NAME

ddb – daos debug tool

SYNOPSIS

ddb [OPTIONS]

DESCRIPTION

The DAOS Debug Tool (ddb) allows a user to navigate through and modify a file in the VOS format. It offers both a command line and interactive shell mode. If neither a single command or '-f' option is provided, then the tool will run in interactive mode. In order to modify the VOS file, the '-w' option must be included.

If the command requires it, the VOS file must be provided with the parameter --vos-path. The VOS file will be opened before any commands are executed. See the commandspecific help for details.

OPTIONS

Application Options

−w, **−−write_mode**
: Open the vos file in write mode.

−f, **−−cmd_file**
: Path to a file containing a sequence of ddb commands to execute.

−p, **−−db_path**
: Path to the sys db.

−s, **−−vos_path**
: Path to the VOS file to open.

−v, **−−version**
: Show version

−−debug
: Without this option, the console log level is set to ERROR. With this option, the console log level is set to INFO. For more detailed logs, provide the --log_dir path to log to a file.

−−log_dir
: If provided, the file log level is set to DEBUG and are written to files in the provided directory. The log file for the CLI is 'ddb-cli.log' and the log file for the engine is 'ddb-engine.log'. If the directory does not exist or is not writable, an error is returned.

Help Options

−h, **−−help**
: Show this help message

ARGUMENTS

Application Arguments

ddb_command
: Optional ddb command to run. If not provided, the tool will run in interactive mode.

ddb_command_args
: Arguments for the ddb command to run. If not provided, the command will be run without any arguments.

COMMANDS

Available Commands

ls List containers, objects, dkeys, akeys, and values

open Opens the vos file at <path>. The '-w' option allows for modifying the vos file with the rm, load, commit_ilog, etc commands. The path <path> should be an absolute path to the pool shard. Part of the path is used to determine what the pool uuid is.

version
Print ddb version

close Close the currently opened vos pool shard

superblock_dump
Dump the pool superblock information

value_dump
Dump a value to the screen or file. The vos path should be a complete path, including the akey and if the value is an array value it should include the extent. If a path to a file was provided then the value will be written to the file, else it will be printed to the screen.

rm Remove a branch of the VOS tree. The branch can be anything from a container and everything under it, to a single value.

value_load
Load a value to a vos path. This can be used to update the value of an existing key, or create a new key. The <src> is a path to a file on the file system. The <dst> is a vos tree path to a value where the data will be loaded. If the <dst> path currently exists, then the destination path must match the value type, meaning, if the value type is an array, then the path must include the extent, otherwise, it must not.

ilog_dump
Dump the ilog

ilog_commit
Process the ilog

ilog_clear
Remove all the ilog entries

dtx_dump
Dump the dtx tables

dtx_cmt_clear
Clear the dtx committed table

smd_sync
Restore the SMD file with backup from blob

vea_dump
Dump information from the vea about free regions

vea_update
Alter the VEA tree to mark a region as free.

dtx_act_commit
Mark the active dtx entry as committed

dtx_act_abort
Mark the active dtx entry as aborted

feature Manage vos pool features

rm_pool
Remove a vos pool.

dtx_act_discard_invalid
Discard the active DTX entry's records if invalid.

dev_list
List all devices

dev_replace
Replace an old device with a new unused device

dtx_stat
Print statistic on the DTX entries

prov_mem
Prepare the memory environment for md-on-ssd mode

dtx_aggr
Aggregate DTX entries until a given aggregation commit time or date

quit exit the shell

PATH

VOS Tree Path

Many of the commands take a VOS tree path. The format for this path is [cont]/[obj]/[dkey]/[akey]/[extent].

cont The full container uuid.

obj The object id.

keys (akey, dkey)
There are multiple types of keys:

* **string keys** are simply the string value. If the size of the key is greater than strlen(key), then the size is included at the end of the string value. Example: 'akey{5}' is the key: akey with a null terminator at the end.
* **number keys** are formatted as '{[type]: NNN}' where type is 'uint8, uint16, uint32, or uint64'. NNN can be a decimal or hex number. Example: '{uint32: 123456}'
* **binary keys** are formatted as '{bin: 0xHHH}' where HHH is the hex representation of the binary key. Example: '{bin: 0x1a2b}'

extent For array values in the format {lo-hi}.

Index Tree Path

To make it easier to navigate the tree, indexes can be used instead of the path part. The index is in the format [i]. Indexes and actual path values can be used together

Path Examples

VOS tree path examples:

```
/3550f5df-e6b1-4415-947e-82e15cf769af/939000573846355970.0.13.1/dkey/akey/[0
```

Index tree path examples:

```
[0]/[1]/[2]/[1]/[9]
```

Mixed tree path examples:

```
/[0]/939000573846355970.0.13.1/[2]/akey{5}/[0-1023]
```

LOGGING

By default, the console output is filtered to show only **ERROR** level messages and above. To adjust the verbosity or capture detailed logs for troubleshooting, use the following options:

Console Output

The console is designed to remain readable during interactive use.

* **Default:** Displays **ERROR** and **CRITICAL** logs.
* **--debug Flag:** Increases verbosity to include **INFO** logs.

Warning:
Even with *--debug* enabled, **DEBUG** level logs are suppressed on the console to prevent output overflow and maintain interactivity.

File Logging

For deep-dive troubleshooting, use the *--log_dir <path>* option. This sets the internal log level to **DEBUG** and redirects the full output to the specified directory.

ddb-cli.log:
Captures logs related to the command line interface.

ddb-engine.log:
Captures logs related to Core engine processes and logic.