qreslib

Version 1.0.0

Generated by Doxygen 1.5.6

Mon Jan 12 15:36:44 2009

Contents

Chapter 1

Todo List

File qres_lib.h Check status of library/QRESMod inside each function

File qres_lib.h qres library level calls take pid_t; qres supervisor-filtered calls take struct task_struct (they use task info in order to enforce qossup policy); qres lowest level calls (cbs) take struct cbs_struct;

File qres_lib.h Supervisor (Tom) Invariant-based Ctrl and Pred (Tom) Translation of qosres to IRIS (Luca) Port to 2.4.27 Test inside 2.4.27 Overhead measurements

Global qres_create_server Implement the QRES_WATCHDOG feature.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Module Documentation

4.1 QoS Resource Reservation Library

Allows applications to take advantage of Resource Reservation scheduling services available within the kernel when the QRES_MOD module is loaded.

Files

- file qres_lib.h

  QRES Library Application Programming Interface.

Functions

- qos_rv qres_init (void)

  Initializes the QoS RES library.

- qos_rv qres_cleanup (void)

  Cleanup resources associated to the QoS RES Library.

- qos_rv qres_create_server (qres_params_t *p_params, qres_sid_t *p_sid)

  : in the following functions, if pid == 0 -> apply to itself

- qos_rv qres_attach_thread (qres_sid_t server_id, pid_t pid, tid_t tid)

  Attach a task to an already existing server.

- qos_rv qres_detach_thread (qres_sid_t sid, pid_t pid, tid_t tid)

  Detach from the QoS Server and possibly destroy it.

- qos_rv qres_destroy_server (qres_sid_t sid)

  Detach all processes from a server and destroy it.

- qos_rv qres_get_sid (pid_t pid, tid_t tid, qres_sid_t *p_sid)

  Get a thread server id, or QOS_E_NOT_FOUND if it is not attached to any server.

- qos_rv qres_set_bandwidth (qres_sid_t sid, qos_bw_t bw)
  Change bandwidth.
- qos_rv qres_get_params (qres_sid_t sid, qres_params_t *p_params)
  Retrieve QoS scheduling parameters.
- qos_rv qres_set_params (qres_sid_t sid, qres_params_t *p_params)
  Change QoS scheduling parameters.
- qos_rv qres_get_bandwidth (qres_sid_t sid, float *bw)
  Get the bandwidth of the server with the supplied ID.
- qos_rv qres_module_exists ()
  Detect module in /proc/modules.
- qos_rv qres_get_exec_time (qres_sid_t sid, qres_time_t *exec_time, qres_time_t *abs_time)
  Retrieve time info relative to the current server.
- qos_rv qres_get_curr_budget (qres_sid_t sid, qres_time_t *curr_budget)
  Retrieve remaining budget for the current server instance.
- qos_rv qres_get_next_budget (qres_sid_t sid, qres_time_t *next_budget)
  Retrieve the budget that is likely to be used for the very next server instance.
- qos_rv qres_get_appr_budget (qres_sid_t sid, qres_time_t *appr_budget)
  Retrieve the budget that has been approved for the subsequent server instances.
- qos_rv qres_get_deadline (qres_sid_t sid, struct timespec *p_deadline)
  Retrieve current scheduling deadline.
- qos_rv qres_set_weight (qres_sid_t sid, unsigned int weight)
  Set scheduling weight (i.e., used by shrub).
- qos_rv qres_get_weight (qres_sid_t sid, unsigned int *p_weight)
  Retrieve scheduling weight (i.e., used by shrub).

4.1.1 Detailed Description

Allows applications to take advantage of Resource Reservation scheduling services available within the kernel when the QRES_MOD module is loaded.

4.1.2 Function Documentation

4.1.2.1 qos_rv qres_attach_thread (qres_sid_t *server_id*, pid_t *pid*, tid_t *tid*)

Attach a task to an already existing server.

