Subject: [patch02/05]: Containers(V2)- Generic Linux kernel changes
Posted by Rohit Seth on Wed, 20 Sep 2006 02:18:42 GMT
View Forum Message <> Reply to Message

This patch contains changes to generic part of kernel code. These changes tracks events like new task creating, task's exit, new page allocation (both file and anonymous) etc.

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

```
 fs/inode.c              |    3 +++
 include/linux/fs.h      |    5 +++++
 include/linux/mm_inline.h |    4 ++++
 include/linux/mm_types.h  |    4 ++++
 include/linux/sched.h     |    6 ++++++
 init/Kconfig            |    8 ++++++++
 kernel/Makefile         |    1 +
 kernel/exit.c           |    2 ++
 kernel/fork.c           |    9 +++++++++
 mm/Makefile             |    2 ++
 mm/filemap.c            |    4 ++++
 mm/page_alloc.c         |    3 +++
 mm/rmap.c               |    8 +++++++-
 mm/swap.c               |    1 +
 mm/vmscan.c             |    1 +
 15 files changed, 60 insertions(+), 1 deletion(-)


--- linux-2.6.18-rc6-mm2.org/include/linux/mm_inline.h 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/include/linux/mm_inline.h 2006-09-14 15:09:08.000000000 -0700
@@ -1,9 +1,11 @@
+#include <linux/container.h>

 static inline void
 add_page_to_active_list(struct zone *zone, struct page *page)
 {
  list_add(&page->lru, &zone->active_list);
  zone->nr_active++;
+ container_inc_activepage_count(page);
 }

 static inline void
@@ -23,6 +25,7 @@ add_page_to_inactive_list_tail(struct zo
 static inline void
 del_page_from_active_list(struct zone *zone, struct page *page)
 {
+ container_dec_activepage_count(page);
  list_del(&page->lru);
```

```
  zone->nr_active--;
 }
@@ -41,6 +44,7 @@ del_page_from_lru(struct zone *zone, str
  if (PageActive(page)) {
   __ClearPageActive(page);
   zone->nr_active--;
+  container_dec_activepage_count(page);
  } else {
   zone->nr_inactive--;
  }
--- linux-2.6.18-rc6-mm2.org/include/linux/fs.h 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/include/linux/fs.h 2006-09-14 15:09:08.000000000 -0700
@@ -449,6 +449,7 @@ struct address_space_operations {
 };

 struct backing_dev_info;
+struct container_struct;
 struct address_space {
  struct inode  *host;  /* owner: inode, block_device */
  struct radix_tree_root page_tree; /* radix tree of all pages */
@@ -465,6 +466,10 @@ struct address_space {
  struct backing_dev_info *backing_dev_info; /* device readahead, etc */
  spinlock_t  private_lock; /* for use by the address_space */
  struct list_head private_list; /* ditto */
+#ifdef CONFIG_CONTAINERS
+ struct container_struct *ctn; /* Pointer to container */
+ struct list_head ctn_mapping_list; /* List of files belonging to same container*/
+#endif
  struct address_space *assoc_mapping; /* ditto */
 } __attribute__((aligned(sizeof(long))));
  /*
--- linux-2.6.18-rc6-mm2.org/include/linux/sched.h 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/include/linux/sched.h 2006-09-14 15:09:08.000000000 -0700
@@ -298,6 +298,8 @@ typedef unsigned long mm_counter_t;
   (mm)->hiwater_vm = (mm)->total_vm; \
 } while (0)

+struct container_struct;
+
 struct mm_struct {
  struct vm_area_struct * mmap;  /* list of VMAs */
  struct rb_root mm_rb;
@@ -1046,6 +1048,10 @@ struct task_struct {
 #ifdef CONFIG_TASK_DELAY_ACCT
  struct task_delay_info *delays;
 #endif
+#ifdef CONFIG_CONTAINERS
+ struct container_struct *ctn;
```

```
+ struct list_head ctn_task_list; /*List of processes belonging to container */
+#endif
 };

 static inline pid_t process_group(struct task_struct *tsk)
--- linux-2.6.18-rc6-mm2.org/include/linux/mm_types.h 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/include/linux/mm_types.h 2006-09-14 15:15:30.000000000 -0700
@@ -6,6 +6,7 @@
 #include <linux/list.h>
 #include <linux/spinlock.h>

+struct container_struct;
 struct address_space;

 /*
@@ -48,6 +49,9 @@ struct page {
  struct list_head lru;  /* Pageout list, eg. active_list
       * protected by zone->lru_lock !
