Subject: Re: [RFC 1/4] memcg: provide root figures from system totals
Posted by Michal Hocko on Mon, 01 Oct 2012 17:00:46 GMT
View Forum Message <> Reply to Message

On Tue 25-09-12 12:52:50, Glauber Costa wrote:
> For the root memcg, there is no need to rely on the res_counters.

This is true only if there are no children groups but once there is at
least one we have to move global statistics into root res_counter and
start using it since then. This is a tricky part because it has to be
done atomically so that we do not miss anything.

> The sum of all mem cgroups plus the tasks in root itself, is necessarily
> the amount of memory used for the whole system. Since those figures are
> already kept somewhere anyway, we can just return them here, without too
> much hassle.
>
> The limit can't be set for the root cgroup, so it is left at 0.

You meant RESOURCE_MAX, didn't you?

> Same is
> true for failcnt, because its actual meaning is how many times we failed
> allocations due to the limit being hit. We will fail allocations in the
> root cgroup, but the limit will never the reason.
>
> TODO includes figuring out what to do with the soft limit and max_usage.
> Comments and suggestions appreciated.
>
> Signed-off-by: Glauber Costa <glommer@parallels.com>
> CC: Michal Hocko <mhocko@suse.cz>
> CC: Kamezawa Hiroyuki <kamezawa.hiroyu@jp.fujitsu.com>
> CC: Johannes Weiner <hannes@cmpxchg.org>
> CC: Mel Gorman <mgorman@suse.de>
> CC: Andrew Morton <akpm@linux-foundation.org>
> ---
> mm/memcontrol.c | 45 +++
> 1 file changed, 45 insertions(+)
>
> diff --git a/mm/memcontrol.c b/mm/memcontrol.c
> index 82c5b8f..bac398b 100644
> --- a/mm/memcontrol.c
> +++ b/mm/memcontrol.c
> @@ -4506,6 +4506,45 @@ static inline u64 mem_cgroup_usage(struct mem_cgroup *memcg,
bool swap)
>   return val << PAGE_SHIFT;
> }
>

```
> +static u64 mem_cgroup_read_root(struct mem_cgroup *memcg, enum res_type type, int
name)
> +{
> + struct sysinfo i;
> +       unsigned long pages[NR_LRU_LISTS];
> +       int lru;
> +
> + si_meminfo(&i);
> + si_swapinfo(&i);
> +
> + if (name == RES_LIMIT)
> +  return RESOURCE_MAX;
> + if (name == RES_SOFT_LIMIT)
> +  return 0;
> + if (name == RES_FAILCNT)
> +  return 0;
> + if (name == RES_MAX_USAGE)
> +  return 0;
> +
> + if (WARN_ON_ONCE(name != RES_USAGE))
> +  return 0;
> +
> + for (lru = LRU_BASE; lru < NR_LRU_LISTS; lru++)
> +  pages[lru] = global_page_state(NR_LRU_BASE + lru);
> +
> + switch (type) {
> + case _MEM:
> +              return pages[LRU_ACTIVE_ANON]   + pages[LRU_ACTIVE_FILE] +
> +   pages[LRU_INACTIVE_ANON] + pages[LRU_INACTIVE_FILE];
> + case _MEMSWAP:
> +              return pages[LRU_ACTIVE_ANON]   + pages[LRU_ACTIVE_FILE] +
> +   pages[LRU_INACTIVE_ANON] + pages[LRU_INACTIVE_FILE] +
> +   i.totalswap - i.freeswap;
> + case _KMEM:
> +  return 0;
> + default:
> +  BUG();
> + };
> +}
> +
>  static ssize_t mem_cgroup_read(struct cgroup *cont, struct cftype *cft,
>           struct file *file, char __user *buf,
>           size_t nbytes, loff_t *ppos)
> @@ -4522,6 +4561,11 @@ static ssize_t mem_cgroup_read(struct cgroup *cont, struct cftype
*cft,
>   if (!do_swap_account && type == _MEMSWAP)
>    return -EOPNOTSUPP;
>
```

> + if (mem_cgroup_is_root(memcg)) {
> + val = mem_cgroup_read_root(memcg, type, name);
> + goto root_bypass;
> + }
> +
>   switch (type) {
>   case _MEM:
>   if (name == RES_USAGE)
> @@ -4542,6 +4586,7 @@ static ssize_t mem_cgroup_read(struct cgroup *cont, struct cftype
*cft,
>   BUG();
>   }
>
> +root_bypass:
>   len = scnprintf(str, sizeof(str), "%llu\n", (unsigned long long)val);
>   return simple_read_from_buffer(buf, nbytes, ppos, str, len);
> }
> --
> 1.7.11.4
>
> --
> To unsubscribe from this list: send the line "unsubscribe cgroups" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

--
Michal Hocko
SUSE Labs