Subject: [RFC][PATCH 1/3] Containers: Pagecache accounting and control subsystem (v3)
Posted by Vaidyanathan Srinivas on Wed, 23 May 2007 14:50:54 GMT
View Forum Message <> Reply to Message

Pagecache controller setup

This patch basically adds user interface files in container fs
similar to the rss control files.

pagecache_usage, pagecache_limit and pagecache_failcnt are added
to each container. All units are 'pages' as in rss controller.

pagecache usage is all file backed pages used by the container
which includes swapcache as well.

Separate res_counter for pagecache has been added.

Signed-off-by: Vaidyanathan Srinivasan <svaidy@linux.vnet.ibm.com>
---
 mm/rss_container.c | 42 ++
 1 file changed, 42 insertions(+)

--- linux-2.6.20.orig/mm/rss_container.c
+++ linux-2.6.20/mm/rss_container.c
@@ -16,6 +16,7 @@

 struct rss_container {
  struct res_counter res;
+ struct res_counter pagecache_res;
  struct list_head inactive_list;
  struct list_head active_list;
  atomic_t rss_reclaimed;
@@ -266,6 +267,7 @@ static int rss_create(struct container_s
   return -ENOMEM;

  res_counter_init(&rss->res);
+ res_counter_init(&rss->pagecache_res);
  INIT_LIST_HEAD(&rss->inactive_list);
  INIT_LIST_HEAD(&rss->active_list);
  rss_container_attach(rss, cont);
@@ -308,6 +310,21 @@ static ssize_t rss_read_reclaimed(struct
    ppos, buf, s - buf);
 }

+static ssize_t pagecache_read(struct container *cont, struct cftype *cft,
+ struct file *file, char __user *userbuf,

```
+  size_t nbytes, loff_t *ppos)
+{
+ return res_counter_read(&rss_from_cont(cont)->pagecache_res,
+    cft->private, userbuf, nbytes, ppos);
+}
+
+static ssize_t pagecache_write(struct container *cont, struct cftype *cft,
+  struct file *file, const char __user *userbuf,
+  size_t nbytes, loff_t *ppos)
+{
+ return res_counter_write(&rss_from_cont(cont)->pagecache_res,
+    cft->private, userbuf, nbytes, ppos);
+}

 static struct cftype rss_usage = {
  .name = "rss_usage",
@@ -333,6 +350,25 @@ static struct cftype rss_reclaimed = {
  .read = rss_read_reclaimed,
 };

+static struct cftype pagecache_usage = {
+ .name = "pagecache_usage",
+ .private = RES_USAGE,
+ .read = pagecache_read,
+};
+
+static struct cftype pagecache_limit = {
+ .name = "pagecache_limit",
+ .private = RES_LIMIT,
+ .read = pagecache_read,
+ .write = pagecache_write,
+};
+
+static struct cftype pagecache_failcnt = {
+ .name = "pagecache_failcnt",
+ .private = RES_FAILCNT,
+ .read = pagecache_read,
+};
+
 static int rss_populate(struct container_subsys *ss,
   struct container *cont)
 {
@@ -346,6 +382,12 @@ static int rss_populate(struct container
   return rc;
  if ((rc = container_add_file(cont, &rss_reclaimed)) < 0)
   return rc;
+ if ((rc = container_add_file(cont, &pagecache_usage)) < 0)
+  return rc;
```

```
+ if ((rc = container_add_file(cont, &pagecache_failcnt)) < 0)
+  return rc;
+ if ((rc = container_add_file(cont, &pagecache_limit)) < 0)
+  return rc;

  return 0;
 }
```