Subject: Re: [PATCH] Allow signalling container-init
Posted by Sukadev Bhattiprolu on Fri, 10 Aug 2007 00:48:12 GMT
View Forum Message <> Reply to Message

Pavel Emelianov [xemul@openvz.org] wrote:
| Oleg Nesterov wrote:
| >On 08/09, sukadev@us.ibm.com wrote:
| >>Oleg Nesterov [oleg@tv-sign.ru] wrote:
| >>| On 08/08, sukadev@us.ibm.com wrote:
| >>| >
| >>| > From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| >>| > Subject: [PATCH] Allow signalling container-init
| >>| >
| >>| > Only the global-init process must be special - any other
| >>container-init
| >>| > process must be killable to prevent run-away processes in the system.
| >>|
| >>| I think you are right, but....
| >>|
| >>| > --- lx26-23-rc1-mm1.orig/kernel/signal.c 2007-08-07
| >>13:52:12.000000000 -0700
| >>| > +++ lx26-23-rc1-mm1/kernel/signal.c 2007-08-08
| >>15:09:27.000000000 -0700
| >>| > @@ -1861,11 +1861,9 @@ relock:
| >>| >   continue;
| >>| >
| >>| >   /*
| >>| > -   * Init of a pid space gets no signals it doesn't
| >>want from
| >>| > -   * within that pid space. It can of course get
| >>signals from
| >>| > -   * its parent pid space.
| >>| > +   * Global init gets no signals it doesn't want.
| >>| >   */
| >>| > - if (current == task_child_reaper(current))
| >>| > + if (is_global_init(current->group_leader))
| >>| >   continue;
| >>|
| >>| ...this breaks exec() from /sbin/init. Note that de_thread() kills other
| >>| sub-threads with SIGKILL. With this patch de_thread() will hang waiting
| >>| for other threads to die.
| >>
| >>Again for threaded-init I guess :-(
| >>
| >>Well, we discussed last week about allowing non-root users to clone their
| >>pid namespace. The user can then create a container-init and this
| >>process would become immune to signal even by a root user ?
| >

| >please see below,
| >
| >>|
| >>| I think it is better to not change the current behaviour which is not
| >>| perfect (buggy), until we actually protect /sbin/init from unwanted
| >>| signals.
| >>
| >>Can we preserve the existing behavior by checking only the main thread
| >>of global init (i.e pass in 'current' rather than 'current->group_leader'
| >>to is_global_init()) ?
| >
| >Yes, this is what I meant, this is what we have in Linus's tree.
| >This way a container-init could be killed. This all is not correct,
| >but we shouldn't replace one bug with another.
|
| Well, I agree with Oleg. I think that we should keep the patches
| without the signal handling until this set is in (at least) -mm.
| init pid namespace will work without it as used to do, and we'll
| develop a better signal handling and fix existing BUGs.
|
| I know that this creates a hole for creating unkillable process,
| but since this is for root user only (CAP_SYS_ADMIN) this is OK.

But I think there is a difference bw what you are saying and what
Oleg is saying.

Oleg pls correct me if I am wrong, but from what I understand, we
just need modify my earlier fix so we can still terminate the
container from a parent namespace but preserve the existing behavior
w.r.t threaded-inits.

Here is the modified patch for this.

Suka

From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
Subject: [PATCH] Allow signalling container-init

Only the global-init process must be special - any other container-init
process must be killable to prevent run-away processes in the system.

TODO: Ideally we should allow killing the container-init only from parent
 container and prevent it being killed from within the container.
 But that is a more complex change and will be addressed by a follow-on
 patch. For now allow the container-init to be terminated by any process
 with sufficient privileges.

Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>

---
 kernel/signal.c | 6 ++----
 1 file changed, 2 insertions(+), 4 deletions(-)

Index: lx26-23-rc1-mm1/kernel/signal.c
===
--- lx26-23-rc1-mm1.orig/kernel/signal.c 2007-08-07 13:52:12.000000000 -0700
+++ lx26-23-rc1-mm1/kernel/signal.c 2007-08-09 17:22:19.000000000 -0700
@@ -1861,11 +1861,9 @@ relock:
     continue;

    /*
-    * Init of a pid space gets no signals it doesn't want from
-    * within that pid space. It can of course get signals from
-    * its parent pid space.
+   * Global init gets no signals it doesn't want.
     */
-   if (current == task_child_reaper(current))
+   if (is_global_init(current))
     continue;

    if (sig_kernel_stop(signr)) {
_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers