Subject: Re: [PATCH] Masquerade sender information
Posted by ebiederm on Sun, 04 Nov 2007 04:12:53 GMT
View Forum Message <> Reply to Message

Cedric Le Goater <clg@fr.ibm.com> writes:

> Eric W. Biederman wrote:
>> sukadev@us.ibm.com writes:
>>
>>> +static void masquerade_sender(struct task_struct *t, struct sigqueue *q)
>>> +{
>>> +	/*
>>> +	 * If the sender does not have a pid_t in the receiver's active
>>> +	 * pid namespace, set si_pid to 0 and pretend signal originated
>>> +	 * from the kernel.
>>> +	 */
>>> +	if (!pid_ns_equal(t)) {
>>> +		q->info.si_pid = 0;
>>> +		q->info.si_uid = 0;
>>> +		q->info.si_code = SI_KERNEL;
>>> +	}
>>> +}
>>
>> It looks like we are hooked in the right place. However the way we
>> are handling this appears wrong.
>>
>> First. If we have an si_code that does not use si_pid then we should
>> not be changing si_pid, because the structure is a union and that field
>> is not always a pid value.
>>
>>
>> My gut feel says the code should be something like:
>>
>> switch (q->info->si_code & __SI_MASK) {
>> case __SI_KILL:
>> case __SI_CHILD:
>> case __SI_RT:
>> case __MESQ:
>> 	q->info->si_pid = task_pid_nr_ns(current, t->nsproxy->pid_ns);
>> 	break;
>> }
>
> IMHO, it should be
>
> q->info->si_pid = 0.
>
> we're trying to cover the case where the sender does not have a pid_t in
> the receiver's active pid namespace.

Yes. However you are currently missing the case where the target pid namespace is a parent pid namespace. So besides applying the change to liberally we also missed the case when sending to a parent pid namespace. task_tgid_nr_ns(current, t->nsporxy->pid_ns) handles that.

Technically I think that is safe right now, but I think I would like to pass in the task_struct of the sender because we have a few odd instances where current is not the sender although every case I have traced we do continue to be in the same process group.

Eric

_____

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