Subject: How to draw values for /proc/stat
Posted by Glauber Costa on Mon, 05 Dec 2011 09:32:33 GMT
View Forum Message <> Reply to Message

Hi,

Specially Peter and Paul, but all the others:

As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer to that, there is a question - one I've asked before but without that much of an audience - of whether /proc files read from process living on cgroups should display global or per-cgroup resources.

In the past, I was arguing for a knob to control that, but I recently started to believe that a knob here will only overcomplicate matters: if you live in a cgroup, you should display only the resources you can possibly use. Global is for whoever is in the main cgroup.

Now, it comes two questions:
1) Do you agree with that, for files like /proc/stat ? I think the most important part is to be consistent inside the system, regardless of what is done

2) Will cpuacct stay? I think if it does, that becomes almost mandatory (at least the bind mount idea is pretty much over here), because drawing value for /proc/stat becomes quite complex.
The cpuacct cgroup can provide user, sys, etc values. But we also have:

* nr_context_switches,
* jiffies since boot,
* total_forks,
* nr_running,
* nr_iowait,

Now I doubt any of us want to see /proc/stat extended to accommodate things like nr_context_switches, or even worse, nr_running. The way I see it, there are two options here:

a) moving everything to cpu cgroup so we keep all values being drawn from the same place
b) Collect that info from multiple places in a transparent way. ctx, nr_running and nr_iowait will probably come from cpu. jiffies can come from wherever, and maybe we can even draw total_forks from Frederic's and avoid counting it twice.

Subject: Re: How to draw values for /proc/stat

On Mon, 5 Dec 2011 07:32:33 -0200
Glauber Costa <glommer@parallels.com> wrote:

> Hi,
>
> Specially Peter and Paul, but all the others:
>
> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
> to that, there is a question - one I've asked before but without that
> much of an audience - of whether /proc files read from process living on
> cgroups should display global or per-cgroup resources.
>
> In the past, I was arguing for a knob to control that, but I recently
> started to believe that a knob here will only overcomplicate matters:
> if you live in a cgroup, you should display only the resources you can
> possibly use. Global is for whoever is in the main cgroup.
>

Hm. I have a suggestion and a concern.

(A suggestion)
  How about having a mount option for procfs ?
  For example,
 mount -t proc .... -o cgroup_virtualized
  Then, /proc/stat etc shows per-cgroup information.

(A concern)
  /proc/stat will be a mixture of virtualized values and not-virtualized values.
  1. Don't users need to know whether each value is virtualized or not ?
  2. Can we have a way to show "this value is virtualized!" annotation ?

> Now, it comes two questions:
> 1) Do you agree with that, for files like /proc/stat ? I think the most
> important part is to be consistent inside the system, regardless of what
> is done
>
I think some kind of care for users are required as I wrote above.

> 2) Will cpuacct stay? I think if it does, that becomes almost mandatory
> (at least the bind mount idea is pretty much over here), because drawing
> value for /proc/stat becomes quite complex.
> The cpuacct cgroup can provide user, sys, etc values. But we also have:
>

If virtualized /proc/stat works, I don't think 'account only' cgroup is
necessary. It can be obsolete.

Thanks,
-Kame

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Tue, 06 Dec 2011 00:17:06 GMT
View Forum Message <> Reply to Message

On 12/05/2011 10:05 PM, KAMEZAWA Hiroyuki wrote:
> On Mon, 5 Dec 2011 07:32:33 -0200
> Glauber Costa<glommer@parallels.com> wrote:
>
>> Hi,
>>
>> Specially Peter and Paul, but all the others:
>>
>> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
>> to that, there is a question - one I've asked before but without that
>> much of an audience - of whether /proc files read from process living on
>> cgroups should display global or per-cgroup resources.
>>
>> In the past, I was arguing for a knob to control that, but I recently
>> started to believe that a knob here will only overcomplicate matters:
>> if you live in a cgroup, you should display only the resources you can
>> possibly use. Global is for whoever is in the main cgroup.
>>
>
> Hm. I have a suggestion and a concern.
>
> (A suggestion)
>   How about having a mount option for procfs ?
>   For example,
> mount -t proc .... -o cgroup_virtualized
>   Then, /proc/stat etc shows per-cgroup information.
>
> (A concern)
>   /proc/stat will be a mixture of virtualized values and not-virtualized values.
>   1. Don't users need to know whether each value is virtualized or not ?
>   2. Can we have a way to show "this value is virtualized!" annotation ?

A mount options works for me.
However, "work" doesn't mean it is really necessary, and that's the real
question: is there a real use case for someone resource-constrained to
see system-wide values ?

What is exactly the expectation here? If you want to see whichever
resources you're entitled to, just showing the cgroups version on /proc
would be simpler, with less confusion, and do the right thing.

As for your concerns:
1) As said above, my point of view is that no, they do not. You only see
what you can touch.
2) I think this defeats the goal of transparency. Users of fully
virtualized VMs for instances, have no annotations like that (of course
it is possible to hint they are virtualized from many other sources).
But in the end of the day, a resource is a resource, virtualized or not.

>
>> Now, it comes two questions:
>> 1) Do you agree with that, for files like /proc/stat ? I think the most
>> important part is to be consistent inside the system, regardless of what
>> is done
>>
> I think some kind of care for users are required as I wrote above.
>
Please note again that I don't necessarily dislike the idea of a mount
option, if we must do something. I just don't see the point.

