Subject: Re: [RFC][PATCH 15/16] Enable signaling child reaper from parent ns.
Posted by serue on Thu, 24 May 2007 15:59:40 GMT
View Forum Message <> Reply to Message

Quoting sukadev@us.ibm.com (sukadev@us.ibm.com):
>
> Subject: Enable signaling child reaper from parent ns.
>
> From: Sukadev Bhattiprolu <sukadev@us.ibm.com>
>
> The reaper of a child namespace must receive signals from its parent pid
> namespace but not receive any signals from its own namespace.
>
> This is a very early draft :-) and following tests seem to pass
>
> - Successfully kill child reaper from parent namespace (init_pid_ns)
>
> - Fail to kill child reaper from within its namespace (non init_pid_ns)
>
> - kill -1 1 from init_pid_ns seemed to work (rescanned inittab)
>
> TODO:
> - Test async io and SIGIO delivery.
>
> - Allow any legitimate signals that the child reaper can receive
> from within its namespace? (we block all signals now)
>
> - Sending SIGKILL to the child reaper of a namespace terminates the
> namespace But if the namespace remounted /proc from user space,
> /proc would remain mounted even after reaper and other process in
> the namespace go away.
>
> Signed-off-by: Sukadev Bhattiprolu <sukadev@us.ibm.com>
> ---
> kernel/signal.c | 22 ++++++++++++++++++++-
> 1 file changed, 21 insertions(+), 1 deletion(-)
>
> Index: lx26-21-mm2/kernel/signal.c
> ==========
> --- lx26-21-mm2.orig/kernel/signal.c 2007-05-22 16:59:42.000000000 -0700
> +++ lx26-21-mm2/kernel/signal.c 2007-05-22 16:59:57.000000000 -0700
> @@ -507,6 +507,20 @@ static int check_kill_permission(int sig
> && !capable(CAP_KILL))
> return error;
>
> + /*
> + * If t is the reaper of its namespace and someone from that
> + * namespace is trying to send a signal.

> + *
> + * Note: If some one from parent namespace is sending a signal,
> + *   task_child_reaper() != t and we allow the signal.
> + *
> + * In the child namespace, does this block even legitimate signals
> + * like the ones telinit sends to /sbin/init ?
> + *
> + */
> + if ((!is_global_init(t)) && (t == task_child_reaper(t)))

Couldn't you more clearly achieve what you want by doing:
 if ((!is_global_init(t)) && (t == task_child_reaper(current)))

Still like you say I think you need to study more how current code does
the right thing for the global init. Reproduce exactly that if t ==
task_child_reaper(current), else treat like any other task. And though
I said "reproduce", I should think you could do it without separate
checks as you have here.

-serge

> + return -EPERM;
> +
>   error = security_task_kill(t, info, sig, 0);
>   if (!error)
>   audit_signal_info(sig, t); /* Let audit system see the signal */
> @@ -1910,7 +1924,13 @@ relock:
>   /*
>   * 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.
> +   * its parent pid space. But we have no way of knowing the
> +   * namespace from which the signal was sent. For now check
> +   * if we are global init here and add additional checks in
> +   * sys_kill() and friends.
> +   *
> +   * Note that t == task_child_reaper(t) implies t is the global
> +   * init (and we are in init_pid_ns).
>   */
>   if (current == task_child_reaper(current))
>   continue;

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