Subject: Re: [PATCH 06/11] memcg: kmem controller infrastructure
Posted by akpm on Tue, 26 Jun 2012 19:20:50 GMT
View Forum Message <> Reply to Message

On Tue, 26 Jun 2012 22:14:51 +0400
Glauber Costa <glommer@parallels.com> wrote:

> On 06/26/2012 10:01 PM, Andrew Morton wrote:
> > On Tue, 26 Jun 2012 19:01:15 +0400 Glauber Costa <glommer@parallels.com> wrote:
> >
> >> On 06/26/2012 03:17 AM, Andrew Morton wrote:
> >>> + memcg_uncharge_kmem(memcg, size);
> >>> + mem_cgroup_put(memcg);
> >>> +}
> >>> +EXPORT_SYMBOL(__mem_cgroup_free_kmem_page);
> >>> #endif /* CONFIG_CGROUP_MEM_RES_CTLR_KMEM */
> >>>
> >>> #if defined(CONFIG_INET) &&
defined(CONFIG_CGROUP_MEM_RES_CTLR_KMEM)
> >>> @@ -5645,3 +5751,69 @@ static int __init enable_swap_account(char *s)
> >>> __setup("swapaccount=", enable_swap_account);
> >>>
> >>> #endif
> >>> +
> >>> +#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
> >> gargh. CONFIG_MEMCG_KMEM, please!
> >>
> >
> > Here too. I like it as much as you do.
> >
> > But that is consistent with the rest of the file, and I'd rather have
> > it this way.
> >
> > There's not much point in being consistent with something which is so
> > unpleasant. I'm on a little campaign to rename
> > CONFIG_CGROUP_MEM_RES_CTLR to CONFIG_MEMCG, only nobody has taken my
> > bait yet. Be first!
> >
>
> If you are okay with a preparation mechanical patch to convert the whole
> file, I can change mine too.
>
> But you'll be responsible for arguing with whoever stepping up opposing
> this =p
>

From: Andrew Morton <akpm@linux-foundation.org>
Subject: memcg: rename config variables

Sanity:

CONFIG_CGROUP_MEM_RES_CTLR -> CONFIG_MEMCG
CONFIG_CGROUP_MEM_RES_CTLR_SWAP -> CONFIG_MEMCG_SWAP
CONFIG_CGROUP_MEM_RES_CTLR_SWAP_ENABLED ->
CONFIG_MEMCG_SWAP_ENABLED
CONFIG_CGROUP_MEM_RES_CTLR_KMEM -> CONFIG_MEMCG_KMEM

Cc: Glauber Costa <glommer@parallels.com>
Cc: Michal Hocko <mhocko@suse.cz>
Cc: Johannes Weiner <hannes@cmpxchg.org>
Cc: KAMEZAWA Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>
Cc: Hugh Dickins <hughd@google.com>
Cc: Tejun Heo <tj@kernel.org>
Cc: Aneesh Kumar K.V <aneesh.kumar@linux.vnet.ibm.com>
Cc: David Rientjes <rientjes@google.com>
Cc: KOSAKI Motohiro <kosaki.motohiro@jp.fujitsu.com>
Signed-off-by: Andrew Morton <akpm@linux-foundation.org>
---

 include/linux/cgroup_subsys.h | 2 +-
 include/linux/memcontrol.h | 14 +++++++-------
 include/linux/mmzone.h | 8 ++++----
 include/linux/page_cgroup.h | 10 +++++-----
 include/linux/sched.h | 2 +-
 init/Kconfig | 14 +++++++-------
 kernel/fork.c | 2 +-
 mm/hwpoison-inject.c | 2 +-
 mm/memcontrol.c | 20 ++++++++++----------
 mm/memory-failure.c | 2 +-
 mm/mmzone.c | 2 +-
 mm/oom_kill.c | 2 +-
 mm/page_cgroup.c | 2 +-
 mm/vmscan.c | 4 ++--
 14 files changed, 43 insertions(+), 43 deletions(-)

diff -puN kernel/fork.c~a kernel/fork.c
--- a/kernel/fork.c~a
+++ a/kernel/fork.c
@@ -1302,7 +1302,7 @@ static struct task_struct *copy_process(
 #ifdef CONFIG_DEBUG_MUTEXES
  p->blocked_on = NULL; /* not blocked yet */
 #endif
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
  p->memcg_batch.do_batch = 0;
  p->memcg_batch.memcg = NULL;

```
 #endif
diff -puN mm/hwpoison-inject.c~a mm/hwpoison-inject.c
--- a/mm/hwpoison-inject.c~a
+++ a/mm/hwpoison-inject.c