>> 2) Will cpuacct stay? I think if it does, that becomes almost mandatory
>> (at least the bind mount idea is pretty much over here), because drawing
>> value for /proc/stat becomes quite complex.
>> The cpuacct cgroup can provide user, sys, etc values. But we also have:
>>
>
> If virtualized /proc/stat works, I don't think 'account only' cgroup is
> necessary. It can be obsolete.
>
Let's keep in mind that there are more to the story, and I want to be
sure to address everyones PoVs here. The impression I've got is that the
reasons to keep cpuacct around came mainly from 2 sources:

1) Balbir wants statistics that don't interfere with scheduling decisions
2) Paul thinks we should avoid cluttering cpu cgroup.

Obsoleting cpuacct clearly makes somethings simpler, but can probably
defeat some of those goals. So still need to hear about this...

Subject: Re: How to draw values for /proc/stat
Posted by Zhu Yanhai on Wed, 07 Dec 2011 14:17:51 GMT
View Forum Message <> Reply to Message

2011/12/5 Glauber Costa <glommer@parallels.com>:

> Hi,
>
> Specially Peter and Paul, but all the others:
>
> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer to
> that, there is a question - one I've asked before but without that much of
> an audience - of whether /proc files read from process living on cgroups
> should display global or per-cgroup resources.
>
> In the past, I was arguing for a knob to control that, but I recently
> started to believe that a knob here will only overcomplicate matters:
> if you live in a cgroup, you should display only the resources you can
> possibly use. Global is for whoever is in the main cgroup.
>
> Now, it comes two questions:
> 1) Do you agree with that, for files like /proc/stat ? I think the most
> important part is to be consistent inside the system, regardless of what is
> done
>
> 2) Will cpuacct stay? I think if it does, that becomes almost mandatory (at
> least the bind mount idea is pretty much over here), because drawing value
> for /proc/stat becomes quite complex.
> The cpuacct cgroup can provide user, sys, etc values. But we also have:
>
> * nr_context_switches,
> * jiffies since boot,
> * total_forks,
> * nr_running,
> * nr_iowait,
>
> Now I doubt any of us want to see /proc/stat extended to accommodate things
> like nr_context_switches, or even worse, nr_running. The way I see it, there
> are two options here:
>
> a) moving everything to cpu cgroup so we keep all values being drawn
> from the same place
> b) Collect that info from multiple places in a transparent way. ctx,
> nr_running and nr_iowait will probably come from cpu. jiffies can
> come from wherever, and maybe we can even draw total_forks
> from Frederic's and avoid counting it twice.
> --
> 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

Hi,
I think making /proc files read from process living on cgroups display
per-cgroup resources is a good idea, at least from a common user's

perspective. We are (well, we will) setup a large cluster with
lxc/cgroup for some backend online services in the next months, and
one gap we see is the entries under /proc are not virtualized enough,
especially those performance counters, not only schedule counters
(e.g. /proc/diskstat). Although we can read some numbers in the host
from blkio controller's counters like blkio.io_serviced,
blkio.io_service_time etc, it would be very convenient if the entries
under /proc are virtualized, as we can deploy various existing
maintenance tools directly in the containers, without developing
another monitors. So the over-cost for maintenance can be low.

Let's include lxc-user mailing list for this topic.

--
Thanks,
Zhu Yanhai

Subject: Re: How to draw values for /proc/stat
Posted by Peter Zijlstra on Fri, 09 Dec 2011 14:03:31 GMT
View Forum Message <> Reply to Message

On Mon, 2011-12-05 at 07:32 -0200, Glauber Costa wrote:
> Hi,
>
> Specially Peter and Paul, but all the others:
>
> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
> to that, there is a question - one I've asked before but without that
> much of an audience - of whether /proc files read from process living on
> cgroups should display global or per-cgroup resources.
>
> In the past, I was arguing for a knob to control that, but I recently
> started to believe that a knob here will only overcomplicate matters:
> if you live in a cgroup, you should display only the resources you can
> possibly use. Global is for whoever is in the main cgroup.
>
> Now, it comes two questions:
> 1) Do you agree with that, for files like /proc/stat ? I think the most
> important part is to be consistent inside the system, regardless of what
> is done

Personally I don't give a rats arse about (/proc vs) cgroups :-)
Currently /proc is unaffected by whatever cgroup you happen to be in and
that seems to make some sort of sense.

Namespaces seem to be about limiting visibility, cgroups about
controlling resources.

The two things are hopelessly disjoint atm, but I believe someone was looking at this mess.

IOW a /proc namespace coupled to cgroup scope would do what you want. Now my head hurts..

> 2) Will cpuacct stay?

I really want to kill it. Balbir seems to want to retain a control-less accounting capability, but I really don't see the point in that. Nor is anybody selling it convincingly.

So I think I'll simply deprecate it and schedule it for removal and anybody wanting something like this is free to send patches to implement what they want in the cpu controller and convince me its worth the pain.

> I think if it does, that becomes almost mandatory

You're referring to #1 here, right?

> (at least the bind mount idea is pretty much over here), because drawing
> value for /proc/stat becomes quite complex.
> The cpuacct cgroup can provide user, sys, etc values. But we also have:
>
> * nr_context_switches,
> * jiffies since boot,
> * total_forks,
> * nr_running,
> * nr_iowait,
>
> Now I doubt any of us want to see /proc/stat extended to accommodate
> things like nr_context_switches, or even worse, nr_running. The way I
> see it, there are two options here:

Why would we want to display all those anyway?

