# Troubleshooting

## DAOS Errors

DAOS error numbering starts at 1000. The most common errors are documented in the table below.

|  |  |  |
| --- | --- | --- |
| DAOS Error | Value | Description |
| DER\_NO\_PERM | 1001 | No permission |
| DER\_NO\_HDL | 1002 | Invalid handle |
| DER\_INVAL | 1003 | Invalid parameters |
| DER\_EXIST | 1004 | Entity already exists |
| DER\_NONEXIST | 1005 | The specified entity does not exist |
| DER\_UNREACH | 1006 | Unreachable node |
| DER\_NOSPACE | 1007 | No space left on storage target |
| DER\_ALREADY | 1008 | Operation already performed |
| DER\_NOMEM | 1009 | Out of memory |
| DER\_NOSYS | 1010 | Function not implemented |
| DER\_TIMEDOUT | 1011 | Time out |
| DER\_BUSY | 1012 | Device or resource busy |
| DER\_AGAIN | 1013 | Try again |
| DER\_PROTO | 1014 | Incompatible protocol |
| DER\_UNINIT | 1015 | Device or resource not initialized |
| DER\_TRUNC | 1016 | Buffer too short |
| DER\_OVERFLOW | 1017 | Data too long for defined data type or buffer size |
| DER\_CANCELED | 1018 | Operation canceled |
| DER\_OOG | 1019 | Out of group or member list |
| DER\_HG | 1020 | Transport layer mercury error |
| DER\_MISC | 1025 | Miscellaneous error |
| DER\_BADPATH | 1026 | Bad path name |
| DER\_NOTDIR | 1027 | Not a directory |
| DER\_EVICTED | 1032 | Rank has been evicted |
| DER\_DOS | 1034 | Denial of service |
| DER\_BAD\_TARGET | 1035 | Incorrect target for the RPC |
| DER\_HLC\_SYNC | 1037 | HLC synchronization error |
| DER\_IO | 2001 | Generic I/O error |
| DER\_ENOENT | 2003 | Entry not found |
| DER\_NOTYPE | 2004 | Unknown object type |
| DER\_NOSCHEMA | 2005 | Unknown object schema |
| DER\_KEY2BIG | 2012 | Key is too large |
| DER\_REC2BIG | 2013 | Record is too large |
| DER\_IO\_INVAL | 2014 | IO buffers can’t match object extents |
| DER\_EQ\_BUSY | 2015 | Event queue is busy |
| DER\_SHUTDOWN | 2017 | Service should shut down |
| DER\_INPROGRESS | 2018 | Operation now in progress |
| DER\_NOTREPLICA | 2020 | Not a service replica |
| DER\_CSUM | 2021 | Checksum error |
| DER\_REC\_SIZE | 2024 | Record size error |
| DER\_TX\_RESTART | 2025 | Transaction should restart |
| DER\_DATA\_LOSS | 2026 | Data lost or not recoverable |
| DER\_TX\_BUSY | 2028 | TX is not committed |
| DER\_AGENT\_INCOMPAT | 2029 | Agent is incompatible with libdaos |

When an operation fails, DAOS returns a negative DER error. For a full list of errors, please check <https://github.com/daos-stack/daos/blob/master/src/include/daos_errno.h> (DER\_ERR\_GURT\_BASE is equal to 1000, and DER\_ERR\_DAOS\_BASE is equal to 2000).

The function d\_errstr() is provided in the API to convert an error number to an error message.

## Log Files

On the server side, there are three log files created as part of normal server operations:

|  |  |  |
| --- | --- | --- |
| Component | Config Parameter | Example Config Value |
| Control Plane | control\_log\_file | /tmp/daos\_server.log |
| Data Plane | log\_file | /tmp/daos\_engine.\*.log |
| [Privileged Helper](https://daos-stack.github.io/admin/deployment/#elevated-privileges) | helper\_log\_file | /tmp/daos\_admin.log |
| agent | log\_file | /tmp/daos\_agent.log |

### Control Plane Log

The default log level for the control plane is INFO. The following levels may be set using the control\_log\_mask config parameter:

* DEBUG
* INFO
* ERROR

### Data Plane Log

Data Plane (daos\_engine) logging is configured on a per-instance basis. In other words, each section under the servers: section must have its own logging configuration. The log\_file config parameter is converted to a D\_LOG\_FILE environment variable value. For more detail, please see the [Debugging System](#debugging-system) section of this document.

### Privileged Helper Log

By default, the privileged helper only emits ERROR-level logging which is captured by the control plane and included in that log. If the helper\_log\_file parameter is set in the server config, then DEBUG-level logging will be sent to the specified file.

### Daos Agent Log

If the log\_file config parameter is set in the agent config, then DEBUG-level logging will be sent to the specified file.

## Debugging System

DAOS uses the debug system defined in [CaRT](https://github.com/daos-stack/daos/tree/master/src/cart), specifically the GURT library. Both server and client default log is stdout, unless otherwise set by D\_LOG\_FILE environment variable (client) or log\_file config parameter (server).

### Registered Subsystems/Facilities

The debug logging system includes a series of subsystems or facilities which define groups for related log messages (defined per source file). There are common facilities which are defined in GURT, as well as other facilities that can be defined on a per-project basis (such as those for CaRT and DAOS). DD\_SUBSYS can be used to set which subsystems to enable logging. By default all subsystems are enabled (“DD\_SUBSYS=all”).

* DAOS Facilities: common, tree, vos, client, server, rdb, pool, container, object, placement, rebuild, tier, mgmt, bio, tests
* Common Facilities (GURT): MISC, MEM
* CaRT Facilities: RPC, BULK, CORPC, GRP, LM, HG, ST, IV

### Priority Logging

All macros that output logs have a priority level, shown in descending order below.

