# DAOS Environment Variables

This section lists the environment variables used by DAOS.

!!! warning Many of these variables are used for development purposes only, and may be removed or changed in the future.

The description of each variable follows the following format:

* Short description
* Type
* The default behavior if not set.
* A longer description if necessary

Type is defined by this table:

|  |  |
| --- | --- |
| Type | Values |
| BOOL | 0 means false; any other value means true |
| BOOL2 | no means false; any other value means true |
| BOOL3 | set to empty, or any value means true; unset means false |
| INTEGER | Non-negative decimal integer |
| STRING | String |

## Server environment variables

Environment variables in this section only apply to the server side.

|  |  |
| --- | --- |
| Variable | Description |
| RDB\_ELECTION\_TIMEOUT | Raft election timeout used by RDBs in milliseconds. INTEGER. Default to 7000 ms. |
| RDB\_REQUEST\_TIMEOUT | Raft request timeout used by RDBs in milliseconds. INTEGER. Default to 3000 ms. |
| DAOS\_REBUILD | Determines whether to start rebuilds when excluding targets. BOOL2. Default to true. |
| DAOS\_MD\_CAP | Size of a metadata pmem pool/file in MBs. INTEGER. Default to 128 MB. |
| DAOS\_START\_POOL\_SVC | Determines whether to start existing pool services when starting a daos\_server. BOOL. Default to true. |
| CRT\_DISABLE\_MEM\_PIN | Disable memory pinning workaround on a server side. BOOL. Default to 0. |
| DAOS\_SCHED\_PRIO\_DISABLED | Disable server ULT prioritizing. BOOL. Default to 0. |
| DAOS\_SCHED\_RELAX\_MODE | The mode of CPU relaxing on idle. “disabled”:disable relaxing; “net”:wait on network request for INTVL; “sleep”:sleep for INTVL. STRING. Default to “net” |
| DAOS\_SCHED\_RELAX\_INTVL | CPU relax interval in milliseconds. INTEGER. Default to 1 ms. |

## Server and Client environment variables

Environment variables in this section apply to both the server side and the client side.

|  |  |
| --- | --- |
| Variable | Description |
| FI\_OFI\_RXM\_USE\_SRX | Enable shared receive buffers for RXM-based providers (verbs, tcp). BOOL. Auto-defaults to 1. |
| FI\_UNIVERSE\_SIZE | Sets expected universe size in OFI layer to be more than expected number of clients. INTEGER. Auto-defaults to 2048. |

## Client environment variables

Environment variables in this section only apply to the client side.

|  |  |
| --- | --- |
| Variable | Description |
| FI\_MR\_CACHE\_MAX\_COUNT | Enable MR caching in OFI layer. Recommended to be set to 0 (disable) when CRT\_DISABLE\_MEM\_PIN is NOT set to 1. INTEGER. Default to unset. |

## Debug System (Client & Server)

|  |  |
| --- | --- |
| Variable | Description |
| D\_LOG\_FILE | DAOS debug logs (both server and client) are written to stdout by default. The debug location can be modified by setting this environment variable (“D\_LOG\_FILE=/tmp/daos\_debug.log”). |
| D\_LOG\_FILE\_APPEND\_PID | If set and not 0, causes the main PID to be appended at the end of D\_LOG\_FILE path name (both server and client). |
| D\_LOG\_STDERR\_IN\_LOG | If set and not 0, causes stderr messages to be merged in D\_LOG\_FILE. |
| D\_LOG\_SIZE | DAOS debug logs (both server and client) have a 1GB file size limit by default. When this limit is reached, the current log file is closed and renamed with a .old suffix, and a new one is opened. This mechanism will repeat each time the limit is reached, meaning that available saved log records could be found in both ${D\_LOG\_FILE} and last generation of ${D\_LOG\_FILE}.old files, to a maximum of the most recent 2\*D\_LOG\_SIZE records. This can be modified by setting this environment variable (“D\_LOG\_SIZE=536870912”). Sizes can also be specified in human-readable form using k, m, g, K, M, and G. The lower-case specifiers are base-10 multipliers and the upper case specifiers are base-2 multipliers. |
| DD\_SUBSYS | Used to specify which subsystems to enable. DD\_SUBSYS can be set to individual subsystems for finer-grained debugging (“DD\_SUBSYS=vos”), multiple facilities (“DD\_SUBSYS=eio,mgmt,misc,mem”), or all facilities (“DD\_SUBSYS=all”) which is also the default setting. If a facility is not enabled, then only ERR messages or more severe messages will print. |
| DD\_STDERR | Used to specify the priority level to output to stderr. Options in decreasing priority level order: FATAL, CRIT, ERR, WARN, NOTE, INFO, DEBUG. By default, all CRIT and more severe DAOS messages will log to stderr (“DD\_STDERR=CRIT”), and the default for CaRT/GURT is FATAL. |
| D\_LOG\_MASK | Used to specify what type/level of logging will be present for either all of the registered subsystems or a select few. Options in decreasing priority level order: FATAL, CRIT, ERR, WARN, NOTE, INFO, DEBUG. DEBUG option is used to enable all logging (debug messages as well as all higher priority level messages). Note that if D\_LOG\_MASK is not set, it will default to logging all messages excluding debug (“D\_LOG\_MASK=INFO”). Example: “D\_LOG\_MASK=DEBUG”. This will set the logging level for all facilities to DEBUG, meaning that all debug messages, as well as higher priority messages will be logged (INFO, NOTE, WARN, ERR, CRIT, FATAL). Example 2: “D\_LOG\_MASK=DEBUG,MEM=ERR,RPC=ERR”. This will set the logging level to DEBUG for all facilities except MEM & RPC (which will now only log ERR and higher priority level messages, skipping all DEBUG, INFO, NOTE & WARN messages) |
| DD\_MASK | Used to enable different debug streams for finer-grained debug messages, essentially allowing the user to specify an area of interest to debug (possibly involving many different subsystems) as opposed to parsing through many lines of generic DEBUG messages. All debug streams will be enabled by default (“DD\_MASK=all”). Single debug masks can be set (“DD\_MASK=trace”) or multiple masks (“DD\_MASK=trace,test,mgmt”). Note that since these debug streams are strictly related to the debug log messages, DD\_LOG\_MASK must be set to DEBUG. Priority messages higher than DEBUG will still be logged for all facilities unless otherwise specified by D\_LOG\_MASK (not affected by enabling debug masks). |