Subject: Re: [NETLINK] Don't attach callback to a going-away netlink socket
Posted by Evgeniy Polyakov on Wed, 18 Apr 2007 09:07:20 GMT
View Forum Message <> Reply to Message

On Wed, Apr 18, 2007 at 10:50:42AM +0200, Patrick McHardy (kaber@trash.net) wrote:
> >>It already does (netlink_destroy_callback), but that doesn't help
> >>with this race though since without this patch we don't enter the
> >>error path.
> >
> > I thought that with releasing a socket, which will have a callback
> > attached only results in a leak of the callback? In that case we can
> > just free it in dump() just like it is done in no-error path already.
> > Or do I miss something additional?
>
> That would only work if there is nothing to dump (cb->dump returns 0).
> Otherwise it is not freed.

That is what I referred to as error path. Btw, with positive return
value we end up in subsequent call to input which will free callback
under lock as expected.

I do not object against the patch, just want to make a clear vision about
dumps - if callback is allocated to be used in dump only, then we could
just free it there without passing to next round.

> >>The problem is asynchronous processing of the dump request in the
> >>context of a different process. Process requests a dump, message
> >>is queued and process returns from sendmsg since some other process
> >>is already processing the queue. Then the process closes the socket,
> >>resulting in netlink_release being called. When the dump request
> >>is finally processed the race Pavel described might happen. This
> >>can only happen for netlink families that use mutex_try_lock for
> >>queue processing of course.
> >
> >
> > Doesn't it called from ->sk_data_ready() which is synchronous with
> > respect to sendmsg, not sure about conntrack though, but it looks so?
>
>
> Yes, but for kernel sockets we end up calling the input function,
> which when mutex_trylock is used returns immediately when some
> other process is already processing the queue, so the requesting
> process might close the socket before the request is processed.

So far it is only netfilter and gennetlink, we would see huge dump
from netlink_sock_destruct.
Anyway, that is possible situation, thanks for clearing this up.

--
Evgeniy Polyakov