Subject: [-mm PATCH 7/7] Memory controller OOM handling
Posted by Balbir Singh on Wed, 04 Jul 2007 22:22:53 GMT
View Forum Message <> Reply to Message

Out of memory handling for containers over their limit. A task from the
container over limit is chosen using the existing OOM logic and killed.

TODO:
1. As discussed in the OLS BOF session, consider implementing a user
space policy for OOM handling.

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

 include/linux/memcontrol.h | 1 +
 mm/memcontrol.c | 1 +
 mm/oom_kill.c | 42 ++++++++++++++++++++++++++++++++++++++----
 3 files changed, 40 insertions(+), 4 deletions(-)

```
diff -puN include/linux/memcontrol.h~mem-control-out-of-memory include/linux/memcontrol.h
--- linux-2.6.22-rc6/include/linux/memcontrol.h~mem-control-out-of-memory 2007-07-04
15:05:34.000000000 -0700
+++ linux-2.6.22-rc6-balbir/include/linux/memcontrol.h 2007-07-04 15:05:34.000000000 -0700
@@ -33,6 +33,7 @@ extern unsigned long mem_container_isola
      int mode, struct zone *z,
      struct mem_container *mem_cont,
      int active);
+extern void mem_cont_out_of_memory(struct mem_container *mem);

 #else /* CONFIG_CONTAINER_MEM_CONT */
 static inline void mm_init_container(struct mm_struct *mm,
diff -puN mm/memcontrol.c~mem-control-out-of-memory mm/memcontrol.c
--- linux-2.6.22-rc6/mm/memcontrol.c~mem-control-out-of-memory 2007-07-04
15:05:34.000000000 -0700
+++ linux-2.6.22-rc6-balbir/mm/memcontrol.c 2007-07-04 15:05:34.000000000 -0700
@@ -240,6 +240,7 @@ int mem_container_charge(struct page *pa
   if (res_counter_check_under_limit(&mem->res))
    continue;

+  mem_cont_out_of_memory(mem);
   goto free_mp;
  }

diff -puN mm/oom_kill.c~mem-control-out-of-memory mm/oom_kill.c
--- linux-2.6.22-rc6/mm/oom_kill.c~mem-control-out-of-memory 2007-07-04 15:05:34.000000000
-0700
+++ linux-2.6.22-rc6-balbir/mm/oom_kill.c 2007-07-04 15:05:34.000000000 -0700
@@ -24,6 +24,7 @@
```

```
 #include <linux/cpuset.h>
 #include <linux/module.h>
 #include <linux/notifier.h>
+#include <linux/memcontrol.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 mem_container *mem)
 {
  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_CONTAINER_MEM_CONT
+ if (mem != NULL && mm->mem_container != mem) {
+  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 mem_container *mem)
 {
  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, mem);
   if (points > *ppoints || !chosen) {
```

```
    chosen = p;
    *ppoints = points;
@@ -372,6 +382,30 @@ static int oom_kill_process(struct task_
  return oom_kill_task(p);
 }

+#ifdef CONFIG_CONTAINER_MEM_CONT
+void mem_cont_out_of_memory(struct mem_container *mem)
+{
+ unsigned long points = 0;
+ struct task_struct *p;
+
+ container_lock();
+ rcu_read_lock();
+retry:
+ p = select_bad_process(&points, mem);
+ if (PTR_ERR(p) == -1UL)
+  goto out;
+
+ if (!p)
+  p = current;
+
+ if (oom_kill_process(p, points, "Memory container out of memory"))
+  goto retry;
+out:
+ rcu_read_unlock();
+ container_unlock();
+}
+#endif
+
 static BLOCKING_NOTIFIER_HEAD(oom_notify_list);

 int register_oom_notifier(struct notifier_block *nb)
@@ -444,7 +478,7 @@ 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;
_

--
 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