Subject: [RFC] cgroup basic comounting
Posted by Glauber Costa on Fri, 16 Dec 2011 12:29:09 GMT
View Forum Message <> Reply to Message

Turns out that most of the infrastructure we need to put two controllers in the same hierarchy is by far already into place. All we need to do is not failing when we specify two of them.

With this, we can effectively guarantee that by comounting cpu and cpuacct, we'll have the same set of tasks, therefore allowing us to use cpu cgroup data to fill in the usage fields in cpuacct.

I decided not to stabilish any dependency between cgroups as Li previously did: cgroups may or may not be comounted, and any of them can be combined (I don't see a reason to prevent any combination).

After testing and some trials, I could verify that the current mount behavior plays well under the plans, so I didn't change it. That is:

 * If subsystems A and B aren't mounted, we can comount them.
 * If subsystem A is mounted, but B is not:
   * we can comount them if A has no children,
   * we fail otherwise
 * If subsystems A and B are comounted at a location, we can't
   mount any of them separately at another point. We do can mount
   them together.
 * If subsystems A and B are comounted at a location,
   * we can comount a third subsystem C, if they have no children
   * we fail otherwise

Paul,

Please let me know if this is tuned with the idea you had in mind.
If this is okay, I patch that extracts usage from cpu cgroup data
in case of comount would follow.

Signed-off-by: Glauber Costa <glommer@parallels.com>
CC: Paul Turner <pjt@google.com>
CC: Li Zefan <lizf@cn.fujitsu.com>
---
 kernel/cgroup.c | 4 ++--
 1 files changed, 2 insertions(+), 2 deletions(-)

diff --git a/kernel/cgroup.c b/kernel/cgroup.c
index 1fd7867..e894a4f 100644
--- a/kernel/cgroup.c
+++ b/kernel/cgroup.c
@@ -1211,9 +1211,9 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts

```
*opts)
    set_bit(i, &opts->subsys_bits);
    one_ss = true;

-   break;
+   continue;
   }
-  if (i == CGROUP_SUBSYS_COUNT)
+  if (opts->subsys_bits == 0)
    return -ENOENT;
  }

--
1.7.6.4
```

Subject: Re: [RFC] cgroup basic comounting
Posted by Paul Menage on Fri, 16 Dec 2011 16:35:34 GMT
View Forum Message <> Reply to Message

On Fri, Dec 16, 2011 at 4:29 AM, Glauber Costa <glommer@parallels.com> wrote
> diff --git a/kernel/cgroup.c b/kernel/cgroup.c
> index 1fd7867..e894a4f 100644
> --- a/kernel/cgroup.c
> +++ b/kernel/cgroup.c
> @@ -1211,9 +1211,9 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts
*opts)
>   set_bit(i, &opts->subsys_bits);
>   one_ss = true;
>
> -   break;
> +   continue;
>   }
> -   if (i == CGROUP_SUBSYS_COUNT)
> +   if (opts->subsys_bits == 0)
>   return -ENOENT;

This is broken - it will silently ignore unknown/misspelled subsystems
that are specified after a valid subsystem. Replacing the break with a
continue is harmless but doesn't make sense - if the token already
matched a subsystem name then it won't match any other subsystem.

What change are you actually trying to effect with this patch?

Paul

Subject: Re: [RFC] cgroup basic comounting
Posted by Li Zefan on Mon, 19 Dec 2011 07:58:21 GMT
View Forum Message <> Reply to Message

Glauber Costa wrote:
> Turns out that most of the infrastructure we need to put two controllers in the
> same hierarchy is by far already into place. All we need to do is not failing
> when we specify two of them.
>

You don't need to change anything to mount with 2 cgroup subsystems:

 # mount -t cgroup -o cpu,cpuacct xxx /mnt

But you may want to revise and make use of the subsys->bind() callback, which
is called at mount/remount/umount when we attach/remove a controller to/from
a hierarchy. It's the place you can check if two controllers are going to
be comounted/seperated.