Parameters:

pid The process ID of the task to affect, or zero for the current process

tid The thread ID of the thread to affect, or zero for the current thread (only valid if pid is zero as well)

server_id The identifier of the server to which the task has to be attached

4.1.2.2 qos_rv qres_cleanup (void)

Cleanup resources associated to the QoS RES Library.

After this call, it is not possible to use any qres_xx() call

4.1.2.3 qos_rv qres_create_server (qres_params_t * *p_params*, qres_sid_t * *p_sid*)

: in the following functions, if pid == 0 -> apply to itself

Create a new server with specified parameters.

Change scheduling policy of the specified process to benefit of QoS support into the kernel with provided parameters.

Parameters:

p_params If the QRES_F_PERSISTENT flag is set in p_params->flags, then the server is not automatically destroyed after detach of the last thread. Instead, it keeps existing, where further threads may be attached to it by using the a qres_sid_t value for identification.

p_sid Pointer to a qres_sid_t variable that is filled with the server ID of the created server. This ID is needed to perform any further operation on the created server (attaching/detaching threads, getting or setting parameters, destroying the server).

Note:

If the QRES_WATCHDOG has been enabled in the module configuration, then, if a persistent server remains empty (with no attached threads) for too much time, the system destroys it automatically. This is a precaution for avoiding persistence of "unreachable" servers in the system, due to programming bugs or application crashes.

Todo

Implement the QRES_WATCHDOG feature.

4.1.2.4 qos_rv qres_destroy_server (qres_sid_t *sid*)

Detach all processes from a server and destroy it.

Detach all attached threads from a QRES server, then destroy it. This causes the QRES system to free any resources previously associated to the destroyed server, and to the attached tasks, if any.

Note:

After a successfull completition of this call, the specified server ID is not valid anymore. Though, it may be reused later by the system to identify a new server created subsequently through a qres_create_server() call.

4.1.2.5 qos_rv qres_detach_thread (qres_sid_t *sid*, pid_t *pid*, tid_t *tid*)

Detach from the QoS Server and possibly destroy it.

Detach the specified process from the QoS server and return the scheduling policy to the standard default system scheduler behaviour. If no more processes reside in the server, destroy resources associated to it.

For parameters specification, please see qres_attach_to_server()

See also:

qres_attach_to_server()

4.1.2.6 qos_rv qres_get_appr_budget (qres_sid_t *sid*, qres_time_t * *appr_budget*)

Retrieve the budget that has been approved for the subsequent server instances.

Whenever a change of budget is requested, either as a result of a qres_set_params() or as a result of the creation of a new server, each server that has been approved a budget increase from the supervisor may undergo (if such increase in not immediately and entirely available) a transitory period during which the current budget, as returned by qres_get_curr_budget(), may change multiple times, until it reaches the approved value.

4.1.2.7 qos_rv qres_get_bandwidth (qres_sid_t *sid*, float * *bw*)

Get the bandwidth of the server with the supplied ID.

4.1.2.8 qos_rv qres_get_exec_time (qres_sid_t *sid*, qres_time_t * *exec_time*, qres_time_t * *abs_time*)

Retrieve time info relative to the current server.

Parameters:

exec_time returns the sum of the execution times of all the processes belonging to the same server as the one identified by the sid parameter, measured since the call to * qres_create_server()

abs_time returns the absolute time, measured since some fixed point in the past (e.g. machine boot)

Returns:

The server execution time, of one of the QOS_E_ error codes if negative.

4.1.2.9 qos_rv qres_get_next_budget (qres_sid_t *sid*, qres_time_t * *next_budget*)

Retrieve the budget that is likely to be used for the very next server instance.

The value returned by this function may change asynchronously from, and in a way that depends on, the bandwidth changes requested on all the servers in the system.

4.1.2.10 qos_rv qres_get_params (qres_sid_t *sid*, qres_params_t * *p_params*)

Retrieve QoS scheduling parameters.

The returned values may be slightly different than those set through qres_set_params().