> a) moving everything to cpu cgroup so we keep all values being drawn
> from the same place

This is /cgroup/$controller.stat, right?

> b) Collect that info from multiple places in a transparent way. ctx,
> nr_running and nr_iowait will probably come from cpu. jiffies can
> come from wherever, and maybe we can even draw total_forks
> from Frederic's and avoid counting it twice.

trouble with that is 'drawing from someplace' is a total nightmare, what

happens if that fork muck from Frederic isn't co-mounted with the cpu
controller?

Subject: Re: How to draw values for /proc/stat
Posted by Peter Zijlstra on Fri, 09 Dec 2011 14:07:00 GMT
View Forum Message <> Reply to Message

On Wed, 2011-12-07 at 22:17 +0800, Zhu Yanhai wrote:
> Let's include lxc-user mailing list for this topic.

How many rotten cgroup related lists are there, it already had cgroups@
and containers@, which IMO is 1 too many already.

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Fri, 09 Dec 2011 14:55:53 GMT
View Forum Message <> Reply to Message

On 12/09/2011 12:03 PM, Peter Zijlstra wrote:
> On Mon, 2011-12-05 at 07:32 -0200, Glauber Costa wrote:
>> Hi,
>>
>> Specially Peter and Paul, but all the others:
>>
>> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
>> to that, there is a question - one I've asked before but without that
>> much of an audience - of whether /proc files read from process living on
>> cgroups should display global or per-cgroup resources.
>>
>> In the past, I was arguing for a knob to control that, but I recently
>> started to believe that a knob here will only overcomplicate matters:
>> if you live in a cgroup, you should display only the resources you can
>> possibly use. Global is for whoever is in the main cgroup.
>>
>> Now, it comes two questions:
>> 1) Do you agree with that, for files like /proc/stat ? I think the most
>> important part is to be consistent inside the system, regardless of what
>> is done
>
> Personally I don't give a rats arse about (/proc vs) cgroups :-)
> Currently /proc is unaffected by whatever cgroup you happen to be in and
> that seems to make some sort of sense.
>
> Namespaces seem to be about limiting visibility, cgroups about
> controlling resources.
>

> The two things are hopelessly disjoint atm, but I believe someone was
> looking at this mess.

I did take a look at this (if anyone else was, I'd like to know so we
can share some ideas), but I am not convinced we should do anything to
join them anymore. We virtualization people are to the best of my
knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot bigger.

What I am mostly concerned about now, is how consistent they will be.
/proc always being always global indeed does make sense, but my question
still stands: if you live in a resource-controlled world, why should you
even see resources you will never own ?

> IOW a /proc namespace coupled to cgroup scope would do what you want.
> Now my head hurts..

Mine too. The idea is good, but too broad. Boils down to: How do you
couple them? And none of the methods I thought about seemed to make any
sense.

If we really want to have the values in /proc being opted-in, I think
Kamezawa's idea of a mount option is the winner so far.

>> 2) Will cpuacct stay?
>
> I really want to kill it. Balbir seems to want to retain a control-less
> accounting capability, but I really don't see the point in that. Nor is
> anybody selling it convincingly.
>
> So I think I'll simply deprecate it and schedule it for removal and
> anybody wanting something like this is free to send patches to implement
> what they want in the cpu controller and convince me its worth the pain.
>
>> I think if it does, that becomes almost mandatory
>
> You're referring to #1 here, right?

Yes. But that was before Kame suggested a mount option. That would do
just fine in this use case as well/

>> (at least the bind mount idea is pretty much over here), because drawing
>> value for /proc/stat becomes quite complex.
>> The cpuacct cgroup can provide user, sys, etc values. But we also have:
>>
>> * nr_context_switches,
>> * jiffies since boot,
>> * total_forks,

>> * nr_running,
>> * nr_iowait,
>>
>> Now I doubt any of us want to see /proc/stat extended to accommodate
>> things like nr_context_switches, or even worse, nr_running. The way I
>> see it, there are two options here:
>
> Why would we want to display all those anyway?

Because we care about isolation. Our users (and more importantly, their
tools), need to feel they own the box. That's the whole point of it.

Whatever is displayed to the users, need to be (with a switch/mount opt,
if my initial point is really defeated) related to the resources he is
entitled to, not to some global entity he'll never own.
>
>> a) moving everything to cpu cgroup so we keep all values being drawn
>> from the same place
>
> This is /cgroup/$controller.stat, right?

Sorry?

(I just noticed I wasn't that clear myself) I referred more to the act
of collecting values, and where from, than how to.

>
>> b) Collect that info from multiple places in a transparent way. ctx,
>> nr_running and nr_iowait will probably come from cpu. jiffies can
>> come from wherever, and maybe we can even draw total_forks
>> from Frederic's and avoid counting it twice.
>
> trouble with that is 'drawing from someplace' is a total nightmare, what
> happens if that fork muck from Frederic isn't co-mounted with the cpu
> controller?

If it is not co-mounted, we draw the global value. If you don't mount
it, I someone does not mount it, I can assure you he doesn't care about
it. We for sure will.

