Subject: [PATCH 0/4] Container Freezer: Reuse Suspend Freezer
Posted by Matt Helsley on Mon, 07 Jul 2008 22:58:23 GMT
View Forum Message <> Reply to Message

This patchset reuses the container infrastructure and the swsusp freezer to
freeze a group of tasks.

The freezer subsystem in the container filesystem defines a file named
freezer.state. Writing "FROZEN" to the state file will freeze all tasks in the
cgroup. Subsequently writing "RUNNING" will unfreeze the tasks in the cgroup.
Reading will return the current state.

* Examples of usage :

```
  # mkdir /containers/freezer
  # mount -t cgroup -ofreezer,signal freezer  /containers
  # mkdir /containers/0
  # echo $some_pid > /containers/0/tasks
```

to get status of the freezer subsystem :

```
  # cat /containers/0/freezer.state
  RUNNING
```

to freeze all tasks in the container :

```
  # echo FROZEN > /containers/0/freezer.state
  # cat /containers/0/freezer.state
  FREEZING
  # cat /containers/0/freezer.state
  FROZEN
```

to unfreeze all tasks in the container :

```
  # echo RUNNING > /containers/0/freezer.state
  # cat /containers/0/freezer.state
  RUNNING
```

to kill all tasks in the container :

```
  # echo 9 > /containers/0/signal.kill
```

I've reworked Cedric's patches to use task_lock() to protect access to the
task's cgroup.

Paul, Pavel asked me to send these to Rafael next. They are patches to make
the freezer useful for checkpoint/restart using cgroups so it would be nice
to get an explicit [N]Ack from you first.

Rafael, if Paul agrees, please consider applying these patches.

Changes since v3:
v4 (Almost all of these changes are confined to patch 3):
 Reworked the series to use task_lock() instead of RCU.
 Reworked the series to use write_string() and read_seq_string()
  cgroup methods.
 Fixed the race Paul Menage identified.
 Fixed up check_if_frozen() to do more than just test the FROZEN
  flag. In some cases tasks could be stopped (T) and marked
  FREEZING. When that happens we can safely assume that it
  will be frozen immediately upon waking up in the kernel.
  Waiting for it to get marked with PF_FROZEN in order to
  transition to the FROZEN state would block unnecessarily.
 Removed freezer_ prefix from static functions in cgroup_freezer.c.
 Simplified STATE_ switch.
 Updated the locking comments.
v3:
 Ported to 2.6.26-rc5-mm2 with Rafael's freezer patches
 Tested on 24 combinations of 3 architectures (x86, x86_64, ppc64)
  with 8 different kernel configs varying power management
  and cgroup config variables. Each patch builds and boots
  in these 24 combinations.
 Passes functional testing.
v2 (roughly patches 3 and 5):
 Moved the "kill" file into a separate cgroup subsystem (signal) and
  it's own patch.
 Changed the name of the file from freezer.freeze to freezer.state.
 Switched from taking 1 and 0 as input to the strings "FROZEN" and
  "RUNNING", respectively. This helps keep the interface
  human-usable if/when we need to more states.
 Checked that stopped or interrupted is "frozen enough"
  Since try_to_freeze() is called upon wakeup of these tasks
  this should be fine. This idea comes from recent changes to
  the freezer.
 Checked that if (task == current) whilst freezing cgroup we're ok
 Fixed bug where -EBUSY would always be returned when freezing
 Added code to handle userspace retries for any remaining -EBUSY

Cheers,
 -Matt Helsley

--

Subject: [PATCH 4/4] Container Freezer: Skip frozen cgroups during power management resume
Posted by Matt Helsley on Mon, 07 Jul 2008 22:58:27 GMT
View Forum Message <> Reply to Message

From: Cedric Le Goater <clg@fr.ibm.com>
Subject: [PATCH 4/4] Container Freezer: Skip frozen cgroups during power management resume

When a system is resumed after a suspend, it will also unfreeze
frozen cgroups.

This patchs modifies the resume sequence to skip the tasks which
are part of a frozen control group.

Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>
Signed-off-by: Matt Helsley <matthltc@us.ibm.com>
Tested-by: Matt Helsley <matthltc@us.ibm.com>
---
 kernel/power/process.c | 4 ++++
 1 file changed, 4 insertions(+)

Index: linux-2.6.26-rc5-mm2/kernel/power/process.c
===
--- linux-2.6.26-rc5-mm2.orig/kernel/power/process.c
+++ linux-2.6.26-rc5-mm2/kernel/power/process.c
@@ -11,10 +11,11 @@
 #include <linux/interrupt.h>
 #include <linux/suspend.h>
 #include <linux/module.h>
 #include <linux/syscalls.h>
 #include <linux/freezer.h>
+#include <linux/cgroup_freezer.h>

 /*
  * Timeout for stopping processes
  */
 #define TIMEOUT (20 * HZ)
@@ -133,10 +134,13 @@ static void thaw_tasks(bool nosig_only)
    continue;

   if (nosig_only && should_send_signal(p))
    continue;

+ if (cgroup_frozen(p))
+   continue;
+
   thaw_process(p);
  } while_each_thread(g, p);
  read_unlock(&tasklist_lock);

}

--

_____

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

---

Subject: Re: [PATCH 0/4] Container Freezer: Reuse Suspend Freezer
Posted by Matt Helsley on Mon, 07 Jul 2008 23:02:24 GMT
View Forum Message <> Reply to Message

