
Subject: [PATCH -mm 2/5] swapcgroup (v3): add a member to swap_info_struct
Posted by [Daisuke Nishimura](#) on Fri, 04 Jul 2008 06:18:51 GMT
[View Forum Message](#) <> [Reply to Message](#)

This patch add a member to swap_info_struct for cgroup.

This member, array of pointers to mem_cgroup, is used to remember to which cgroup each swap entries are charged.

The memory for this array of pointers is allocated on swapon, and freed on swapoff.

Change log

v2->v3

- Rebased on 2.6.26-rc5-mm3
- add helper functions and removed `#ifdef` from `sys_swapon()/sys_swapoff()`.
- add check on `mem_cgroup_subsys.disabled`

v1->v2

- Rebased on 2.6.26-rc2-mm1
- Implemented as a add-on to memory cgroup.

Signed-off-by: Daisuke Nishimura <nishimura@mxp.nes.nec.co.jp>

```
include/linux/memcontrol.h | 20 ++++++
include/linux/swap.h       |  3 +++
mm/memcontrol.c            | 36 ++++++
mm/swapfile.c              | 11 ++++++
4 files changed, 69 insertions(+), 1 deletions(-)
```

diff --git a/include/linux/memcontrol.h b/include/linux/memcontrol.h

index ee1b2fc..b6ff509 100644

--- a/include/linux/memcontrol.h

+++ b/include/linux/memcontrol.h

@@ -24,6 +24,7 @@ struct mem_cgroup;

struct page_cgroup;

struct page;

struct mm_struct;

+struct swap_info_struct;

#ifdef CONFIG_CGROUP_MEM_RES_CTLR

@@ -165,5 +166,22 @@ static inline long mem_cgroup_calc_reclaim(struct mem_cgroup *mem,
{

#endif /* CONFIG_CGROUP_MEM_CONT */

```

-#endif /* _LINUX_MEMCONTROL_H */
+#ifdef CONFIG_CGROUP_SWAP_RES_CTLR
+extern struct mem_cgroup **swap_info_clear_memcg(struct swap_info_struct *p);
+extern int swap_info_alloc_memcg(struct swap_info_struct *p,
+ unsigned long maxpages);
+#else
+static inline
+struct mem_cgroup **swap_info_clear_memcg(struct swap_info_struct *p)
+{
+ return NULL;
+}

+static inline
+int swap_info_alloc_memcg(struct swap_info_struct *p, unsigned long maxpages)
+{
+ return 0;
+}
+#endif
+
+#endif /* _LINUX_MEMCONTROL_H */
diff --git a/include/linux/swap.h b/include/linux/swap.h
index a3af95b..6e1b03d 100644
--- a/include/linux/swap.h
+++ b/include/linux/swap.h
@@ -142,6 +142,9 @@ struct swap_info_struct {
     struct swap_extent *curr_swap_extent;
     unsigned old_block_size;
     unsigned short * swap_map;
+#ifdef CONFIG_CGROUP_SWAP_RES_CTLR
+ struct mem_cgroup **memcg;
+#endif
     unsigned int lowest_bit;
     unsigned int highest_bit;
     unsigned int cluster_next;
diff --git a/mm/memcontrol.c b/mm/memcontrol.c
index ddc842b..81bb7fa 100644
--- a/mm/memcontrol.c
+++ b/mm/memcontrol.c
@@ -1209,3 +1209,39 @@ struct cgroup_subsys mem_cgroup_subsys = {
     .attach = mem_cgroup_move_task,
     .early_init = 0,
 };
+
+#ifdef CONFIG_CGROUP_SWAP_RES_CTLR
+/* called with swap_lock held */
+struct mem_cgroup **swap_info_clear_memcg(struct swap_info_struct *p)
+{
+ struct mem_cgroup **mem;

```

```

+
+ /* just clear p->memcg, without checking mem_cgroup_subsys.disabled */
+ mem = p->memcg;
+ p->memcg = NULL;
+
+ return mem;
+}
+
+/* called without swap_lock held */
+int swap_info_alloc_memcg(struct swap_info_struct *p, unsigned long maxpages)
+{
+ int ret = 0;
+
+ if (mem_cgroup_subsys.disabled)
+ goto out;
+
+ p->memcg = vmalloc(maxpages * sizeof(struct mem_cgroup *));
+ if (!p->memcg) {
+ /* make swapon fail */
+ printk(KERN_ERR "Unable to allocate memory for memcg\n");
+ ret = -ENOMEM;
+ goto out;
+ }
+ memset(p->memcg, 0, maxpages * sizeof(struct mem_cgroup *));
+
+out:
+ return ret;
+}
+
+
diff --git a/mm/swapfile.c b/mm/swapfile.c
index bf7d13d..312c573 100644
--- a/mm/swapfile.c
+++ b/mm/swapfile.c
@@ -1228,6 +1228,7 @@ asmlinkage long sys_swapoff(const char __user * specialfile)
    unsigned short *swap_map;
    struct file *swap_file, *victim;
    struct address_space *mapping;
+ struct mem_cgroup **memcg = NULL;
    struct inode *inode;
    char * pathname;
    int i, type, prev;
@@ -1328,10 +1329,12 @@ asmlinkage long sys_swapoff(const char __user * specialfile)
    p->max = 0;
    swap_map = p->swap_map;
    p->swap_map = NULL;
+ memcg = swap_info_clear_memcg(p);
    p->flags = 0;

```

```

spin_unlock(&swap_lock);
mutex_unlock(&swapon_mutex);
vfree(swap_map);
+ vfree(memcg);
inode = mapping->host;
if (S_ISBLK(inode->i_mode)) {
    struct block_device *bdev = I_BDEV(inode);
@@ -1475,6 +1478,7 @@ asmlinkage long sys_swapon(const char __user * specialfile, int
swap_flags)
    unsigned long maxpages = 1;
    int swapfilesize;
    unsigned short *swap_map;
+ struct mem_cgroup **memcg = NULL;
    struct page *page = NULL;
    struct inode *inode = NULL;
    int did_down = 0;
@@ -1498,6 +1502,7 @@ asmlinkage long sys_swapon(const char __user * specialfile, int
swap_flags)
    p->swap_file = NULL;
    p->old_block_size = 0;
    p->swap_map = NULL;
+ swap_info_clear_memcg(p);
    p->lowest_bit = 0;
    p->highest_bit = 0;
    p->cluster_nr = 0;
@@ -1670,6 +1675,10 @@ asmlinkage long sys_swapon(const char __user * specialfile, int
swap_flags)
    1 /* header page */;
    if (error)
        goto bad_swap;
+
+ error = swap_info_alloc_memcg(p, maxpages);
+ if (error)
+     goto bad_swap;
+ }

    if (nr_good_pages) {
@@ -1729,11 +1738,13 @@ bad_swap_2:
    swap_map = p->swap_map;
    p->swap_file = NULL;
    p->swap_map = NULL;
+ memcg = swap_info_clear_memcg(p);
    p->flags = 0;
    if (!(swap_flags & SWAP_FLAG_PREFER))
        ++least_priority;
    spin_unlock(&swap_lock);
    vfree(swap_map);
+ vfree(memcg);

```

```
if (swap_file)
    filp_close(swap_file, NULL);
out:
```

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