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Sun, 11 Dec 2011 14:50:56 GMT
View Forum Message <> Reply to Message

On 12/09/2011 03:55 PM, Glauber Costa wrote:
> On 12/09/2011 12:03 PM, Peter Zijlstra wrote:
>> On Mon, 2011-12-05 at 07:32 -0200, Glauber Costa wrote:

>>> Hi,
>>>
>>> Specially Peter and Paul, but all the others:
>>>
>>> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
>>> to that, there is a question - one I've asked before but without that
>>> much of an audience - of whether /proc files read from process living on
>>> cgroups should display global or per-cgroup resources.
>>>
>>> In the past, I was arguing for a knob to control that, but I recently
>>> started to believe that a knob here will only overcomplicate matters:
>>> if you live in a cgroup, you should display only the resources you can
>>> possibly use. Global is for whoever is in the main cgroup.
>>>
>>> Now, it comes two questions:
>>> 1) Do you agree with that, for files like /proc/stat ? I think the most
>>> important part is to be consistent inside the system, regardless of what
>>> is done
>>
>> Personally I don't give a rats arse about (/proc vs) cgroups :-)
>> Currently /proc is unaffected by whatever cgroup you happen to be in and
>> that seems to make some sort of sense.
>>
>> Namespaces seem to be about limiting visibility, cgroups about
>> controlling resources.
>>
>> The two things are hopelessly disjoint atm, but I believe someone was
>> looking at this mess.
>
> I did take a look at this (if anyone else was, I'd like to know so we
> can share some ideas), but I am not convinced we should do anything to
> join them anymore. We virtualization people are to the best of my
> knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot bigger.
>
> What I am mostly concerned about now, is how consistent they will be.
> /proc always being always global indeed does make sense, but my question
> still stands: if you live in a resource-controlled world, why should you
> even see resources you will never own ?
>
>
>> IOW a /proc namespace coupled to cgroup scope would do what you want.
>> Now my head hurts..
>
> Mine too. The idea is good, but too broad. Boils down to: How do you
> couple them? And none of the methods I thought about seemed to make any
> sense.
>
> If we really want to have the values in /proc being opted-in, I think

> Kamezawa's idea of a mount option is the winner so far.
>

Ok:

How about the following patch to achieve this ?

```
From 1e587bf3d97d850c5ba8b9bf375c2e74b38a9891 Mon Sep 17 00:00:00 2001
From: Glauber Costa <glommer@parallels.com>
Date: Fri, 9 Dec 2011 16:03:26 +0300
Subject: [PATCH] Add "proc_overlay" option for cgroup

This patch adds the "proc_overlay" file to cgroup. It is also
available as a mount option, which can be overriden later by
any of the individual cgroups in the hierarchy.

With this option set, we tell the kernel that for the processes
inside cgroups for which proc_overlay = true, we want /proc files
related to this cgroup to show per-cgroup values, not global ones.

This is a must-have in virtualized environments, where the isolation
guarantees should not let processes in a cgroup see resources of
other groups. Adding an option allow us to achieve this without
interferring with other cgroup-users that are not concerned with
isolation.

Signed-off-by: Glauber Costa <glommer@parallels.com>
---
 include/linux/cgroup.h |    6 ++++++
 kernel/cgroup.c        |   37 +++++++++++++++++++++++++++++++++++++
 2 files changed, 43 insertions(+), 0 deletions(-)

diff --git a/include/linux/cgroup.h b/include/linux/cgroup.h
index 1b7f9d5..f0bc2e9 100644
--- a/include/linux/cgroup.h
+++ b/include/linux/cgroup.h
@@ -158,6 +158,7 @@ enum {
   * Clone cgroup values when creating a new child cgroup
   */
  CGRP_CLONE_CHILDREN,
+ CGRP_PROC_OVERLAY,
 };

 /* which pidlist file are we talking about? */
@@ -245,6 +246,11 @@ struct cgroup {
  spinlock_t event_list_lock;
 };
```

```
+static inline bool cgroup_proc_overlay(struct cgroup *cgrp)
+{
+ return test_bit(CGRP_PROC_OVERLAY, &cgrp->flags);
+}
+
 /*
  * A css_set is a structure holding pointers to a set of
  * cgroup_subsys_state objects. This saves space in the task struct
diff --git a/kernel/cgroup.c b/kernel/cgroup.c
index e700abe..313c06c 100644
--- a/kernel/cgroup.c
+++ b/kernel/cgroup.c
@@ -233,6 +233,7 @@ enum {
  ROOT_NOPREFIX, /* mounted subsystems have no named prefix */
  ROOT_CLONE_CHILDREN, /* mounted subsystems starts with clone_children */
  ROOT_NOSUBSYS, /* explicitly asked for 'none' subsystems */
+ ROOT_PROC_OVERLAY, /* mounted subsystems starts with proc_overlay */
 };

 static int cgroup_is_releasable(const struct cgroup *cgrp)
@@ -253,6 +254,10 @@ static inline int clone_children(const struct cgroup *cgrp)
  return test_bit(CGRP_CLONE_CHILDREN, &cgrp->flags);
 }

+static inline int proc_overlay(const struct cgroup *cgrp)
+{
+ return test_bit(CGRP_PROC_OVERLAY, &cgrp->flags);
+}
 /*
  * for_each_subsys() allows you to iterate on each subsystem attached to
  * an active hierarchy
@@ -1054,6 +1059,8 @@ static int cgroup_show_options(struct seq_file *seq, struct vfsmount
*vfs)
   seq_printf(seq, ",release_agent=%s", root->release_agent_path);
  if (clone_children(&root->top_cgroup))
   seq_puts(seq, ",clone_children");
+ if (test_bit(ROOT_PROC_OVERLAY, &root->flags))
+  seq_puts(seq, ",proc_overlay");
  if (strlen(root->name))
   seq_printf(seq, ",name=%s", root->name);
  mutex_unlock(&cgroup_mutex);
@@ -1116,6 +1123,10 @@ static int parse_cgroupfs_options(char *data, struct cgroup_sb_opts
*opts)
    set_bit(ROOT_NOPREFIX, &opts->flags);
    continue;
   }
+  if (!strcmp(token, "proc_overlay")) {
+   set_bit(ROOT_PROC_OVERLAY, &opts->flags);
```

```
+   continue;
+  }
   if (!strcmp(token, "clone_children")) {
    set_bit(ROOT_CLONE_CHILDREN, &opts->flags);
    continue;
@@ -1406,6 +1417,8 @@ static struct cgroupfs_root *cgroup_root_from_opts(struct
cgroup_sb_opts *opts)
   strcpy(root->name, opts->name);
  if (test_bit(ROOT_CLONE_CHILDREN, &opts->flags))
   set_bit(CGRP_CLONE_CHILDREN, &root->top_cgroup.flags);
+ if (test_bit(ROOT_PROC_OVERLAY, &opts->flags))
+  set_bit(CGRP_PROC_OVERLAY, &root->top_cgroup.flags);
  return root;
 }

@@ -3442,6 +3455,22 @@ static int cgroup_write_notify_on_release(struct cgroup *cgrp,
  return 0;
 }

+static u64 cgroup_proc_overlay_read(struct cgroup *cgrp,
+        struct cftype *cft)
+{
+ return proc_overlay(cgrp);
+}
+
+static int cgroup_proc_overlay_write(struct cgroup *cgrp,
+         struct cftype *cft, u64 val)
+{
+ if (val)
+  set_bit(CGRP_PROC_OVERLAY, &cgrp->flags);
+ else
+  clear_bit(CGRP_PROC_OVERLAY, &cgrp->flags);
+ return 0;
+}
+
 /*
  * Unregister event and free resources.
  *
@@ -3662,6 +3691,11 @@ static struct cftype files[] = {
   .read_u64 = cgroup_clone_children_read,
   .write_u64 = cgroup_clone_children_write,
  },
+ {
+  .name = "cgroup.proc_overlay",
+  .read_u64 = cgroup_proc_overlay_read,
+  .write_u64 = cgroup_proc_overlay_write,
+ },
 };
```

```
 static struct cftype cft_release_agent = {
@@ -3794,6 +3828,9 @@ static long cgroup_create(struct cgroup *parent, struct dentry *dentry,
  if (clone_children(parent))
   set_bit(CGRP_CLONE_CHILDREN, &cgrp->flags);

+ if (parent->root->flags & ROOT_PROC_OVERLAY)
+  set_bit(CGRP_PROC_OVERLAY, &cgrp->flags);
+
  for_each_subsys(root, ss) {
   struct cgroup_subsys_state *css = ss->create(ss, cgrp);

--
1.7.6.4
```

