

# ***Solution for Unknown Status Occurring on Overloaded Servers***

---

---

This technical note addresses the issue of some servers shutting down when overloaded. It also provides a workaround for resolving the problem.

## **Servers affected**

- Job Queue Manager (JQM) server
- Destination Manager (DSM) server
- Configuration Manager (CM) server

## **Problem**

When a user attempts to execute an HP Output Server command on a batch of jobs, the JQM, CM and DSM servers go into an 'unknown' status, or they generate a core dump. Executing an HP Output Server command on a batch of jobs forces the servers to attempt to allocate more memory than allowed by the specified data segment stack size.

## **Solution**

To avoid receiving an 'unknown' status, perform one of the following options:

- Increase the data segment stack size.
- Use indexed attributes for HP Output Server commands.

### **Increase the data segment stack size**

Increase the data segment stack size by using the `ulimit` command. Each operating system uses a separate data segment stack size. Find the current data segment stack size by using the following command:

```
ulimit -a
```

Example

Output for `ulimit -a` command on the Solaris platform:

```
$ulimit -a
```

*Solution*

```
time (seconds) - unlimited
file (blocks) - unlimited
data (kbytes) - 96000
stack (kbytes) - 8192
coredump (blocks) - unlimited
nofiles (descriptors) - 256
vmemory (kbytes) - unlimited
```

---

NOTE: Check the help information of ulimit command on each operating system to change the data segment stack size.

---

## Use indexed attributes for HP Output Server commands

When a command is entered, the server buffers all the data for the request and sends it as output. Users should avoid making command requests that could overload the server.

For example, `pdls -c job -a all jqm:`

```
orpdls -c job -a all -x"-job-status retained" jqm:
```

Following is a list of indexed attributes used for HP Output Server commands:

Server Name	Table Name	Indexed Attribute
Accounting and Inventory Manager (AIM)	table_001	COLUMN_001_001, COLUMN_001_003
	table_002	COLUMN_002_001, COLUMN_002_002, COLUMN_002_003, COLUMN_002_005, COLUMN_002_006, COLUMN_002_010
	table_003	COLUMN_003_001, COLUMN_003_002, COLUMN_003_007, COLUMN_003_008
	table_004	COLUMN_004_001
	table_005	COLUMN_005_001
	table_006	COLUMN_006_001, COLUMN_006_002, COLUMN_006_003
	table_007	COLUMN_007_001, COLUMN_007_002, COLUMN_007_003
	table_008	COLUMN_008_001, COLUMN_008_002
	table_009	COLUMN_009_001, COLUMN_009_002, COLUMN_009_003, COLUMN_009_004, COLUMN_009_005, COLUMN_009_006, COLUMN_009_007, COLUMN_009_008, COLUMN_009_009, COLUMN_009_010, COLUMN_009_011, COLUMN_009_012
	table_010	COLUMN_010_001
	table_011	COLUMN_011_001
	table_012	COLUMN_012_001, COLUMN_012_002, COLUMN_012_007, COLUMN_012_009
	table_013	COLUMN_013_006
	table_014	COLUMN_014_005, COLUMN_014_009
	table_015	COLUMN_015_003

Server Name	Table Name	Indexed Attribute	
	table_016	COLUMN_016_003	
	table_017	COLUMN_017_004	
	table_018	COLUMN_018_001	
	table_019	COLUMN_019_001, COLUMN_019_002	
	schema	version	
Privilege Manager (PM)	ctdb_tbl	dbkey	
	gdb_group	group_name	
	gdb_members	group_name, member_name	
	pdb_tbl	dbkey, crypt	
	ptdb_tbl	dbkey	
Event manager (EM)	clients	client_id	
	subscriptions	client_id	
CM	002 (printer)	__name, printer-realization, printer-connection-mode, managing-server, template, template-classification, template-type, template-device-name, physical-device-type	
	003 (server)	__name, managing-server, template, template-classification, template-type, template-device-name	
	051 (queue)	__name, scheduler-assigned, managing-server, template, template-classification, template-type, template-device-name	
	101 (transformer)	__name, managing-server, template, template-classification, template-type, template-device-name	
	102 (form)	__name, managing-server, template, template-classification, template-type, template-device-name	
	103 (paging system)	__name, managing-server, template, template-classification, template-type, template-device-name	
	105 (device type)	__name, managing-server, template, template-classification, template-type, template-device-name, device-type-manufacturer, device-type-model	
	107 (capabilities)	__name, managing-server, template, template-classification, template-type, template-device-name	
	Delivery Manager (DLM)	001 (job)	__name, current-job-state, submission-time
		003 (server)	__name

Server Name	Table Name	Indexed Attribute
	016 (dlm job)	__name, current-dlm-job-state, dlm-job-submission-time
JQM	001 (job)	__name, job-name, job-owner, destination, current-job-state, queue-assigned, job-completion-status, physical-printer-assigned, job-retention-time
	002 (printer)	__name, printer-realization, queue-supported
	003 (server)	__name
	051 (queue)	__name, scheduler-assigned
	107 (capabilities)	__name
DSM	002 (printer)	__name, queue-supported, printer-connection-mode, physical-device-type
	003 (server)	__name
	101 (transformer)	__name
	107 (capabilities)	__name

NOTE: For information on which attributes need to be indexed and which do not, read the readme file located at \$DAZEL\_HOME/nls/oid/C/README.DPAOIDS

## HP-UX specific case

HP-UX uses different ARENA's to allocate memory to each thread. This leads to a potential increase in the memory usage for each operation, as the requests get served by different threads each time. But this increase is limited to the number of threads used by the server, as the memory growth becomes significantly reduced or eliminated.

