Subject: [-mm PATCH 3/7] Memory controller accounting setup
Posted by Balbir Singh on Wed, 04 Jul 2007 22:21:49 GMT
View Forum Message <> Reply to Message

Basic setup routines, the mm_struct has a pointer to the container that
it belongs to and the the page has a meta_page associated with it.

Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>
---

include/linux/memcontrol.h | 32 ++++++++++++++++++++++++++++++++
include/linux/mm_types.h | 4 +++
include/linux/sched.h | 4 +++
kernel/fork.c | 10 ++++++---
mm/memcontrol.c | 47 ++---
5 files changed, 91 insertions(+), 6 deletions(-)

diff -puN include/linux/memcontrol.h~mem-control-accounting-setup include/linux/memcontrol.h
--- linux-2.6.22-rc6/include/linux/memcontrol.h~mem-control-accounting-setup 2007-07-04 15:05:24.000000000 -0700
+++ linux-2.6.22-rc6-balbir/include/linux/memcontrol.h 2007-07-04 15:05:24.000000000 -0700
@@ -15,5 +15,37 @@
#ifndef _LINUX_MEMCONTROL_H
#define _LINUX_MEMCONTROL_H

+struct mem_container;
+struct meta_page;
+
+#ifdef CONFIG_CONTAINER_MEM_CONT
+
+extern void mm_init_container(struct mm_struct *mm, struct task_struct *p);
+extern void mm_free_container(struct mm_struct *mm);
+extern void page_assign_meta_page(struct page *page, struct meta_page *mp);
+extern struct meta_page *page_get_meta_page(struct page *page);
+
+#else /* CONFIG_CONTAINER_MEM_CONT */
+static inline void mm_init_container(struct mm_struct *mm,
+ struct task_struct *p)
+{
+}
+
+static inline void mm_free_container(struct mm_struct *mm)
+{
+}
+
+static inline void page_assign_meta_page(struct page *page,
+ struct meta_page *mp)

```
+{
+}
+
+static inline struct meta_page *page_get_meta_page(struct page *page)
+{
+ return NULL;
+}
+
+#endif /* CONFIG_CONTAINER_MEM_CONT */
+
 #endif /* _LINUX_MEMCONTROL_H */

diff -puN include/linux/mm_types.h~mem-control-accounting-setup include/linux/mm_types.h
--- linux-2.6.22-rc6/include/linux/mm_types.h~mem-control-accounting-setup 2007-07-04
15:05:24.000000000 -0700
+++ linux-2.6.22-rc6-balbir/include/linux/mm_types.h 2007-07-04 15:05:24.000000000 -0700
@@ -5,6 +5,7 @@
 #include <linux/threads.h>
 #include <linux/list.h>
 #include <linux/spinlock.h>
+#include <linux/memcontrol.h>

 struct address_space;

@@ -83,6 +84,9 @@ struct page {
  unsigned int gfp_mask;
  unsigned long trace[8];
 #endif
+#ifdef CONFIG_CONTAINER_MEM_CONT
+ struct meta_page *meta_page;
+#endif
 };

 #endif /* _LINUX_MM_TYPES_H */
diff -puN include/linux/sched.h~mem-control-accounting-setup include/linux/sched.h
--- linux-2.6.22-rc6/include/linux/sched.h~mem-control-accounting-setup 2007-07-04
15:05:24.000000000 -0700
+++ linux-2.6.22-rc6-balbir/include/linux/sched.h 2007-07-04 15:05:24.000000000 -0700
@@ -87,6 +87,7 @@ struct sched_param {
 #include <linux/timer.h>
 #include <linux/hrtimer.h>
 #include <linux/task_io_accounting.h>
+#include <linux/memcontrol.h>

 #include <asm/processor.h>

@@ -416,6 +417,9 @@ struct mm_struct {
  /* aio bits */
```

```
  rwlock_t  ioctx_list_lock;
  struct kioctx  *ioctx_list;
+#ifdef CONFIG_CONTAINER_MEM_CONT
+ struct mem_container *mem_container;
+#endif
 };

 struct sighand_struct {
diff -puN kernel/fork.c~mem-control-accounting-setup kernel/fork.c
--- linux-2.6.22-rc6/kernel/fork.c~mem-control-accounting-setup 2007-07-04 15:05:24.000000000
-0700
+++ linux-2.6.22-rc6-balbir/kernel/fork.c 2007-07-04 15:05:24.000000000 -0700
@@ -330,7 +330,7 @@ static inline void mm_free_pgd(struct mm

 #include <linux/init_task.h>

-static struct mm_struct * mm_init(struct mm_struct * mm)