@@ -123,7 +123,7 @@ static int pfn_inject_init(void)
  if (!dentry)
   goto fail;

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
  dentry = debugfs_create_u64("corrupt-filter-memcg", 0600,
         hwpoison_dir, &hwpoison_filter_memcg);
  if (!dentry)
diff -puN mm/memcontrol.c~a mm/memcontrol.c
--- a/mm/memcontrol.c~a
+++ a/mm/memcontrol.c
@@ -61,12 +61,12 @@ struct cgroup_subsys mem_cgroup_subsys _
 #define MEM_CGROUP_RECLAIM_RETRIES 5
 static struct mem_cgroup *root_mem_cgroup __read_mostly;

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 /* Turned on only when memory cgroup is enabled && really_do_swap_account = 1 */
 int do_swap_account __read_mostly;

 /* for remember boot option*/
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP_ENABLED
+#ifdef CONFIG_MEMCG_SWAP_ENABLED
 static int really_do_swap_account __initdata = 1;
 #else
 static int really_do_swap_account __initdata = 0;
@@ -407,7 +407,7 @@ static void mem_cgroup_get(struct mem_cg
 static void mem_cgroup_put(struct mem_cgroup *memcg);

 /* Writing them here to avoid exposing memcg's inner layout */
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
+#ifdef CONFIG_MEMCG_KMEM
 #include <net/sock.h>
 #include <net/ip.h>

@@ -466,9 +466,9 @@ struct cg_proto *tcp_proto_cgroup(struct
 }
 EXPORT_SYMBOL(tcp_proto_cgroup);
 #endif /* CONFIG_INET */
-#endif /* CONFIG_CGROUP_MEM_RES_CTLR_KMEM */
+#endif /* CONFIG_MEMCG_KMEM */

-#if defined(CONFIG_INET) && defined(CONFIG_CGROUP_MEM_RES_CTLR_KMEM)
```

```
+#if defined(CONFIG_INET) && defined(CONFIG_MEMCG_KMEM)
 static void disarm_sock_keys(struct mem_cgroup *memcg)
 {
  if (!memcg_proto_activated(&memcg->tcp_mem.cg_proto))
@@ -3085,7 +3085,7 @@ mem_cgroup_uncharge_swapcache(struct pag
 }
 #endif

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 /*
  * called from swap_entry_free(). remove record in swap_cgroup and
  * uncharge "memsw" account.
@@ -4518,7 +4518,7 @@ static int mem_cgroup_oom_control_write(
  return 0;
 }

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
+#ifdef CONFIG_MEMCG_KMEM
 static int memcg_init_kmem(struct mem_cgroup *memcg, struct cgroup_subsys *ss)
 {
  return mem_cgroup_sockets_init(memcg, ss);
@@ -4608,7 +4608,7 @@ static struct cftype mem_cgroup_files[]
   .read_seq_string = mem_control_numa_stat_show,
  },
 #endif
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
  {
   .name = "memsw.usage_in_bytes",
   .private = MEMFILE_PRIVATE(_MEMSWAP, RES_USAGE),
@@ -4795,7 +4795,7 @@ struct mem_cgroup *parent_mem_cgroup(str
 }
 EXPORT_SYMBOL(parent_mem_cgroup);

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 static void __init enable_swap_cgroup(void)
 {
  if (!mem_cgroup_disabled() && really_do_swap_account)
@@ -5526,7 +5526,7 @@ struct cgroup_subsys mem_cgroup_subsys =
  .__DEPRECATED_clear_css_refs = true,
 };

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 static int __init enable_swap_account(char *s)
 {
```

```
  /* consider enabled if no parameter or 1 is given */
diff -puN mm/memory-failure.c~a mm/memory-failure.c
--- a/mm/memory-failure.c~a
+++ a/mm/memory-failure.c
@@ -128,7 +128,7 @@ static int hwpoison_filter_flags(struct
  * can only guarantee that the page either belongs to the memcg tasks, or is
  * a freed page.
