFI variables use a 32-bit Attribute field. This bit field defines the attributes of the variable. The Attribute bit fields are defined below:

```
#define EFI_VARIABLE_NON_VOLATILE                 0x00000001  // NV
#define EFI_VARIABLE_BOOTSERVICE_ACCESS           0x00000002  // BS
#define EFI_VARIABLE_RUNTIME_ACCESS               0x00000004  // RT
```

The NvmeHostID:ss:bb:dd:ff and NvmeHostNQN:ss:bb:dd:ff variables are volatile. The variables will have the BS and RT bits set. The NV bit will be clear.

The NvmeHostID and NvmeHostNQN variables are persistent. The variables will have the NV, BS and RT bits set.

3.4 Variable GUID

To avoid variable name conflicts, the UEFI environment supports multiple namespaces. There is one Global Namespace that is defined in the UEFI spec. Variables in the Global Namespace must be defined in the UEFI spec.

Initially, the NVME variables defined in this document will use a new namespace GUID. The new GUID is defined below:

// {d5d46f52-086a-41a8-bee7-14f74e8d1382}

static const GUID <<name>> =

{ 0xd5d46f52, 0x086a, 0x41a8, { 0xbe, 0xe7, 0x14, 0xf7, 0x4e, 0x8d, 0x13, 0x82 } };

Future versions of the UEFI spec. might contain the NVME variables. In this case, the Global Namespace GUID will be used to access the NVME variables. This means the UEFI and OS drivers will need to support the New GUID and Global Namespace GUID. The logic for accessing the NVME variables is defined below:

If (NVME variables defined in Global Namespace GUID)

    Use the Global Namespace GUID

Else

    Use the New GUID

3.5 Variable Initialization

The NVME Port variables will be created/set by the UEFI driver. The variables are volatile. The NVME Port variables will be initialized each time the system boots. If an adapter is added/removed/moved, the variables will be updated when the system boots.

The NVME System variables will be created/set by the OS. The variables are persistent. The NVME System variables will typically be initialized during the OS install process.