+static struct mm_struct * mm_init(struct mm_struct * mm, struct task_struct *p)
 {
  atomic_set(&mm->mm_users, 1);
  atomic_set(&mm->mm_count, 1);
@@ -347,11 +347,14 @@ static struct mm_struct * mm_init(struct
  mm->ioctx_list = NULL;
  mm->free_area_cache = TASK_UNMAPPED_BASE;
  mm->cached_hole_size = ~0UL;
+ mm_init_container(mm, p);

  if (likely(!mm_alloc_pgd(mm))) {
   mm->def_flags = 0;
   return mm;
  }
+
+ mm_free_container(mm);
  free_mm(mm);
  return NULL;
 }
@@ -366,7 +369,7 @@ struct mm_struct * mm_alloc(void)
  mm = allocate_mm();
  if (mm) {
   memset(mm, 0, sizeof(*mm));
-  mm = mm_init(mm);
+  mm = mm_init(mm, current);
  }
  return mm;
 }
@@ -380,6 +383,7 @@ void fastcall __mmdrop(struct mm_struct
 {
  BUG_ON(mm == &init_mm);
```

```
  mm_free_pgd(mm);
+ mm_free_container(mm);
  destroy_context(mm);
  free_mm(mm);
 }
@@ -500,7 +504,7 @@ static struct mm_struct *dup_mm(struct t
  mm->token_priority = 0;
  mm->last_interval = 0;

- if (!mm_init(mm))
+ if (!mm_init(mm, tsk))
   goto fail_nomem;

  if (init_new_context(tsk, mm))
diff -puN mm/memcontrol.c~mem-control-accounting-setup mm/memcontrol.c
--- linux-2.6.22-rc6/mm/memcontrol.c~mem-control-accounting-setup 2007-07-04
15:05:24.000000000 -0700
+++ linux-2.6.22-rc6-balbir/mm/memcontrol.c 2007-07-04 15:05:24.000000000 -0700
@@ -15,6 +15,7 @@
 #include <linux/res_counter.h>
 #include <linux/memcontrol.h>
 #include <linux/container.h>
+#include <linux/mm.h>

 struct container_subsys mem_container_subsys;

@@ -33,6 +34,13 @@ struct mem_container {
   * the counter to account for memory usage
   */
  struct res_counter res;
+ /*
+  * Per container active and inactive list, similar to the
+  * per zone LRU lists.
+  * TODO: Consider making these lists per zone
+  */
+ struct list_head active_list;
+ struct list_head inactive_list;
 };

 /*
@@ -54,6 +62,31 @@ static inline struct mem_container *mem_
     css);
 }

+void mm_init_container(struct mm_struct *mm, struct task_struct *p)
+{
+ struct mem_container *mem;
+
```

```
+ mem = mem_container_from_cont(task_container(p,
+     mem_container_subsys_id));
+ css_get(&mem->css);
+ mm->mem_container = mem;
+}
+
+void mm_free_container(struct mm_struct *mm)
+{
+ css_put(&mm->mem_container->css);
+}
+
+void page_assign_meta_page(struct page *page, struct meta_page *mp)
+{
+ page->meta_page = mp;
+}
+
+struct meta_page *page_get_meta_page(struct page *page)
+{
+ return page->meta_page;
+}
+
 static ssize_t mem_container_read(struct container *cont, struct cftype *cft,
    struct file *file, char __user *userbuf, size_t nbytes,
    loff_t *ppos)
@@ -89,13 +122,21 @@ static struct cftype mem_container_failc
  .read = mem_container_read,
 };

+static struct mem_container init_mem_container;
+
 static int mem_container_create(struct container_subsys *ss,
     struct container *cont)
 {
  struct mem_container *mem;

- mem = kzalloc(sizeof(struct mem_container), GFP_KERNEL);
- if (!mem)
+ if (unlikely((cont->parent) == NULL)) {
+  mem = &init_mem_container;
+  css_get(&mem->css);
+  init_mm.mem_container = mem;
+ } else
+  mem = kzalloc(sizeof(struct mem_container), GFP_KERNEL);
+
+ if (mem == NULL)
   return -ENOMEM;

  res_counter_init(&mem->res);
```

```
@@ -137,5 +178,5 @@ struct container_subsys mem_container_su
  .create = mem_container_create,
  .destroy = mem_container_destroy,
  .populate = mem_container_populate,
- .early_init = 0,
+ .early_init = 1,
 };
```

–

--
Warm Regards,
Balbir Singh
Linux Technology Center
IBM, ISTL

_____

Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers