Full credit is given to the above companies including the OS that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'nvme-id-ctrl.1'

$ man nvme-id-ctrl.1

NVME-ID-CTRL(1)   NVMe Manual   NVME-ID-CTRL(1)

NAME

    nvme-id-ctrl - Send NVMe Identify Controller, return result and

    structure

SYNOPSIS

    nvme id-ctrl <device> [-v | --vendor-specific] [-b | --raw-binary]

             [-o <fmt> | --output-format=<fmt>]

DESCRIPTION

    For the NVMe device given, sends an identify controller command and

    provides the result and returned structure.

    The <device> parameter is mandatory and may be either the NVMe

    character device (ex: /dev/nvme0), or a namespace block device (ex:

    /dev/nvme0n1).

    On success, the structure may be returned in one of several ways

    depending on the option flags; the structure may be parsed by the

    program or the raw buffer may be printed to stdout.

OPTIONS

    -b, --raw-binary

Print the raw buffer to stdout. Structure is not parsed by program.

This overrides the vendor specific and human readable options.

-v, --vendor-specific

In addition to parsing known fields, this option will dump the

vendor specific region of the structure in hex with ascii

interpretation.

-H, --human-readable

This option will parse and format many of the bit fields into

human-readable formats.

-o <format>, --output-format=<format>

Set the reporting format to normal, json, or binary. Only one

output format can be used at a time.

EXAMPLES

? Has the program interpret the returned buffer and display the known

fields in a human readable format:

    # nvme id-ctrl /dev/nvme0

? In addition to showing the known fields, has the program to display

the vendor unique field:

    # nvme id-ctrl /dev/nvme0 --vendor-specific

    # nvme id-ctrl /dev/nvme0 -v

The above will dump the vs buffer in hex since it doesn?t know how

to interpret it.

? Have the program return the raw structure in binary:

    # nvme id-ctrl /dev/nvme0 --raw-binary > id_ctrl.raw

    # nvme id-ctrl /dev/nvme0 -b > id_ctrl.raw

It is probably a bad idea to not redirect stdout when using this

mode.

? Alternatively you may want to send the data to another program that

can parse the raw buffer.

    # nvme id-ctrl /dev/nvme0 --raw-binary | nvme_parse_id_ctrl

The parse program in the above example can be a program that shows

the structure in a way you like. The following program is such an

example that will parse it and can accept the output through a

pipe, '|', as shown in the above example, or you can 'cat' a saved

output buffer to it.

```c
/* File: nvme_parse_id_ctrl.c */
#include <linux/nvme.h>
#include <stdio.h>
#include <unistd.h>
int main(int argc, char **argv)
{
        unsigned char buf[sizeof(struct nvme_id_ctrl)];
        struct nvme_id_ctrl *ctrl = (struct nvme_id_ctrl *)buf;
        if (read(STDIN_FILENO, buf, sizeof(buf)))
                return 1;
        printf("vid   : %#x\n", ctrl->vid);
        printf("ssvid : %#x\n", ctrl->ssvid);
        return 0;
}
```

NVME

   Part of the nvme-user suite

NVMe   06/23/2023   NVME-ID-CTRL(1)