  */
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 u64 hwpoison_filter_memcg;
 EXPORT_SYMBOL_GPL(hwpoison_filter_memcg);
 static int hwpoison_filter_task(struct page *p)
diff -puN mm/mmzone.c~a mm/mmzone.c
--- a/mm/mmzone.c~a
+++ a/mm/mmzone.c
@@ -96,7 +96,7 @@ void lruvec_init(struct lruvec *lruvec,
  for_each_lru(lru)
   INIT_LIST_HEAD(&lruvec->lists[lru]);

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
  lruvec->zone = zone;
 #endif
 }
diff -puN mm/oom_kill.c~a mm/oom_kill.c
--- a/mm/oom_kill.c~a
+++ a/mm/oom_kill.c
@@ -541,7 +541,7 @@ static void check_panic_on_oom(enum oom_
   sysctl_panic_on_oom == 2 ? "compulsory" : "system-wide");
 }

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
 void mem_cgroup_out_of_memory(struct mem_cgroup *memcg, gfp_t gfp_mask,
         int order)
 {
diff -puN mm/page_cgroup.c~a mm/page_cgroup.c
--- a/mm/page_cgroup.c~a
+++ a/mm/page_cgroup.c
@@ -317,7 +317,7 @@ void __meminit pgdat_page_cgroup_init(st
 #endif


-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP

 static DEFINE_MUTEX(swap_cgroup_mutex);
```

```
 struct swap_cgroup_ctrl {
diff -puN mm/vmscan.c~a mm/vmscan.c
--- a/mm/vmscan.c~a
+++ a/mm/vmscan.c
@@ -133,7 +133,7 @@ long vm_total_pages; /* The total number
 static LIST_HEAD(shrinker_list);
 static DECLARE_RWSEM(shrinker_rwsem);

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
 static bool global_reclaim(struct scan_control *sc)
 {
  return !sc->target_mem_cgroup;
@@ -2152,7 +2152,7 @@ unsigned long try_to_free_pages(struct z
  return nr_reclaimed;
 }

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG

 unsigned long mem_cgroup_shrink_node_zone(struct mem_cgroup *memcg,
       gfp_t gfp_mask, bool noswap,
diff -puN init/Kconfig~a init/Kconfig
--- a/init/Kconfig~a
+++ a/init/Kconfig
@@ -686,7 +686,7 @@ config RESOURCE_COUNTERS
    This option enables controller independent resource accounting
    infrastructure that works with cgroups.

-config CGROUP_MEM_RES_CTLR
+config MEMCG
  bool "Memory Resource Controller for Control Groups"
  depends on RESOURCE_COUNTERS
  select MM_OWNER
@@ -709,9 +709,9 @@ config CGROUP_MEM_RES_CTLR
    This config option also selects MM_OWNER config option, which
    could in turn add some fork/exit overhead.

-config CGROUP_MEM_RES_CTLR_SWAP
+config MEMCG_SWAP
  bool "Memory Resource Controller Swap Extension"
- depends on CGROUP_MEM_RES_CTLR && SWAP
+ depends on MEMCG && SWAP
  help
    Add swap management feature to memory resource controller. When you
    enable this, you can limit mem+swap usage per cgroup. In other words,
@@ -726,9 +726,9 @@ config CGROUP_MEM_RES_CTLR_SWAP
    if boot option "swapaccount=0" is set, swap will not be accounted.