On Mon, 2008-07-07 at 15:58 -0700, Matt Helsley wrote:
> This patchset reuses the container infrastructure and the swsusp freezer to
> freeze a group of tasks.
>
> The freezer subsystem in the container filesystem defines a file named
> freezer.state. Writing "FROZEN" to the state file will freeze all tasks in the
> cgroup. Subsequently writing "RUNNING" will unfreeze the tasks in the cgroup.
> Reading will return the current state.
>
> * Examples of usage :
>
>   # mkdir /containers/freezer
>   # mount -t cgroup -ofreezer,signal freezer /containers
>   # mkdir /containers/0
>   # echo $some_pid > /containers/0/tasks
>
> to get status of the freezer subsystem :
>
>   # cat /containers/0/freezer.state
>   RUNNING
>
> to freeze all tasks in the container :
>
>   # echo FROZEN > /containers/0/freezer.state
>   # cat /containers/0/freezer.state
>   FREEZING
>   # cat /containers/0/freezer.state
>   FROZEN
>
> to unfreeze all tasks in the container :
>
>   # echo RUNNING > /containers/0/freezer.state
>   # cat /containers/0/freezer.state
>   RUNNING

>
> to kill all tasks in the container :
>
>   # echo 9 > /containers/0/signal.kill
>
> I've reworked Cedric's patches to use task_lock() to protect access to the
> task's cgroup.
>
> Paul, Pavel asked me to send these to Rafael next. They are patches to make
> the freezer useful for checkpoint/restart using cgroups so it would be nice
> to get an explicit [N]Ack from you first.
>
> Rafael, if Paul agrees, please consider applying these patches.
>
> Changes since v3:
> v4 (Almost all of these changes are confined to patch 3):
> Reworked the series to use task_lock() instead of RCU.
> Reworked the series to use write_string() and read_seq_string()
>   cgroup methods.

 FYI - This means these patches need Paul's patches introducing
write_string(). I can certainly restore the old code for .read
and .write, but I was anticipating write_string() making it into various
trees first. If that's not necessarily the case please let me know.

Cheers,
 -Matt

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

---

Subject: Re: [PATCH 0/4] Container Freezer: Reuse Suspend Freezer
Posted by KAMEZAWA Hiroyuki on Tue, 08 Jul 2008 03:27:40 GMT
View Forum Message <> Reply to Message

Hi, could I make brief questions ?

On Mon, 07 Jul 2008 15:58:23 -0700
Matt Helsley <matthltc@us.ibm.com> wrote:
> to get status of the freezer subsystem :
>
>   # cat /containers/0/freezer.state
>   RUNNING
>
> to freeze all tasks in the container :

>
>   # echo FROZEN > /containers/0/freezer.state
>   # cat /containers/0/freezer.state
>   FREEZING
>   # cat /containers/0/freezer.state
>   FROZEN
>
I'm just curious.

1. When we see FREEZING and have to retry ?
   While there are some threads which wait for some event ?

2. What happens when FROZEN threads are moved to other group ?
   Can we move them ?
   Can we wake them up (if it can be moved) ?
   What operations are allowed to frozen threads ?

Thanks,
-Kame

Subject: Re: [PATCH 0/4] Container Freezer: Reuse Suspend Freezer
Posted by Matt Helsley on Tue, 08 Jul 2008 19:39:14 GMT
View Forum Message <> Reply to Message

Cedric was accidentally not Cc'd on the introduction (PATCH 0). Adding
him.

On Tue, 2008-07-08 at 12:31 +0900, KAMEZAWA Hiroyuki wrote:
> Hi, could I make brief questions ?
>
> On Mon, 07 Jul 2008 15:58:23 -0700
> Matt Helsley <matthltc@us.ibm.com> wrote:
> > to get status of the freezer subsystem :
> >
> >   # cat /containers/0/freezer.state
> >   RUNNING
> >
> > to freeze all tasks in the container :
> >
> >   # echo FROZEN > /containers/0/freezer.state
> >   # cat /containers/0/freezer.state
> >   FREEZING

> > # cat /containers/0/freezer.state
> > FROZEN
> >
> I'm just curious.
>
> 1. When we see FREEZING and have to retry ?

One example is when some processes are in the a specific portion of vfork() you might see FREEZING and have to retry.

> While there are some threads which wait for some event ?

Depending on which kind of "wait" they are performing, yes. If it's uninterruptible sleep and the PF_FROZEN flag is not set then you might see FREEZING.

> 2. What happens when FROZEN threads are moved to other group ?
> Can we move them ?

If the destination cgroup is not FROZEN, yes.

If the destination cgroup is FREEZING this works as expected.

If the destination cgroup is RUNNING you'd have a problem unfreezing the task. This happends because the cgroup has a state inconsistent with the task's state. To unfreeze the task you'd have to try to freeze and then unfreeze the destination cgroup.

There are several ways I could change this.

One is to try and disallow users from moving frozen tasks. That doesn't seem like a good approach since it would require a new cgroups interface "can_detach()".

However we can prevent attach so I could add checks that look at the attaching tasks's state and refuse the attach when the task's state (unfrozen, frozen) is inconsistent with the cgroup state (RUNNING, FREEZING, FROZEN). I'll send a fifth patch on top of this series showing this idea.

Rather than refuse to allow attach we could change the destination cgroup's state during attach so that the two states are consistent. However, this introduces more ugly cases for userspace to be aware of.

> Can we wake them up (if it can be moved) ?

You can't wake them until you've unfrozen them.

> What operations are allowed to frozen threads ?

Any operation that doesn't require the threads to run.

Cheers,
-Matt Helsley

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