File Attachments

1) 0001-Add-proc_overlay-option-for-cgroup.patch, downloaded 1384 times

---

Subject: Re: How to draw values for /proc/stat
Posted by KOSAKI Motohiro on Sun, 11 Dec 2011 19:11:43 GMT
View Forum Message <> Reply to Message

>>> IOW a /proc namespace coupled to cgroup scope would do what you want.
>>> Now my head hurts..
>>
>> Mine too. The idea is good, but too broad. Boils down to: How do you
>> couple them? And none of the methods I thought about seemed to make any
>> sense.
>>
>> If we really want to have the values in /proc being opted-in, I think
>> Kamezawa's idea of a mount option is the winner so far.

```
 > diff --git a/include/linux/cgroup.h b/include/linux/cgroup.h
 > index 1b7f9d5..f0bc2e9 100644
 > --- a/include/linux/cgroup.h
 > +++ b/include/linux/cgroup.h
 > @@ -158,6 +158,7 @@ enum {
 >   * Clone cgroup values when creating a new child cgroup
 >   */
 >  CGRP_CLONE_CHILDREN,
 > + CGRP_PROC_OVERLAY,
 > };
```

I'm not cgroup expert, but I doubt it is mount option. I suspect it's
cgroup option. That's said, if we have following two directories,

/cgroup-for-virtualization
/cgroup-for-resource-management

are both directory affected the overlay flag? I don't think it is not
optimal. Why? we must care some system software (e.g. kvm, systemd) are
using cgroup internally and we expected this trend will grow more.

So, I doubt namespace issue can be solved by such tiny patch.

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Sun, 11 Dec 2011 20:48:29 GMT
View Forum Message <> Reply to Message

On 12/11/2011 08:11 PM, KOSAKI Motohiro wrote:
>
>>>> IOW a /proc namespace coupled to cgroup scope would do what you want.
>>>> Now my head hurts..
>>>
>>> Mine too. The idea is good, but too broad. Boils down to: How do you
>>> couple them? And none of the methods I thought about seemed to make any
>>> sense.
>>>
>>> If we really want to have the values in /proc being opted-in, I think
>>> Kamezawa's idea of a mount option is the winner so far.
>
> > diff --git a/include/linux/cgroup.h b/include/linux/cgroup.h
> > index 1b7f9d5..f0bc2e9 100644
> > --- a/include/linux/cgroup.h
> > +++ b/include/linux/cgroup.h
> > @@ -158,6 +158,7 @@ enum {
> > * Clone cgroup values when creating a new child cgroup
> > */
> > CGRP_CLONE_CHILDREN,
> > + CGRP_PROC_OVERLAY,
> > };
>
> I'm not cgroup expert, but I doubt it is mount option. I suspect it's
> cgroup option. That's said, if we have following two directories,

Actually, the way I proposed, you have both ways. The mount option is
more a default value for convenience, that is effective until you change
a file. That's the same way as clone_children already do, and I believe
it to be a sane thing.

