Subject: [PATCH 6/8] Per container OOM killer
Posted by Pavel Emelianov on Mon, 04 Jun 2007 13:40:17 GMT
View Forum Message <> Reply to Message

When container is completely out of memory some tasks should die.
This is unfair to kill the current task, so a task with the largest
RSS is chosen and killed. The code re-uses current OOM killer
select_bad_process() for task selection.

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
--- ./include/linux/rss_container.h.rssoom 2007-06-04 12:06:27.000000000 +0400
+++ ./include/linux/rss_container.h 2007-06-04 12:21:11.000000000 +0400
@@ -73,6 +73,8 @@ void container_rss_add(struct page_conta
 void container_rss_del(struct page_container *);
 void container_rss_release(struct page_container *);

+void container_out_of_memory(struct rss_container *);
+
 void mm_init_container(struct mm_struct *mm, struct task_struct *tsk);
 void mm_free_container(struct mm_struct *mm);
 #else
--- ./mm/oom_kill.c.rssoom 2007-06-04 12:05:26.000000000 +0400
+++ ./mm/oom_kill.c 2007-06-04 12:21:11.000000000 +0400
@@ -24,6 +24,7 @@
 #include <linux/cpuset.h>
 #include <linux/module.h>
 #include <linux/notifier.h>
+#include <linux/rss_container.h>

 int sysctl_panic_on_oom;
 /* #define DEBUG */
@@ -47,7 +48,8 @@ int sysctl_panic_on_oom;
  *    of least surprise ... (be careful when you change it)
  */

-unsigned long badness(struct task_struct *p, unsigned long uptime)
+unsigned long badness(struct task_struct *p, unsigned long uptime,
+  struct rss_container *rss)
 {
  unsigned long points, cpu_time, run_time, s;
  struct mm_struct *mm;
@@ -60,6 +62,13 @@ unsigned long badness(struct task_struct
   return 0;
  }
```

```
+#ifdef CONFIG_RSS_CONTAINER
+ if (rss != NULL && mm->rss_container != rss) {
+  task_unlock(p);
+  return 0;
+ }
+#endif
+
  /*
   * The memory size of the process is the basis for the badness.
   */
@@ -204,7 +213,8 @@ static inline int constrained_alloc(stru
  *
  * (not docbooked, we don't want this one cluttering up the manual)
  */
-static struct task_struct *select_bad_process(unsigned long *ppoints)
+static struct task_struct *select_bad_process(unsigned long *ppoints,
+  struct rss_container *rss)
 {
  struct task_struct *g, *p;
  struct task_struct *chosen = NULL;
@@ -258,7 +268,7 @@ static struct task_struct *select_bad_pr
   if (p->oomkilladj == OOM_DISABLE)
    continue;

-  points = badness(p, uptime.tv_sec);
+  points = badness(p, uptime.tv_sec, rss);
   if (points > *ppoints || !chosen) {
    chosen = p;
    *ppoints = points;
@@ -444,7 +454,7 @@ retry:
    * Rambo mode: Shoot down a process and hope it solves whatever
    * issues we may have.
    */
-  p = select_bad_process(&points);
+  p = select_bad_process(&points, NULL);

   if (PTR_ERR(p) == -1UL)
    goto out;
@@ -473,3 +483,27 @@ out:
  if (!test_thread_flag(TIF_MEMDIE))
   schedule_timeout_uninterruptible(1);
 }
+
+#ifdef CONFIG_RSS_CONTAINER
+void container_out_of_memory(struct rss_container *rss)
+{
+ unsigned long points = 0;
+ struct task_struct *p;
```

```
+
+ container_lock();
+ read_lock(&tasklist_lock);
+retry:
+ p = select_bad_process(&points, rss);
+ if (PTR_ERR(p) == -1UL)
+  goto out;
+
+ if (!p)
+  p = current;
+
+ if (oom_kill_process(p, points, "Container out of memory"))
+  goto retry;
+out:
+ read_unlock(&tasklist_lock);
+ container_unlock();
+}
+#endif
```
