Subject: Re: [PATCH][XFRM] Fix potential race vs xfrm_state(only)_find and xfrm_hash_resize.
Posted by davem on Fri, 14 Dec 2007 19:39:51 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Thu, 13 Dec 2007 13:56:14 +0300

> The _find calls calculate the hash value using the
> xfrm_state_hmask, without the xfrm_state_lock. But the
> value of this mask can change in the _resize call under
> the state_lock, so we risk to fail in finding the desired
> entry in hash.
>
> I think, that the hash value is better to calculate
> under the state lock.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Thanks for the bug fix.

I know why I coded it this way, I wanted to give GCC more
room to schedule the loads away from the uses in the hash
calculation.

Once you cram it after the spin lock acquire, it can't load unrelated
values earlier to soften the load/use cost on cache misses.

Of course it's invalid because the hash mask can change as you
noticed.

I wish there was a way to conditionally clobber memory, then we could
tell GCC exactly what memory objects are protected by the lock and
thus help in situations like this so much.