* D\_FATAL(fmt, …) FATAL
* D\_CRIT(fmt, …) CRIT
* D\_ERROR(fmt, …) ERR
* D\_WARN(fmt, …) WARN
* D\_NOTE(fmt, …) NOTE
* D\_INFO(fmt, …) INFO
* D\_DEBUG(mask, fmt, …) DEBUG

The priority level that outputs to stderr is set with DD\_STDERR. By default in DAOS (specific to the project), this is set to CRIT (“DD\_STDERR=CRIT”) meaning that all CRIT and more severe log messages will dump to stderr. However, this is separate from the priority of logging to “/tmp/daos.log”. The priority level of logging can be set with D\_LOG\_MASK, which by default is set to INFO (“D\_LOG\_MASK=INFO”), which will result in all messages excluding DEBUG messages being logged. D\_LOG\_MASK can also be used to specify the level of logging on a per-subsystem basis as well (“D\_LOG\_MASK=DEBUG,MEM=ERR”).

### Debug Masks/Streams:

DEBUG messages account for a majority of the log messages, and finer-granularity might be desired. Mask bits are set as the first argument passed in D\_DEBUG(mask, …). To accomplish this, DD\_MASK can be set to enable different debug streams. Similar to facilities, there are common debug streams defined in GURT, as well as other streams that can be defined on a per-project basis (CaRT and DAOS). All debug streams are enabled by default (“DD\_MASK=all”).

* DAOS Debug Masks:
  + md = metadata operations
  + pl = placement operations
  + mgmt = pool management
  + epc = epoch system
  + df = durable format
  + rebuild = rebuild process
  + daos\_default = (group mask) io, md, pl, and rebuild operations
* Common Debug Masks (GURT):
  + any = generic messages, no classification
  + trace = function trace, tree/hash/lru operations
  + mem = memory operations
  + net = network operations
  + io = object I/Otest = test programs

### Common Use Cases

* Generic setup for all messages (default settings)
* $ D\_LOG\_MASK=DEBUG  
  $ DD\_SUBSYS=all  
  $ DD\_MASK=all
* Disable all logs for performance tuning
* $ D\_LOG\_MASK=ERR -> will only log error messages from all facilities  
  $ D\_LOG\_MASK=FATAL -> will only log system fatal messages
* Disable a noisy debug logging subsystem
* $ D\_LOG\_MASK=DEBUG,MEM=ERR -> disables MEM facility by  
  restricting all logs from that facility to ERROR or higher priority only
* Enable a subset of facilities of interest
* $ DD\_SUBSYS=rpc,tests  
  $ D\_LOG\_MASK=DEBUG -> required to see logs for RPC and TESTS  
  less severe than INFO (the majority of log messages)
* Fine-tune the debug messages by setting a debug mask
* $ D\_LOG\_MASK=DEBUG  
  $ DD\_MASK=mgmt -> only logs DEBUG messages related to pool  
  management

Refer to the DAOS Environment Variables document for more information about the debug system environment.

## Common DAOS Problems

### Incompatible Agent

When DER\_AGENT\_INCOMPAT is received, it means that the client library libdaos.so is likely mismatched with the DAOS Agent. The libdaos.so, DAOS Agent and DAOS Server must be built from compatible sources so that the GetAttachInfo protocol is the same between each component. Depending on your situation, you will need to either update the DAOS Agent or the libdaos.so to the newer version in order to maintain compatibility with each other.

### HLC Sync

When DER\_HLC\_SYNC is received, it means that sender and receiver HLC timestamps are off by more than maximum allowed system clock offset (1 second by default).

In order to correct this situation synchronize all server clocks to the same reference time, using services like NTP.

### Shared Memory Errors

When DER\_SHMEM\_PERMS is received it means that this I/O Engine lacked the permissions to access the shared memory megment left behind by a previous run of the I/O Engine on the same machine. This happens when the I/O Engine fails to remove the shared memory segment upon shutdown, and, there is a mismatch between the user/group used to launch the I/O Engine between these successive runs. To remedy the problem, manually identify the shared memory segment and remove it.

Issue ipcs to view the Shared Memory Segments. The output will show a list of segments organized by key.

$ipcs  
  
------ Message Queues --------  
key msqid owner perms used-bytes messages  
  
------ Shared Memory Segments --------  
key shmid owner perms bytes nattch status  
0xffffffff 49938432 root 666 40 0  
0x10242048 98598913 jbrosenz 660 1048576 0  
0x10242049 98631682 jbrosenz 660 1048576 0  
  
------ Semaphore Arrays --------  
key semid owner perms nsems

Shared Memory Segments with keys [0x10242048 .. (0x10242048 + number of I/O Engines running)] are the segments that must be removed. Use ipcrm to remove the segment.

For example, to remove the shared memory segment left behind by I/O Engine instance 0, issue:

sudo ipcrm -M 0x10242048

To remove the shared memory segment left behind by I/O Engine instance 1, issue:

sudo ipcrm -M 0x10242049

## Bug Report

Bugs should be reported through our issue tracker[[1]](#footnote-40) with a test case to reproduce the issue (when applicable) and debug logs.

After creating a ticket, logs should be gathered from the locations described in the [Log Files](#log-files) section of this document and attached to the ticket.

To avoid problems with attaching large files, please attach the logs in a compressed container format, such as .zip or .tar.bz2.

1. https://jira.hpdd.intel.com [↑](#footnote-ref-40)