This behavior of HP-UX memory allocator can be altered using the `_M_ARENA_OPTS` environment variable.

Example

```
export _M_ARENA_OPTS=1:8
```

`_M_ARENA_OPTS` is interpreted as:

```
_M_ARENA_OPTS=#arenas:#extensionpages
```

In the above example, ARENAS is the number of ARENA's to be used for the application while *extensionpages* determines the number of pages to be expanded each time. The default value for `_M_ARENA_OPTS` is 8:32. To resolve the problem of server crash due to memory issue, use a single ARENA and set the *extensionpage* based on your requirements

To prevent an unknown status from occurring on different servers (due to a memory issue), refer to the examples given below:

Example 1

In the JQM server during heavy activity, the `pdls` command is used on a job and the memory consumption exceeds the data segment limit of the system. If the jobs are submitted while querying the job status, the server consumes more memory and fails.

Restarting JQM server causes the server to use a single ARENA for all the memory allocation requests preventing the server memory from increasing for any of the subsequent requests.

---

NOTE: The memory growth for a specific server can also be restricted by setting the `-server-environment-variables` attribute for that server. Setting the `_M_ARENA_OPTS` variable may adversely affect the performance of an application, as per HP-UX manual for malloc, use of different ARENA for each thread leads to better server performance.

---

The `_M_ARENA_OPTS` can also be set for a specific server (JQM or any other) by using `-server-environment-variables` attribute. You can set this attribute by using the following command:

```
config_server -u -x"-server-environment-variables \"_M_ARENA_OPTS=1:8\" jqm
```

---

NOTE: For more information on `_M_ARENA_OPTS`, refer to the following link:  
<http://devsrc1.external.hp.com/STK/cgi-bin/man2html?debug=0&manpage=/usr/share/man/man3.Z/malloc.3c>

---

### Example 2

In the DSM server, use of an advanced template (with 150 destinations) increases the memory size to 200MB. The combination of memory usage and APF destination causes this increase. But the estimate of memory usage of these destinations is 200k to 500k per physical destination. The memory increase is caused due to a specific feature present in the HP-UX heap manager that sometimes causes DSM server to gain memory. By default, the HP-UX heap manager uses sub-heaps, which it calls ARENAS. These ARENAS are thread specific, and there are eight of them (default). Unless changed, this causes HP Output Server processes, such as DSM server, to get larger than needed.

To make the HP-UX heap manager behave like any other heap manager, execute the following command:

```
config_server -u -x"-server-environment-variables \"_M_ARENA_OPTS=1:32\" dsm
```

This command changes the heap manager's behavior for DSM server. With this setting of `_M_ARENA_OPTS`, there is only one ARENA (like all the other heap managers.) This setting is recommended for any of the DSMs that run APF physicals.

### Example 3

In the CM server, executing the `pdconfig` command results in substantial growth of memory. A memory spike occurs on the server when the `'pdconfig -d -c /tmp/x.cf'` command is executed. This increases the memory growth to 121MB. A feature in HP-UX heap manager (malloc and free) creates separate heaps for each thread in the multi-threaded program. This problem occurs when different `RPC WorkerThreads` allocate memory to serve the `pdls -c class RPC` that comes from the `pdconfig` client.

The large size of the configuration causes CM server to allocate about 40MB to serve one `pdls -cp ids_ccm:rpc` call. All of this memory is allocated in one ARENA since it is all allocated only on one thread. When another `pdls -cp ids_ccm:rpc` call is made, this RPC is served by another RPC worker thread. This causes the CM server to allocate another 40MB in another heap ARENA. In this case, the original 40MB, which was allocated and later cleared, is not reused. Instead the heap manager asks HP-UX for another 40MB of virtual memory. Eventually `pdls` calls fill up the ARENAS to exceed the 128MB per-process data limit.

To rectify this problem, the behavior of the HP-UX heap manager should be modified by adding the following environment variable in the `HostConfig.sgml` entry for the environment-variables host attribute:

```
_M_ARENA_OPTS=1:8
```

*Conclusion*

This command forces the HP-UX heap manager to use a single ARENA. So when the CM server is run, the first `pdconfig -d` command invocation raises the memory size of the CM server from 14MB to 58MB while the subsequent `pdconfig -d` invocations do not raise the memory size above 58MB.

---

NOTE: The use of the `_M_ARENA_OPTS` environment variable is described in the HP-UX `malloc` man page.

---

## Conclusion

The servers buffer all the output for a given command operation in their memory prior to marshalling the data to the server client. This causes the servers to fail when the number of jobs is large. The only way to avoid the failure is to increase the data segment size (this can only reduce the probability of server crash) or avoid issuing a command on any batch of jobs.