> With this, we can effectively guarantee that by comounting cpu and cpuacct,
> we'll have the same set of tasks, therefore allowing us to use cpu cgroup data
> to fill in the usage fields in cpuacct.
>
> I decided not to stabilish any dependency between cgroups as Li previously did:
> cgroups may or may not be comounted, and any of them can be combined (I don't
> see a reason to prevent any combination).
>
> After testing and some trials, I could verify that the current mount behavior
> plays well under the plans, so I didn't change it. That is:
>
> * If subsystems A and B aren't mounted, we can comount them.
> * If subsystem A is mounted, but B is not:
>   * we can comount them if A has no children,
>   * we fail otherwise
> * If subsystems A and B are comounted at a location, we can't
>   mount any of them separately at another point. We do can mount
>   them together.
> * If subsystems A and B are comounted at a location,
>   * we can comount a third subsystem C, if they have no children
>   * we fail otherwise
>
> Paul,
>
> Please let me know if this is tuned with the idea you had in mind.
> If this is okay, I patch that extracts usage from cpu cgroup data
> in case of comount would follow.
>
> Signed-off-by: Glauber Costa <glommer@parallels.com>
> CC: Paul Turner <pjt@google.com>

> CC: Li Zefan <lizf@cn.fujitsu.com>
> ---
> kernel/cgroup.c | 4 ++--
> 1 files changed, 2 insertions(+), 2 deletions(-)
>
> diff --git a/kernel/cgroup.c b/kernel/cgroup.c
> index 1fd7867..e894a4f 100644
> --- a/kernel/cgroup.c
> +++ b/kernel/cgroup.c
> @@ -1211,9 +1211,9 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts
*opts)
>   set_bit(i, &opts->subsys_bits);
>   one_ss = true;
>
> -   break;
> +   continue;
>   }
> - if (i == CGROUP_SUBSYS_COUNT)
> + if (opts->subsys_bits == 0)
>   return -ENOENT;
>   }
>

Subject: Re: [RFC] cgroup basic comounting
Posted by Glauber Costa on Mon, 19 Dec 2011 08:00:19 GMT
View Forum Message <> Reply to Message

On 12/19/2011 11:58 AM, Li Zefan wrote:
> Glauber Costa wrote:
>> Turns out that most of the infrastructure we need to put two controllers in the
>> same hierarchy is by far already into place. All we need to do is not failing
>> when we specify two of them.
>>
>
> You don't need to change anything to mount with 2 cgroup subsystems:
>
> # mount -t cgroup -o cpu,cpuacct xxx /mnt
>
> But you may want to revise and make use of the subsys->bind() callback, which
> is called at mount/remount/umount when we attach/remove a controller to/from
> a hierarchy. It's the place you can check if two controllers are going to
> be comounted/seperated.
>
>> With this, we can effectively guarantee that by comounting cpu and cpuacct,
>> we'll have the same set of tasks, therefore allowing us to use cpu cgroup data
>> to fill in the usage fields in cpuacct.

Yeah, that patch was bogus, sorry for the noise.

What I should really have posted is the test code, but I guess I'll go
over that one as well one more time, and then post it.

Thanks
>> I decided not to stabilish any dependency between cgroups as Li previously did:
>> cgroups may or may not be comounted, and any of them can be combined (I don't
>> see a reason to prevent any combination).
>>
>> After testing and some trials, I could verify that the current mount behavior
>> plays well under the plans, so I didn't change it. That is:
>>
>> * If subsystems A and B aren't mounted, we can comount them.
>> * If subsystem A is mounted, but B is not:
>> * we can comount them if A has no children,
>> * we fail otherwise
>> * If subsystems A and B are comounted at a location, we can't
>> mount any of them separately at another point. We do can mount
>> them together.
>> * If subsystems A and B are comounted at a location,
>> * we can comount a third subsystem C, if they have no children
>> * we fail otherwise
>>
>> Paul,
>>
>> Please let me know if this is tuned with the idea you had in mind.
>> If this is okay, I patch that extracts usage from cpu cgroup data
>> in case of comount would follow.
>>
>> Signed-off-by: Glauber Costa<glommer@parallels.com>
>> CC: Paul Turner<pjt@google.com>
>> CC: Li Zefan<lizf@cn.fujitsu.com>
>> ---
>> kernel/cgroup.c | 4 ++--
>> 1 files changed, 2 insertions(+), 2 deletions(-)
>>
>> diff --git a/kernel/cgroup.c b/kernel/cgroup.c
>> index 1fd7867..e894a4f 100644
>> --- a/kernel/cgroup.c
>> +++ b/kernel/cgroup.c
>> @@ -1211,9 +1211,9 @@ static int parse_cgroupfs_options(char *data, struct
cgroup_sb_opts *opts)
>> set_bit(i,&opts->subsys_bits);
>> one_ss = true;
>>
>> - break;
>> + continue;

>>   }
>> - if (i == CGROUP_SUBSYS_COUNT)
>> + if (opts->subsys_bits == 0)
>>   return -ENOENT;
>>   }
>>
> --
> 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