```

```
    Now, memory usage of swap_cgroup is 2 bytes per entry. If swap page
    size is 4096bytes, 512k per 1Gbytes of swap.
-config CGROUP_MEM_RES_CTLR_SWAP_ENABLED
+config MEMCG_SWAP_ENABLED
  bool "Memory Resource Controller Swap Extension enabled by default"
- depends on CGROUP_MEM_RES_CTLR_SWAP
+ depends on MEMCG_SWAP
  default y
  help
    Memory Resource Controller Swap Extension comes with its price in
@@ -739,9 +739,9 @@ config CGROUP_MEM_RES_CTLR_SWAP_ENABLED
    For those who want to have the feature enabled by default should
    select this option (if, for some reason, they need to disable it
    then swapaccount=0 does the trick).
-config CGROUP_MEM_RES_CTLR_KMEM
+config MEMCG_KMEM
  bool "Memory Resource Controller Kernel Memory accounting (EXPERIMENTAL)"
- depends on CGROUP_MEM_RES_CTLR && EXPERIMENTAL
+ depends on MEMCG && EXPERIMENTAL
  default n
  help
    The Kernel Memory extension for Memory Resource Controller can limit
diff -puN include/linux/cgroup_subsys.h~a include/linux/cgroup_subsys.h
--- a/include/linux/cgroup_subsys.h~a
+++ a/include/linux/cgroup_subsys.h
@@ -31,7 +31,7 @@ SUBSYS(cpuacct)

 /* */

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
 SUBSYS(mem_cgroup)
 #endif

diff -puN include/linux/memcontrol.h~a include/linux/memcontrol.h
--- a/include/linux/memcontrol.h~a
+++ a/include/linux/memcontrol.h
@@ -38,7 +38,7 @@ struct mem_cgroup_reclaim_cookie {
  unsigned int generation;
 };

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
 /*
  * All "charge" functions with gfp_mask should use GFP_KERNEL or
  * (gfp_mask & GFP_RECLAIM_MASK). In current implementatin, memcg doesn't
@@ -124,7 +124,7 @@ extern void mem_cgroup_print_oom_info(st
 extern void mem_cgroup_replace_page_cache(struct page *oldpage,
```

```
     struct page *newpage);

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 extern int do_swap_account;
 #endif

@@ -193,7 +193,7 @@ void mem_cgroup_split_huge_fixup(struct
 bool mem_cgroup_bad_page_check(struct page *page);
 void mem_cgroup_print_bad_page(struct page *page);
 #endif
-#else /* CONFIG_CGROUP_MEM_RES_CTLR */
+#else /* CONFIG_MEMCG */
 struct mem_cgroup;

 static inline int mem_cgroup_newpage_charge(struct page *page,
@@ -384,9 +384,9 @@ static inline void mem_cgroup_replace_pa
     struct page *newpage)
 {
 }
-#endif /* CONFIG_CGROUP_MEM_RES_CTLR */
+#endif /* CONFIG_MEMCG */

-#if !defined(CONFIG_CGROUP_MEM_RES_CTLR) || !defined(CONFIG_DEBUG_VM)
+#if !defined(CONFIG_MEMCG) || !defined(CONFIG_DEBUG_VM)
 static inline bool
 mem_cgroup_bad_page_check(struct page *page)
 {
@@ -406,7 +406,7 @@ enum {
 };

 struct sock;
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_KMEM
+#ifdef CONFIG_MEMCG_KMEM
 void sock_update_memcg(struct sock *sk);
 void sock_release_memcg(struct sock *sk);
 #else
@@ -416,6 +416,6 @@ static inline void sock_update_memcg(str
 static inline void sock_release_memcg(struct sock *sk)
 {
 }
-#endif /* CONFIG_CGROUP_MEM_RES_CTLR_KMEM */
+#endif /* CONFIG_MEMCG_KMEM */
 #endif /* _LINUX_MEMCONTROL_H */

