Subject: [RFC][PATCH][4/4] RSS controller documentation (
Posted by Balbir Singh on Sat, 24 Feb 2007 14:45:46 GMT

View Forum Message <> Reply to Message

```
---

Signed-off-by: <balbir@in.ibm.com>
---

 Documentation/memctlr.txt |   70 ++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 70 insertions(+)

diff -puN /dev/null Documentation/memctlr.txt
--- /dev/null 2007-02-02 22:51:23.000000000 +0530
+++ linux-2.6.20-balbir/Documentation/memctlr.txt 2007-02-24 19:41:23.000000000 +0530
@@ -0,0 +1,70 @@
+Introduction
+------------
+
+The memory controller is a controller module written under the containers
+framework. It can be used to limit the resource usage of a group of
+tasks grouped by the container.
+
+Accounting
+----------
+
+The memory controller tracks the RSS usage of the tasks in the container.
+The definition of RSS was debated on lkml in the following thread
+
+ http://lkml.org/lkml/2006/10/10/130
+
+This patch is flexible, it is easy to adapt the patch to any definition
+of RSS. The current accounting is based on the current definition of
+RSS. Each page mapped is charged to the container.
+
+The accounting is done at two levels, each process has RSS accounting in
+the mm_struct and in the container it belongs to. The mm_struct accounting
+is used when a task switches (migrates to a different) container(s). The
+accounting information for the task is subtracted from the source container
+and added to the destination container. If as result of the migration, the
+destination container goes over limit, no action is taken until some task
+in the destination container runs and tries to map a new page in its
+page table.
+
+The current RSS usage can be seen in the memcontrol_usage file. The value
+is in units of pages.
+
+Control
```

```
+-------
+
+The memcontrol_limit file allows the user to set a limit on the number of
+pages that can be mapped by the processes in the container. A special
+value of 0 (which is the default limit of any new container), indicates
+that the container can use unlimited amount of RSS.
+
+Reclaim
+-------
+
+When the limit set in the container is hit, the memory controller starts
+reclaiming pages belonging to the container (simulating a local LRU in
+some sense). isolate_lru_pages() has been modified to isolate lru
+pages belonging to a specific container. Parallel reclaims on the same
+container are not allowed, other tasks end up waiting for the any existing
+reclaim to finish.
+
+The reclaim code uses two internal knobs, retries and pushback. pushback
+specifies the percentage of memory to be reclaimed when the container goes
+over limit. The retries knob, controls how many times reclaim is retried
+before the task is killed (because reclaim failed).
+
+Shared pages are treated specially during reclaim. They are not force
+reclaimed, they are only unmapped from containers which are over limit.
+This ensures that other containers do not pay a penalty for a shared
+page being reclaimed when a paritcular container goes over its limit.
+
+NOTE: All limits are hard limits.
+
+Future Plans
+------------
+
+The current controller implements only RSS control. It is planned to add
+the following components
+
+1. Page Cache control
+2. mlock'ed memory control
+3. kernel memory allocation control (memory allocated on behalf of a task)
```

–

--
Warm Regards,
Balbir Singh