Note:

Returned budget values may be slightly higher than the ones set through qres_init_server() or qres_set_params(), because any granted (budget, period) pair corresponds to a qos_bw_t value (fixed-point representation of the ratio budget/period).

4.1.2.11 qos_rv qres_init (void)

Initializes the QoS RES library.

Checks that the kernel supports QoS Management, then initializes the QRES library.

Returns:

QOS_OK if initialization was succesful QOS_E_MISSING_COMPONENT if kernel does not support QoS management

Chapter 5

File Documentation

5.1 qres_lib.h File Reference

QRES Library Application Programming Interface.

```
#include <linux/aquosa/qos_types.h>
#include <linux/aquosa/qos_debug.h>
#include <linux/aquosa/qres_gw.h>
#include <errno.h>
```

Functions

- qos_rv qres_init (void)
  Initializes the QoS RES library.
- qos_rv qres_cleanup (void)
  Cleanup resources associated to the QoS RES Library.
- qos_rv qres_create_server (qres_params_t *p_params, qres_sid_t *p_sid)
  : in the following functions, if pid == 0 -> apply to itself
- qos_rv qres_attach_thread (qres_sid_t server_id, pid_t pid, tid_t tid)
  Attach a task to an already existing server.
- qos_rv qres_detach_thread (qres_sid_t sid, pid_t pid, tid_t tid)
  Detach from the QoS Server and possibly destroy it.
- qos_rv qres_destroy_server (qres_sid_t sid)
  Detach all processes from a server and destroy it.
- qos_rv qres_get_sid (pid_t pid, tid_t tid, qres_sid_t *p_sid)
  Get a thread server id, or QOS_E_NOT_FOUND if it is not attached to any server.
- qos_rv qres_set_bandwidth (qres_sid_t sid, qos_bw_t bw)

Change bandwidth.

- qos_rv qres_get_params (qres_sid_t sid, qres_params_t *p_params)
  Retrieve QoS scheduling parameters.
- qos_rv qres_set_params (qres_sid_t sid, qres_params_t *p_params)
  Change QoS scheduling parameters.
- qos_rv qres_get_bandwidth (qres_sid_t sid, float *bw)
  Get the bandwidth of the server with the supplied ID.
- qos_rv qres_module_exists ()
  Detect module in /proc/modules.
- qos_rv qres_get_exec_time (qres_sid_t sid, qres_time_t *exec_time, qres_time_t *abs_time)
  Retrieve time info relative to the current server.
- qos_rv qres_get_curr_budget (qres_sid_t sid, qres_time_t *curr_budget)
  Retrieve remaining budget for the current server instance.
- qos_rv qres_get_next_budget (qres_sid_t sid, qres_time_t *next_budget)
  Retrieve the budget that is likely to be used for the very next server instance.
- qos_rv qres_get_appr_budget (qres_sid_t sid, qres_time_t *appr_budget)
  Retrieve the budget that has been approved for the subsequent server instances.
- qos_rv qres_get_deadline (qres_sid_t sid, struct timespec *p_deadline)
  Retrieve current scheduling deadline.
- qos_rv qres_set_weight (qres_sid_t sid, unsigned int weight)
  Set scheduling weight (i.e., used by shrub).
- qos_rv qres_get_weight (qres_sid_t sid, unsigned int *p_weight)
  Retrieve scheduling weight (i.e., used by shrub).

5.1.1 Detailed Description

QRES Library Application Programming Interface.

Todo

Check status of library/QRESMod inside each function

Todo

qres library level calls take pid_t; qres supervisor-filtered calls take struct task_struct (they use task info in order to enforce qossup policy); qres lowest level calls (cbs) take struct cbs_struct;

Todo

Supervisor (Tom) Invariant-based Ctrl and Pred (Tom) Translation of qosres to IRIS (Luca) Port to 2.4.27 Test inside 2.4.27 Overhead measurements

Definition in file qres_lib.h.

Index