Subject: [patch01/05]:Containers(V2): Documentation
Posted by Rohit Seth on Wed, 20 Sep 2006 02:17:34 GMT
View Forum Message <> Reply to Message

This patch contains the documentation for containers.

Signed-of-by: Rohit Seth <rohitseth@google.com>

```
--- linux-2.6.18-rc6-mm2.org/Documentation/containers.txt 1969-12-31 16:00:00.000000000 -0800
+++ linux-2.6.18-rc6-mm2.ctn/Documentation/containers.txt 2006-09-19 18:28:04.000000000
-0700
@@ -0,0 +1,65 @@
+Containers allow different workloads to be run on the same platform with
+limits defined on per container basis.  This basically allows a single
+platform to be (soft) partitioned among different workloads (each of
+which could be running many tasks).  The limits could be amount of
+memory, number of tasks among other features.  These two features are
+already implemented in the patch set that I posted.  But it is possible
+to add other controllers like CPU that allows only finite amount of time
+to the processes belonging to a container.
+
+For example, users can run batch jobs like backups using tar, which if run
+uncontained could use lot of page cache, inside a container.  This way the
+memory footprint of the backup job can be contained.
+
+Currently we are tracking user memory (both file based
+and anonymous).  The memory handler is currently deactivating pages
+belonging to a container that has gone over the limit. Even though this
+allows containers to go over board their limits but 1- once they are
+over the limit then they run in degraded manner and 2- if there is any
+memory pressure then the (extra) pages belonging to this container are
+the prime candidates for swapping (for example).  The statistics that
+are shown in each container directory are the current values of each
+resource consumption.
+
+Configfs support is needed in kernel as the container's user interface is
+through configfs. So first enable CONFIG_CONFIGFS_FS and CONFIG_CONTAINERS
+and recompile the kernel.
+
+1- Mount a configfs (for example):
+ mount -t configfs none /mnt/configfs
+   This will create a /mnt/configfs mount point.
+
+2- As the support of containers is built into kernel, so the mount point
+   /mnt/configfs will automatically contain a directory "containers"
+
+3- Create a container by name test_container
+ cd /mnt/configfs/containers
+ mkdir test_container
+
+All the current implemented attributes in the kernel will show up in the
+directory /configfs/containers/test_container
+
+4- Add a task to container
+ cd /mnt/configfs/cotnainers/test_container
+ echo <pid> > addtask
+
+Now the <pid> and its subsequently forked children will belong to container
+test_container.
+
+5- Remove a task from container
+ echo <pid> > rmtask
+
+6- Set a page limit for the container
+ echo some_number_of_pages > page_limit
+
+7- Read the id for the container
+ cat id
+
+8- Get the statistics for this container
+ cat num* (will print active pages, anon_pages, file_pages, num_files,
+   and num_task)
+ cat *hits (will print page_limit_hits and task_limit_hits: the number
+  of times container has gone over page_limit and task_limit)
+9- Freeing a container
+ cd /mnt/configfs/containers/
+ rmdir test_container
```