Subject: Re: cleanup in workq and dst_destroy
Posted by den on Mon, 19 Nov 2007 09:29:38 GMT
View Forum Message <> Reply to Message

Benjamin Thery wrote:
> Denis V. Lunev wrote:
>> Daniel Lezcano wrote:
>>> Denis V. Lunev wrote:
>>>> Daniel Lezcano wrote:
>>>>> Hi all,
>>>>>
>>>>> while doing ipv6 namespace, we were faced to a problem with the loopback
>>>>> and the dst_destroy function.
>>>>>
>>>>> When the network namespace exits, the cleanup function is called by
>>>>> schedule_work and this function will browse the net ops list to call the
>>>>> different exit methods for the registered subsystems.
>>>>>
>>>>> The different subsystems will shutdown their resources and in particular
>>>>> addrconf subsystem will ifdown the loopback. This function will call
>>>>> rt6_ifdown

```
>>>>>  -> fib6_clean_all
>>>>>   -> fib6_clean_node
>>>>>    -> fib6_clean_tree
>>>>>     -> fib6_clean_node
>>>>>      -> fib6_del
>>>>>       -> fib6_del_route
>>>>>        -> rt6_release
>>>>>         ->dst_free
>>>>>          -> __dst_free
```

>>>>>
>>>>> The __dst_free function will schedule_delayed_work the dst_gc_work
>>>>> function.
>>>>>
>>>>> The dst_gc_work will call dst_destroy and finally this one will call
>>>>> dst->ops->destroy ops function which is ip6_dst_destroy.
>>>>>
>>>>> The problem here is we have the workq blocked because we are running
>>>>> inside the netns cleanup function. So the delayed work will not run
>>>>> until we exits the cleanup function. But the loopback is still
>>>>> referenced by the ip6 routes, the netdev_unregister will loop
>>>>> indefinitly => dead lock.
>>>>>
>>>>> By the way, this bug appears with ipv6 but it is perhaps pending with
>>>>> ipv4.
>>>>>
>>>>> Benjamin as proposed to create a separate workq for the network
>>>>> namespace, so in the worst case we have the unregister looping until the

>>>>> ip6 route are shut downed. Is it an acceptable solution ?
>>>>>
>>>> we are doing this staff in the special thread. There are a lot of
>>>> difficult things to perform like synchronize_net & netdev_run_todo inside
>>> The special thread ? do you mean keventd_wq ?
>>>
>> I mean that network namespace deletion, i.e. all subsystem ->exit calls
>> should be run outside of all current mechanisms in the separate thread,
>> specially designated to namespace(s) stop.
>
> Interesting.
> How do you create the thread? Do you use a special workqueue to replace the
> use of the global keventd workqueue, as I proposed, or do you use another
> mechanism to create the thread?
> I mean do you create one thread per exiting namespace (each time a namespace
> is exiting you spawn a new thread for the cleanup) or do you create a workqueue
> at system init where you'll queue all cleanup routines (cleanup_net) for all
> exiting namespaces?
>
> Currently, on our side, we have a small patch that creates a special
> workqueue in net_ns_init(), and we queue clean_net() in this workqueue
> in __put_net().

I think 1 thread in the system is enough. It should accept queued
requests for namespace cleanup. so, this looks pretty same as you do..