Subject: Re: [PATCH 1/2] Container-init must be immune to unwanted signals
Posted by Sukadev Bhattiprolu on Wed, 31 Oct 2007 01:43:20 GMT
View Forum Message <> Reply to Message

Eric W. Biederman [ebiederm@xmission.com] wrote:
| sukadev@us.ibm.com writes:
|
| > Note: this patch applies on top of Eric's patch:
| >
| > http://lkml.org/lkml/2007/10/26/440
| >
| > ---
| >
| > From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| > Subject: [PATCH 1/2] Container-init must be immune to unwanted signals
| >
| > Container-init process must appear like a normal process to its sibling
| > in the parent namespace and should be killable (or not) in the usual way.
| >
| > But it must be immune to any unwanted signals from within its own namespace.
| >
| > At the time of sending the signal, check if receiver is container-init
| > and if signal is an unwanted one. If its unwanted signal, ignore the
| > signal right away.
| >
| > Note:
| > A limitation with this patch is that if the signal is blocked by the
| > container-init at the time of the check, we cannot ignore the signal
| > because the container-init may install a handler for the signal before
| > unblocking it.
| >
| > But if the container-init unblocks the signal without installing the
| > handler, the unwanted signal will still be delivered to the container-
| > init. If the unwanted signal is fatal (i.e default action is to
| > terminate), we end up terminating the container-init and hence the
| > container.
| >
| > We have not been able to find a clean-way to address this blocked
| > signal issue in the kernel. It appears easier to let the container-
| > init decide what it wants to do with signals i.e have it _explicitly_
| > ignore or handle all fatal signals.
| >
| > The next patch in this set prints a warning the first time a
| > container-init process fork()s without ignoring or handling a fatal
| > signal.
| >
| > Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
| > ---

| > include/linux/pid_namespace.h | 6 +++++-
| > kernel/pid.c | 9 ++++++++-
| > kernel/signal.c | 5 ++++-
| > 3 files changed, 17 insertions(+), 3 deletions(-)
| >
| > Index: 2.6.23-mm1/kernel/signal.c
| > ===
| > --- 2.6.23-mm1.orig/kernel/signal.c 2007-10-27 10:08:36.000000000 -0700
| > +++ 2.6.23-mm1/kernel/signal.c 2007-10-27 10:08:36.000000000 -0700
| > @@ -45,7 +45,10 @@ static int sig_init_ignore(struct task_s
| >
| >   // Currently this check is a bit racy with exec(),
| >   // we can _simplify_ de_thread and close the race.
| > - if (likely(!is_global_init(tsk->group_leader)))
| > + if (likely(!is_container_init(tsk->group_leader)))
| > + return 0;
| > +
| > + if (task_in_descendant_pid_ns(tsk) && !in_interrupt())
| >   return 0;
| >
| >   return 1;
|
| Ok. This is where we are handling the pid namespace case.
| This begins to feel correct.
|
| What is the in_interrupt() check for? That looks bogus on
| the face of it.

It was for the send_sigio() case and trying to prevent that signal
from going to /sbin/init.
|
| I would suggest setting the signal handlers in flush_signal_handlers
| to SIG_IGN but that looks like the children of /sbin/init would
| the a different set of signals by default.
|
| Eric

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