       */
+#ifdef CONFIG_CONTAINERS
+ struct container_struct *ctn; /* Pointer to container, may be NULL */
+#endif
  /*
   * On machines where all RAM is mapped into kernel address space,
   * we can simply calculate the virtual address. On machines with
--- linux-2.6.18-rc6-mm2.org/mm/filemap.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/mm/filemap.c 2006-09-14 15:09:08.000000000 -0700
@@ -30,6 +30,7 @@
 #include <linux/security.h>
 #include <linux/syscalls.h>
 #include <linux/cpuset.h>
+#include <linux/container.h>
 #include "filemap.h"
 #include "internal.h"

@@ -126,6 +127,7 @@ void __remove_from_page_cache(struct pag
  page->mapping = NULL;
  mapping->nrpages--;
  __dec_zone_page_state(page, NR_FILE_PAGES);
+ container_dec_filepage_count(page);
 }
 EXPORT_SYMBOL(__remove_from_page_cache);

@@ -461,6 +463,8 @@ int add_to_page_cache(struct page *page,
   }
   write_unlock_irq(&mapping->tree_lock);
   radix_tree_preload_end();
+  if (!error)
```

```
+   container_inc_filepage_count(mapping, page);
  }
  return error;
 }
--- linux-2.6.18-rc6-mm2.org/init/Kconfig 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/init/Kconfig 2006-09-18 10:48:08.000000000 -0700
@@ -560,6 +560,14 @@ config STOP_MACHINE
  depends on (SMP && MODULE_UNLOAD) || HOTPLUG_CPU
  help
    Need stop_machine() primitive.
+
+config CONTAINERS
+ bool "Containers"
+ def_bool y
+ depends on CONFIGFS_FS
+ help
+   This option allows grouping of resources like memory and tasks. It
+   depends on CONFIGFS_FS support in psuedo filesystem.
 endmenu

 menu "Block layer"
--- linux-2.6.18-rc6-mm2.org/mm/Makefile 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/mm/Makefile 2006-09-14 15:09:08.000000000 -0700
@@ -29,3 +29,5 @@ obj-$(CONFIG_MEMORY_HOTPLUG) += memory_h
 obj-$(CONFIG_FS_XIP) += filemap_xip.o
 obj-$(CONFIG_MIGRATION) += migrate.o
 obj-$(CONFIG_SMP) += allocpercpu.o
+obj-$(CONFIG_CONTAINERS) += container.o container_mm.o
+
--- linux-2.6.18-rc6-mm2.org/mm/page_alloc.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/mm/page_alloc.c 2006-09-14 15:09:08.000000000 -0700
@@ -39,6 +39,7 @@
 #include <linux/stop_machine.h>
 #include <linux/sort.h>
 #include <linux/pfn.h>
+#include <linux/container.h>

 #include <asm/tlbflush.h>
 #include <asm/div64.h>
@@ -502,6 +503,7 @@ static void free_one_page(struct zone *z
  zone->pages_scanned = 0;
  __free_one_page(page, zone ,order);
  spin_unlock(&zone->lock);
+ container_init_page_ptr(page, NULL);
 }

 static void __free_pages_ok(struct page *page, unsigned int order)
@@ -798,6 +800,7 @@ static void fastcall free_hot_cold_page(
```

```
  arch_free_page(page, 0);

+ container_init_page_ptr(page, NULL);
  if (PageAnon(page))
   page->mapping = NULL;
  if (free_pages_check(page))
--- linux-2.6.18-rc6-mm2.org/mm/rmap.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/mm/rmap.c 2006-09-14 15:09:08.000000000 -0700
@@ -53,6 +53,7 @@
 #include <linux/rmap.h>
 #include <linux/rcupdate.h>
 #include <linux/module.h>
+#include <linux/container.h>

 #include <asm/tlbflush.h>

@@ -521,6 +522,7 @@ static void __page_set_anon_rmap(struct
    * interrupts because it is not modified via interrupt.
    */
   __inc_zone_page_state(page, NR_ANON_PAGES);
+ container_inc_page_count(page);
 }

 /**
@@ -563,8 +565,10 @@ void page_add_new_anon_rmap(struct page
  */
 void page_add_file_rmap(struct page *page)
 {
- if (atomic_inc_and_test(&page->_mapcount))
+ if (atomic_inc_and_test(&page->_mapcount)) {
    __inc_zone_page_state(page, NR_FILE_MAPPED);
+  container_inc_page_count(page);
+ }
 }

 /**
@@ -598,6 +602,8 @@ void page_remove_rmap(struct page *page)
     set_page_dirty(page);
    __dec_zone_page_state(page,
      PageAnon(page) ? NR_ANON_PAGES : NR_FILE_MAPPED);
+  container_dec_page_count(page);
+
   }
 }

--- linux-2.6.18-rc6-mm2.org/mm/swap.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/mm/swap.c 2006-09-14 15:09:08.000000000 -0700
```

```
@@ -196,6 +196,7 @@ void fastcall lru_cache_add_active(struc
  struct pagevec *pvec = &get_cpu_var(lru_add_active_pvecs);

  page_cache_get(page);
+ container_init_page_ptr(page, current);
  if (!pagevec_add(pvec, page))
   __pagevec_lru_add_active(pvec);
  put_cpu_var(lru_add_active_pvecs);
--- linux-2.6.18-rc6-mm2.org/mm/vmscan.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/mm/vmscan.c 2006-09-14 15:09:08.000000000 -0700
@@ -818,6 +818,7 @@ force_reclaim_mapped:
   SetPageLRU(page);
   VM_BUG_ON(!PageActive(page));
   ClearPageActive(page);
+  container_dec_activepage_count(page);

   list_move(&page->lru, &zone->inactive_list);
   pgmoved++;
--- linux-2.6.18-rc6-mm2.org/kernel/exit.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/kernel/exit.c 2006-09-18 18:18:22.000000000 -0700
@@ -41,6 +41,7 @@
 #include <linux/audit.h> /* for audit_free() */
 #include <linux/resource.h>
 #include <linux/blkdev.h>
+#include <linux/container.h>

 #include <asm/uaccess.h>
 #include <asm/unistd.h>
@@ -171,6 +172,7 @@ repeat:

  sched_exit(p);
  write_unlock_irq(&tasklist_lock);
+ container_remove_task(p, NULL);
  proc_flush_task(p);
  release_thread(p);
  call_rcu(&p->rcu, delayed_put_task_struct);
--- linux-2.6.18-rc6-mm2.org/kernel/fork.c 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/kernel/fork.c 2006-09-19 10:36:20.000000000 -0700
@@ -47,6 +47,7 @@
 #include <linux/cn_proc.h>
 #include <linux/delayacct.h>
 #include <linux/taskstats_kern.h>
+#include <linux/container.h>
 #include <linux/random.h>

 #include <asm/pgtable.h>
@@ -175,6 +176,13 @@ static struct task_struct *dup_task_stru
  }
```

```
  *tsk = *orig;
+
+ container_init_task_ptr(tsk);
+ if (container_add_task(tsk, orig, NULL) == -ENOSPC) {
+  free_task_struct(tsk);
+  free_thread_info(ti);
+  return NULL;
+ }
  tsk->thread_info = ti;
  setup_thread_stack(tsk, orig);

@@ -1295,6 +1303,7 @@ bad_fork_cleanup_count:
  atomic_dec(&p->user->processes);
  free_uid(p->user);
 bad_fork_free:
+ container_remove_task(p, NULL);
  free_task(p);
 fork_out:
  return ERR_PTR(retval);
--- linux-2.6.18-rc6-mm2.org/kernel/Makefile 2006-09-14 15:28:33.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/kernel/Makefile 2006-09-14 15:09:08.000000000 -0700
@@ -52,6 +52,7 @@ obj-$(CONFIG_RELAY) += relay.o
 obj-$(CONFIG_UTS_NS) += utsname.o
 obj-$(CONFIG_TASK_DELAY_ACCT) += delayacct.o
 obj-$(CONFIG_TASKSTATS) += taskstats.o tsacct.o
+obj-$(CONFIG_CONTAINERS) += container_configfs.o

 ifneq ($(CONFIG_SCHED_NO_NO_OMIT_FRAME_POINTER),y)
 # According to Alan Modra <alan@linuxcare.com.au>, the -fno-omit-frame-pointer is
--- linux-2.6.18-rc6-mm2.org/fs/inode.c 2006-09-14 15:28:31.000000000 -0700
+++ linux-2.6.18-rc6-mm2.ctn/fs/inode.c 2006-09-14 15:09:08.000000000 -0700
@@ -22,6 +22,7 @@
 #include <linux/bootmem.h>
 #include <linux/inotify.h>
 #include <linux/mount.h>
+#include <linux/container.h>

 /*
  * This is needed for the following functions:
@@ -164,6 +165,7 @@ static struct inode *alloc_inode(struct
   }
   inode->i_private = 0;
   inode->i_mapping = mapping;
+  container_add_file(mapping, NULL);
  }
  return inode;
 }
```

```
@@ -172,6 +174,7 @@ void destroy_inode(struct inode *inode)
 {
  BUG_ON(inode_has_buffers(inode));
  security_inode_free(inode);
+ container_remove_file(inode->i_mapping);
  if (inode->i_sb->s_op->destroy_inode)
   inode->i_sb->s_op->destroy_inode(inode);
  else
```