Subject: Re: [patch 3/3][NETNS45][V2] remove timewait sockets at cleanup
Posted by den on Thu, 27 Sep 2007 13:21:57 GMT
View Forum Message <> Reply to Message

Daniel Lezcano wrote:

```
> From: Daniel Lezcano <dlezcano@fr.ibm.com>
> 
> Denis Lunev spotted that if we take a reference to the network namespace
> with the timewait sockets, we will need to wait for their expiration to
> have the network namespace freed. This is a waste of time, the timewait
> sockets are for avoiding to receive a duplicate packet from the network,
> if the network namespace is freed, the network stack is removed, so no
> chance to receive any packets from the outside world.
> 
> This patchset remove/destroy the timewait sockets when the
> network namespace is freed.
> 
> Signed-off-by: Daniel Lezcano <dlezcano@fr.ibm.com>
> ---
>  net/ipv4/tcp.c |   53 +++++++++++++++++++++++++++++++++++++++++++++++++++++
>  1 file changed, 53 insertions(+)
> 
```

[...]
This place seems non-trival and broken for me :(May be I am wrong.

```
> +		write_lock_bh(&head->lock);
> +
> +	sk_for_each_safe(sk, node, tmp, &head->twchain) {
> +
> +		tw = inet_twsk(sk);
> +		if (tw->tw_net != net)
> +			continue;
> +
> +		/* deschedule the timewait socket */
> +		spin_lock(&tcp_death_row.death_lock);
> +		if (inet_twsk_del_dead_node(tw)) {
> +			inet_twsk_put(tw);
> +			if (--tcp_death_row.tw_count == 0)
> +				del_timer(&tcp_death_row.tw_timer);
```

There is a call inet_twsk_deschedule which do exactly what we need to

```
void inet_twsk_deschedule(struct inet_timewait_sock *tw,
                          struct inet_timewait_death_row *twdr)
{
        spin_lock(&twdr->death_lock);
        if (inet_twsk_del_dead_node(tw)) {
                inet_twsk_put(tw);
                if (--twdr->tw_count == 0)
                del_timer(&twdr->tw_timer);
        }
        spin_unlock(&twdr->death_lock);
        __inet_twsk_kill(tw, twdr->hashinfo);
}
```

and, from my point of view, your patch [2] is even not needed. We should do

```
restart:
        write_lock_bh(&head->lock);
        sk_for_each_safe(sk, node, tmp, &head->twchain) {
            tw = inet_twsk(sk);
            if (tw->tw_net != net)
                continue;
            sock_hold(sk);
            write_unlock_bh(&head->lock);

            inet_twsk_deschedule(tw, &tcp_death_row);
            inet_twsk_put(tw);
            goto restart;
        }
```

This removes serious locking issue. You have introduced dependency
between write_lock_bh(&head->lock); and
spin_lock(&tcp_death_row.death_lock);
This should be at least checked and documented in the headers. I am not
sure that this is correct.

If my approach is correct, your second patch is not needed.

It will also worth to local_bh_enable() at the very beginning and remove
_bh from write_lock.

Regards,
Den