Subject: Re: [Fwd: [PATCH -RSS 2/2] Fix limit check after reclaim]
Posted by Balbir Singh on Tue, 05 Jun 2007 06:58:54 GMT
View Forum Message <> Reply to Message

Pavel Emelianov wrote:
>> +static inline bool res_counter_check_under_limit(struct res_counter *cnt)
>> +{
>> + bool ret;
>> + unsigned long flags;
>> +
>> + spin_lock_irqsave(&cnt->lock, flags);
>> + ret = res_counter_limit_check_locked(cnt);
>
> We don't have to take the lock for such a check.
>

This check without the lock could be racy and return incorrect
results -- leading to OOM.

--
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

Subject: Re: [Fwd: [PATCH -RSS 2/2] Fix limit check after reclaim]
Posted by xemul on Tue, 05 Jun 2007 07:02:00 GMT
View Forum Message <> Reply to Message

Balbir Singh wrote:
>
> -------- Original Message --------
> Subject: [PATCH -RSS 2/2] Fix limit check after reclaim
> Date: Mon, 04 Jun 2007 21:03:04 +0530
> From: Balbir Singh <balbir@linux.vnet.ibm.com>
> To: Andrew Morton <akpm@osdl.org>
> CC: Linux Containers <containers@lists.osdl.org>, Balbir Singh
<balbir@linux.vnet.ibm.com>, Vaidyanathan Srinivasan <svaidy@linux.vnet.ibm.com>,
Linux Kernel Mailing List <linux-kernel@vger.kernel.org>
> References: <20070604153244.31748.50043.sendpatchset@balbir-laptop>
>
>

>
> This patch modifies the reclaim behaviour such that before calling the
> container out of memory routine, it checks if as a result of the reclaim
> (even though pages might not be fully reclaimed), the resident set size
> of the container decreased before declaring the container as out of memory
>
> Signed-off-by: Balbir Singh <balbir@linux.vnet.ibm.com>
> ---
>
> include/linux/res_counter.h | 23 ++++++++++++++++++++++
> mm/rss_container.c | 11 ++++++++++
> 2 files changed, 34 insertions(+)
>
> diff -puN mm/rss_container.c~rss-fix-limit-check-after-reclaim mm/rss_container.c
> --- linux-2.6.22-rc2-mm1/mm/rss_container.c~rss-fix-limit-check-after-reclaim 2007-06-04
20:13:40.000000000 +0530
> +++ linux-2.6.22-rc2-mm1-balbir/mm/rss_container.c 2007-06-04 20:13:40.000000000 +0530
> @@ -114,6 +114,17 @@ int container_rss_prepare(struct page *p
>   continue;
>   }
>
> + /*
> +   * try_to_free_pages() might not give us a full picture
> +   * of reclaim. Some pages are reclaimed and might be moved
> +   * to swap cache or just unmapped from the container.
> +   * Check the limit again to see if the reclaim reduced the
> +   * current usage of the container before calling the
> +   * container OOM routine
> +   */
> + if (res_counter_check_under_limit(&rss->res))
> +   continue;
> +
>   container_out_of_memory(rss);
>   if (test_thread_flag(TIF_MEMDIE))
>   goto out_charge;
> diff -puN include/linux/res_counter.h~rss-fix-limit-check-after-reclaim include/linux/res_counter.h
> --- linux-2.6.22-rc2-mm1/include/linux/res_counter.h~rss-fix-limit-check-after-reclaim 2007-06-04
20:13:40.000000000 +0530
> +++ linux-2.6.22-rc2-mm1-balbir/include/linux/res_counter.h 2007-06-04 20:15:46.000000000
+0530
> @@ -99,4 +99,27 @@ int res_counter_charge(struct res_counte
> void res_counter_uncharge_locked(struct res_counter *cnt, unsigned long val);
> void res_counter_uncharge(struct res_counter *cnt, unsigned long val);
>
> +static inline bool res_counter_limit_check_locked(struct res_counter *cnt)
> +{
> + if (cnt->usage < cnt->limit)
> + return true;

> +
> + return false;
> +}
> +
> +/*
> + * Helper function to detect if the container is within it's limit or
> + * not. It's currently called from container_rss_prepare()
> + */
> +static inline bool res_counter_check_under_limit(struct res_counter *cnt)
> +{
> + bool ret;
> + unsigned long flags;
> +
> + spin_lock_irqsave(&cnt->lock, flags);
> + ret = res_counter_limit_check_locked(cnt);

We don't have to take the lock for such a check.

> + spin_unlock_irqrestore(&cnt->lock, flags);
> + return ret;
> +}
> +
> #endif
> diff -puN mm/vmscan.c~rss-fix-limit-check-after-reclaim mm/vmscan.c
> _
>

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

---

Subject: Re: Re: [Fwd: [PATCH -RSS 2/2] Fix limit check after reclaim]
Posted by Balbir Singh on Tue, 05 Jun 2007 07:10:10 GMT
View Forum Message <> Reply to Message

Pavel Emelianov wrote:
> Balbir Singh wrote:
>> Pavel Emelianov wrote:
>>>> +static inline bool res_counter_check_under_limit(struct res_counter *cnt)
>>>> +{
>>>> + bool ret;
>>>> + unsigned long flags;
>>>> +
>>>> + spin_lock_irqsave(&cnt->lock, flags);
>>>> + ret = res_counter_limit_check_locked(cnt);
>>> We don't have to take the lock for such a check.

>>>
>>
>> This check without the lock could be racy and return incorrect
>> results -- leading to OOM.
>
> Maybe. Nevertheless, if we do not trust the return value of
> try_to_free_pages() then the code should probably look like
>
> while (1) {
> if (res_counter_charge() == 0)
>   break;
>
> did_progress = try_to_free_pages();
> if (res_counter_charge() == 0)
>   break;
>
> if (!did_progress)
>   out_of_memory();
> }
>

Yes, this looks better.

> But in any case we must know for sure was at least one page
> freed or not...
>
> Thanks,
> Pavel

--
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

---

Subject: Re: Re: [Fwd: [PATCH -RSS 2/2] Fix limit check after reclaim]
Posted by xemul on Tue, 05 Jun 2007 07:13:16 GMT
View Forum Message <> Reply to Message

Balbir Singh wrote:
> Pavel Emelianov wrote:
>>> +static inline bool res_counter_check_under_limit(struct res_counter *cnt)

>>> +{
>>> + bool ret;
>>> + unsigned long flags;
>>> +
>>> + spin_lock_irqsave(&cnt->lock, flags);
>>> + ret = res_counter_limit_check_locked(cnt);
>> We don't have to take the lock for such a check.
>>
>
> This check without the lock could be racy and return incorrect
> results -- leading to OOM.

Maybe. Nevertheless, if we do not trust the return value of
try_to_free_pages() then the code should probably look like

```
while (1) {
 if (res_counter_charge() == 0)
  break;

 did_progress = try_to_free_pages();
 if (res_counter_charge() == 0)
  break;

 if (!did_progress)
  out_of_memory();
}
```

But in any case we must know for sure was at least one page
freed or not...

Thanks,
Pavel

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