> /cgroup-for-virtualization
> /cgroup-for-resource-management
>

> are both directory affected the overlay flag?

It depends. The flag is per-cgroup, therefore per-directory. So even if
you set the mount option, you can override it in an individual cgroup.

> I don't think it is not
> optimal. Why? we must care some system software (e.g. kvm, systemd) are
> using cgroup internally and we expected this trend will grow more.

As I said before, each directory has its own files, so in a standard
system, we would be more than happy to set it to 1 in the cgroups
corresponding to our containers, and leave the rest of the world alone.

> So, I doubt namespace issue can be solved by such tiny patch.
>
I don't fully get what you mean here

Subject: Re: How to draw values for /proc/stat
Posted by KAMEZAWA Hiroyuki on Mon, 12 Dec 2011 00:31:16 GMT
View Forum Message <> Reply to Message

On Sun, 11 Dec 2011 15:50:56 +0100
Glauber Costa <glommer@parallels.com> wrote:

> On 12/09/2011 03:55 PM, Glauber Costa wrote:
> > On 12/09/2011 12:03 PM, Peter Zijlstra wrote:
> >> On Mon, 2011-12-05 at 07:32 -0200, Glauber Costa wrote:
> >>> Hi,
> >>>
> >>> Specially Peter and Paul, but all the others:
> >>>
> >>> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
> >>> to that, there is a question - one I've asked before but without that
> >>> much of an audience - of whether /proc files read from process living on
> >>> cgroups should display global or per-cgroup resources.
> >>>
> >>> In the past, I was arguing for a knob to control that, but I recently
> >>> started to believe that a knob here will only overcomplicate matters:
> >>> if you live in a cgroup, you should display only the resources you can
> >>> possibly use. Global is for whoever is in the main cgroup.
> >>>
> >>> Now, it comes two questions:
> >>> 1) Do you agree with that, for files like /proc/stat ? I think the most
> >>> important part is to be consistent inside the system, regardless of what
> >>> is done
> >>
> >> Personally I don't give a rats arse about (/proc vs) cgroups :-)

> >> Currently /proc is unaffected by whatever cgroup you happen to be in and
> >> that seems to make some sort of sense.
> >>
> >> Namespaces seem to be about limiting visibility, cgroups about
> >> controlling resources.
> >>
> >> The two things are hopelessly disjoint atm, but I believe someone was
> >> looking at this mess.
> >
> > I did take a look at this (if anyone else was, I'd like to know so we
> > can share some ideas), but I am not convinced we should do anything to
> > join them anymore. We virtualization people are to the best of my
> > knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot bigger.
> >
> > What I am mostly concerned about now, is how consistent they will be.
> > /proc always being always global indeed does make sense, but my question
> > still stands: if you live in a resource-controlled world, why should you
> > even see resources you will never own ?
> >
> >
> >> IOW a /proc namespace coupled to cgroup scope would do what you want.
> >> Now my head hurts..
> >
> > Mine too. The idea is good, but too broad. Boils down to: How do you
> > couple them? And none of the methods I thought about seemed to make any
> > sense.
> >
> > If we really want to have the values in /proc being opted-in, I think
> > Kamezawa's idea of a mount option is the winner so far.
> >
>
> Ok:
>
> How about the following patch to achieve this ?

Hmm, What I thought was mount option for procfs. Containers will mount its own
/proc file systems. Do you have any pros. / cons. ?
IIUC, cgroup can be mounted per subsystems. Then, options can be passed per
subsystems. It's a mess but we don't need to bring this to procfs.

How about

```
  # mount -t procfs proc /container_root/proc -o cgroup_aware
```

to show cgroup aware procfs ? I think this will be easy to be used with
namespace/chroot, etc.

Thanks,

-Kame

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Mon, 12 Dec 2011 07:06:34 GMT
View Forum Message <> Reply to Message

On 12/12/2011 04:31 AM, KAMEZAWA Hiroyuki wrote:
> On Sun, 11 Dec 2011 15:50:56 +0100
> Glauber Costa<glommer@parallels.com> wrote:
>
>> On 12/09/2011 03:55 PM, Glauber Costa wrote:
>>> On 12/09/2011 12:03 PM, Peter Zijlstra wrote:
>>>> On Mon, 2011-12-05 at 07:32 -0200, Glauber Costa wrote:
>>>>> Hi,
>>>>>
>>>>> Specially Peter and Paul, but all the others:
>>>>>
>>>>> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my answer
>>>>> to that, there is a question - one I've asked before but without that
>>>>> much of an audience - of whether /proc files read from process living on
>>>>> cgroups should display global or per-cgroup resources.
>>>>>
>>>>> In the past, I was arguing for a knob to control that, but I recently
>>>>> started to believe that a knob here will only overcomplicate matters:
>>>>> if you live in a cgroup, you should display only the resources you can
>>>>> possibly use. Global is for whoever is in the main cgroup.
>>>>>
>>>>> Now, it comes two questions:
>>>>> 1) Do you agree with that, for files like /proc/stat ? I think the most
>>>>> important part is to be consistent inside the system, regardless of what
>>>>> is done
>>>>
>>>> Personally I don't give a rats arse about (/proc vs) cgroups :-)
>>>> Currently /proc is unaffected by whatever cgroup you happen to be in and
>>>> that seems to make some sort of sense.
>>>>
>>>> Namespaces seem to be about limiting visibility, cgroups about
>>>> controlling resources.
>>>>
>>>> The two things are hopelessly disjoint atm, but I believe someone was
>>>> looking at this mess.
>>>
>>> I did take a look at this (if anyone else was, I'd like to know so we
>>> can share some ideas), but I am not convinced we should do anything to
>>> join them anymore. We virtualization people are to the best of my
>>> knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot bigger.
>>>