diff -puN include/linux/mmzone.h~a include/linux/mmzone.h
--- a/include/linux/mmzone.h~a
+++ a/include/linux/mmzone.h
```

```
@@ -200,7 +200,7 @@ struct zone_reclaim_stat {
 struct lruvec {
  struct list_head lists[NR_LRU_LISTS];
  struct zone_reclaim_stat reclaim_stat;
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
  struct zone *zone;
 #endif
 };
@@ -670,7 +670,7 @@ typedef struct pglist_data {
  int nr_zones;
 #ifdef CONFIG_FLAT_NODE_MEM_MAP /* means !SPARSEMEM */
  struct page *node_mem_map;
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
  struct page_cgroup *node_page_cgroup;
 #endif
 #endif
@@ -735,7 +735,7 @@ extern void lruvec_init(struct lruvec *l

 static inline struct zone *lruvec_zone(struct lruvec *lruvec)
 {
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
  return lruvec->zone;
 #else
  return container_of(lruvec, struct zone, lruvec);
@@ -1051,7 +1051,7 @@ struct mem_section {

  /* See declaration of similar field in struct zone */
  unsigned long *pageblock_flags;
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
  /*
   * If !SPARSEMEM, pgdat doesn't have page_cgroup pointer. We use
   * section. (see memcontrol.h/page_cgroup.h about this.)
diff -puN include/linux/page_cgroup.h~a include/linux/page_cgroup.h
--- a/include/linux/page_cgroup.h~a
+++ a/include/linux/page_cgroup.h
@@ -12,7 +12,7 @@ enum {
 #ifndef __GENERATING_BOUNDS_H
 #include <generated/bounds.h>

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR
+#ifdef CONFIG_MEMCG
 #include <linux/bit_spinlock.h>

 /*
```

```
@@ -82,7 +82,7 @@ static inline void unlock_page_cgroup(st
  bit_spin_unlock(PCG_LOCK, &pc->flags);
 }

-#else /* CONFIG_CGROUP_MEM_RES_CTLR */
+#else /* CONFIG_MEMCG */
 struct page_cgroup;

 static inline void __meminit pgdat_page_cgroup_init(struct pglist_data *pgdat)
@@ -102,11 +102,11 @@ static inline void __init page_cgroup_in
 {
 }

-#endif /* CONFIG_CGROUP_MEM_RES_CTLR */
+#endif /* CONFIG_MEMCG */

 #include <linux/swap.h>

-#ifdef CONFIG_CGROUP_MEM_RES_CTLR_SWAP
+#ifdef CONFIG_MEMCG_SWAP
 extern unsigned short swap_cgroup_cmpxchg(swp_entry_t ent,
      unsigned short old, unsigned short new);
 extern unsigned short swap_cgroup_record(swp_entry_t ent, unsigned short id);
@@ -138,7 +138,7 @@ static inline void swap_cgroup_swapoff(i
  return;
 }

-#endif /* CONFIG_CGROUP_MEM_RES_CTLR_SWAP */
+#endif /* CONFIG_MEMCG_SWAP */

 #endif /* !__GENERATING_BOUNDS_H */

diff -puN include/linux/sched.h~a include/linux/sched.h
--- a/include/linux/sched.h~a
+++ a/include/linux/sched.h
@@ -1581,7 +1581,7 @@ struct task_struct {
  /* bitmask and counter of trace recursion */
  unsigned long trace_recursion;
 #endif /* CONFIG_TRACING */
-#ifdef CONFIG_CGROUP_MEM_RES_CTLR /* memcg uses this to do batch job */
+#ifdef CONFIG_MEMCG /* memcg uses this to do batch job */
  struct memcg_batch_info {
   int do_batch; /* incremented when batch uncharge started */
   struct mem_cgroup *memcg; /* target memcg of uncharge */
diff -puN include/linux/swap.h~a include/linux/swap.h
_
```