>>> What I am mostly concerned about now, is how consistent they will be.
>>> /proc always being always global indeed does make sense, but my question
>>> still stands: if you live in a resource-controlled world, why should you
>>> even see resources you will never own ?
>>>
>>>
>>>> IOW a /proc namespace coupled to cgroup scope would do what you want.
>>>> Now my head hurts..
>>>
>>> Mine too. The idea is good, but too broad. Boils down to: How do you
>>> couple them? And none of the methods I thought about seemed to make any
>>> sense.
>>>
>>> If we really want to have the values in /proc being opted-in, I think
>>> Kamezawa's idea of a mount option is the winner so far.
>>>
>>
>> Ok:
>>
>> How about the following patch to achieve this ?
>
> Hmm, What I thought was mount option for procfs. Containers will mount its own
> /proc file systems. Do you have any pros. / cons. ?
> IIUC, cgroup can be mounted per subsystems. Then, options can be passed per
> subsystems. It's a mess but we don't need to bring this to procfs.
>
> How about
>
>   # mount -t procfs proc /container_root/proc -o cgroup_aware
>
> to show cgroup aware procfs ? I think this will be easy to be used with
> namespace/chroot, etc.
>

Don't think it works.

Because whoever mounts the proc filesystem, may not want to be isolated.
But we want him to be.

As an example from our usecase, procfs is mounted inside a container. We
can't assume the container is willing to cooperate. So we need to
establish this from the outside. We can of course force options to be
always added to a procfs mount if it comes from the container, but it is
way more messier than this.

per-cgroup knobs works fine for this because the container cannot
possibly see it or change it in any circumstance.
per-namespace would work as well, but then I don't see how to specify a

want/don't want flag in a sane way.

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Mon, 12 Dec 2011 08:22:25 GMT
View Forum Message <> Reply to Message

On 12/12/2011 11:06 AM, Glauber Costa wrote:
> On 12/12/2011 04:31 AM, KAMEZAWA Hiroyuki wrote:
>> On Sun, 11 Dec 2011 15:50:56 +0100
>> Glauber Costa<glommer@parallels.com> wrote:
>>
>>> On 12/09/2011 03:55 PM, Glauber Costa wrote:
>>>> On 12/09/2011 12:03 PM, Peter Zijlstra wrote:
>>>>> On Mon, 2011-12-05 at 07:32 -0200, Glauber Costa wrote:
>>>>>> Hi,
>>>>>>
>>>>>> Specially Peter and Paul, but all the others:
>>>>>>
>>>>>> As you can see in https://lkml.org/lkml/2011/12/4/178, and in my
>>>>>> answer
>>>>>> to that, there is a question - one I've asked before but without that
>>>>>> much of an audience - of whether /proc files read from process
>>>>>> living on
>>>>>> cgroups should display global or per-cgroup resources.
>>>>>>
>>>>>> In the past, I was arguing for a knob to control that, but I recently
>>>>>> started to believe that a knob here will only overcomplicate matters:
>>>>>> if you live in a cgroup, you should display only the resources you
>>>>>> can
>>>>>> possibly use. Global is for whoever is in the main cgroup.
>>>>>>
>>>>>> Now, it comes two questions:
>>>>>> 1) Do you agree with that, for files like /proc/stat ? I think the
>>>>>> most
>>>>>> important part is to be consistent inside the system, regardless
>>>>>> of what
>>>>>> is done
>>>>>
>>>>> Personally I don't give a rats arse about (/proc vs) cgroups :-)
>>>>> Currently /proc is unaffected by whatever cgroup you happen to be
>>>>> in and
>>>>> that seems to make some sort of sense.
>>>>>
>>>>> Namespaces seem to be about limiting visibility, cgroups about
>>>>> controlling resources.
>>>>>
>>>>> The two things are hopelessly disjoint atm, but I believe someone was

>>>>> looking at this mess.
>>>>
>>>> I did take a look at this (if anyone else was, I'd like to know so we
>>>> can share some ideas), but I am not convinced we should do anything to
>>>> join them anymore. We virtualization people are to the best of my
>>>> knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot
>>>> bigger.
>>>>
>>>> What I am mostly concerned about now, is how consistent they will be.
>>>> /proc always being always global indeed does make sense, but my
>>>> question
>>>> still stands: if you live in a resource-controlled world, why should
>>>> you
>>>> even see resources you will never own ?
>>>>
>>>>
>>>>> IOW a /proc namespace coupled to cgroup scope would do what you want.
>>>>> Now my head hurts..
>>>>
>>>> Mine too. The idea is good, but too broad. Boils down to: How do you
>>>> couple them? And none of the methods I thought about seemed to make any
>>>> sense.
>>>>
>>>> If we really want to have the values in /proc being opted-in, I think
>>>> Kamezawa's idea of a mount option is the winner so far.
>>>>
>>>
>>> Ok:
>>>
>>> How about the following patch to achieve this ?
>>
>> Hmm, What I thought was mount option for procfs. Containers will mount
>> its own
>> /proc file systems. Do you have any pros. / cons. ?
>> IIUC, cgroup can be mounted per subsystems. Then, options can be
>> passed per
>> subsystems. It's a mess but we don't need to bring this to procfs.
>>
>> How about
>>
>> # mount -t procfs proc /container_root/proc -o cgroup_aware
>>
>> to show cgroup aware procfs ? I think this will be easy to be used with
>> namespace/chroot, etc.
>>
>
> Don't think it works.
>

> Because whoever mounts the proc filesystem, may not want to be isolated.
> But we want him to be.
>
> As an example from our usecase, procfs is mounted inside a container. We
> can't assume the container is willing to cooperate. So we need to
> establish this from the outside. We can of course force options to be
> always added to a procfs mount if it comes from the container, but it is
> way more messier than this.
>
> per-cgroup knobs works fine for this because the container cannot
> possibly see it or change it in any circumstance.
> per-namespace would work as well, but then I don't see how to specify a
> want/don't want flag in a sane way.
>

There is another aspect of this as well - that I myself was overlooking.
/proc is not the only place in which this knob to work.

Think of syscalls like sysinfo, for instance. We'd also like this
information to come from a cgroup-specific place. Possibly other places
as well.

This is one more reason for me to believe that if we are going for a
switch, it needs to live in the cgroup - and also that "proc_overlay" is
quite a bad name - but that's okay since this small patch was just a
proof of concept to get the discussion going.

Subject: Re: How to draw values for /proc/stat
Posted by Peter Zijlstra on Mon, 12 Dec 2011 09:33:33 GMT
View Forum Message <> Reply to Message

On Fri, 2011-12-09 at 12:55 -0200, Glauber Costa wrote:
> On 12/09/2011 12:03 PM, Peter Zijlstra wrote:

> > > Namespaces seem to be about limiting visibility, cgroups about
> > > controlling resources.
> > >
> > > The two things are hopelessly disjoint atm, but I believe someone was
> > > looking at this mess.
> >
> > I did take a look at this (if anyone else was, I'd like to know so we
> > can share some ideas), but I am not convinced we should do anything to
> > join them anymore. We virtualization people are to the best of my
> > knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot bigger.
> >
> > What I am mostly concerned about now, is how consistent they will be.
> > /proc always being always global indeed does make sense, but my question

> still stands: if you live in a resource-controlled world, why should you
> even see resources you will never own ?

Since without namespaces you can still see the rest of the world. So it
makes sense to me to still see all resources too.

Also, proportional controllers might not see a consistent slice of the
resource, making the stats rather awkward to interpret.

Furthermore, not everybody might care about these statistics at all and
I know pjt objected to being subjected to the extra accounting (pjt do
speak up etc..).

> If it is not co-mounted, we draw the global value. If you don't mount
> it, I someone does not mount it, I can assure you he doesn't care about
> it. We for sure will.

Anyway, looking at the rest of the emails in this thread the current
proposal is a cgroup mount option that indicates if you want these
per-cgroup stats or not, right?

Subject: Re: How to draw values for /proc/stat
Posted by Glauber Costa on Mon, 12 Dec 2011 09:35:09 GMT
View Forum Message <> Reply to Message

On 12/12/2011 01:33 PM, Peter Zijlstra wrote:
> On Fri, 2011-12-09 at 12:55 -0200, Glauber Costa wrote:
>> On 12/09/2011 12:03 PM, Peter Zijlstra wrote:
>
>>> Namespaces seem to be about limiting visibility, cgroups about
>>> controlling resources.
>>>
>>> The two things are hopelessly disjoint atm, but I believe someone was
>>> looking at this mess.
>>
>> I did take a look at this (if anyone else was, I'd like to know so we
>> can share some ideas), but I am not convinced we should do anything to
>> join them anymore. We virtualization people are to the best of my
>> knowledge the only ones doing namespaces. Cgroups, OTOH, got a lot bigger.
>>
>> What I am mostly concerned about now, is how consistent they will be.
>> /proc always being always global indeed does make sense, but my question
>> still stands: if you live in a resource-controlled world, why should you
>> even see resources you will never own ?
>
> Since without namespaces you can still see the rest of the world. So it
> makes sense to me to still see all resources too.

>
> Also, proportional controllers might not see a consistent slice of the
> resource, making the stats rather awkward to interpret.
>
> Furthermore, not everybody might care about these statistics at all and
> I know pjt objected to being subjected to the extra accounting (pjt do
> speak up etc..).
>
>> If it is not co-mounted, we draw the global value. If you don't mount
>> it, I someone does not mount it, I can assure you he doesn't care about
>> it. We for sure will.
>
> Anyway, looking at the rest of the emails in this thread the current
> proposal is a cgroup mount option that indicates if you want these
> per-cgroup stats or not, right?

Well, it is something in this direction. I don't think it's entirely
clear what exactly it will look like, but it seems we're making progress.
