Subject: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by Pavel Emelianov on Wed, 31 Oct 2007 18:17:47 GMT
View Forum Message <> Reply to Message

Currently we have the NET_NS config option, but the only change it
makes is just return ERR_PTR(-EINVAL) inside the cloning call thus
introducing a bunch of a dead code and making the reference counting
unneeded. This is not very good.

So clean the net_namespace.c to fix this.

I have sent a set of patches to Andrew to make similar thing for
other namespaces, which introduces the NAMESPACES option to turn
all the namespaces off at once (to make embedded people suffer
less). So after that stuff is in, there will be some more patches
to tie all this together.

What is to be done after this set is to make the register_pernet_xxx
stuff smaller. Currently this code weights approximately 500 bytes,
so it worths reducing it, but I haven't found a good solution yet.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Subject: [PATCH 1/5][NETNS] Make the init/exit hooks checks outside the loop
Posted by Pavel Emelianov on Wed, 31 Oct 2007 18:21:34 GMT
View Forum Message <> Reply to Message

When the new pernet something (subsys, device or operations) is
being registered, the init callback is to be called for each
namespace, that currently exitst in the system. During the
unregister, the same is to be done with the exit callback.

However, not every pernet something has both calls, but the
check for the appropriate pointer to be not NULL is performed
inside the for_each_net() loop.

This is (at least) strange, so tune this.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---

diff --git a/net/core/net_namespace.c b/net/core/net_namespace.c
index 662e6ea..4e52921 100644
--- a/net/core/net_namespace.c
+++ b/net/core/net_namespace.c
@@ -187,29 +187,28 @@ static int register_pernet_operations(struct list_head *list,

```
  struct net *net, *undo_net;
  int error;

- error = 0;
  list_add_tail(&ops->list, list);
- for_each_net(net) {
-  if (ops->init) {
+ if (ops->init) {
+  for_each_net(net) {
     error = ops->init(net);
     if (error)
      goto out_undo;
   }
  }
-out:
- return error;
+ return 0;

 out_undo:
  /* If I have an error cleanup all namespaces I initialized */
  list_del(&ops->list);
- for_each_net(undo_net) {
-  if (undo_net == net)
-   goto undone;
-  if (ops->exit)
+ if (ops->exit) {
+  for_each_net(undo_net) {
+   if (undo_net == net)
+    goto undone;
     ops->exit(undo_net);
+  }
  }
 undone:
- goto out;
+ return error;
 }

 static void unregister_pernet_operations(struct pernet_operations *ops)
@@ -217,8 +216,8 @@ static void unregister_pernet_operations(struct pernet_operations *ops)
  struct net *net;

  list_del(&ops->list);
- for_each_net(net)
-  if (ops->exit)
+ if (ops->exit)
+  for_each_net(net)
     ops->exit(net);
 }
```

--
1.5.3.4

Subject: [PATCH 2/5] Relax the reference counting of init_net_ns
Posted by Pavel Emelianov on Wed, 31 Oct 2007 18:23:25 GMT
View Forum Message <> Reply to Message

When the CONFIG_NET_NS is n there's no need in refcounting
the initial net namespace. So relax this code by making a
stupid stubs for the "n" case.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/include/net/net_namespace.h b/include/net/net_namespace.h
index 5279466..1fd449a 100644
--- a/include/net/net_namespace.h
+++ b/include/net/net_namespace.h
@@ -51,13 +51,12 @@ static inline struct net *copy_net_ns(unsigned long flags, struct net
*net_ns)
 }
 #endif

+#ifdef CONFIG_NET_NS
 extern void __put_net(struct net *net);

 static inline struct net *get_net(struct net *net)
 {
-#ifdef CONFIG_NET
  atomic_inc(&net->count);
-#endif
  return net;
 }

@@ -75,26 +74,44 @@ static inline struct net *maybe_get_net(struct net *net)

 static inline void put_net(struct net *net)
 {
-#ifdef CONFIG_NET
  if (atomic_dec_and_test(&net->count))
   __put_net(net);
-#endif
 }

 static inline struct net *hold_net(struct net *net)
```

```
 {
-#ifdef CONFIG_NET
  atomic_inc(&net->use_count);
-#endif
  return net;
 }

 static inline void release_net(struct net *net)
 {
-#ifdef CONFIG_NET
  atomic_dec(&net->use_count);
-#endif
 }
+#else
+static inline struct net *get_net(struct net *net)
+{
+ return net;
+}
+
+static inline void put_net(struct net *net)
+{
+}
+
+static inline struct net *hold_net(struct net *net)
+{
+ return net;
+}
+
+static inline void release_net(struct net *net)
+{
+}
+
+static inline struct net *maybe_get_net(struct net *net)
+{
+ return net;
+}
+#endif

 #define for_each_net(VAR)    \
  list_for_each_entry(VAR, &net_namespace_list, list)
--
1.5.3.4
```

Subject: [PATCH 3/5] Hide the dead code in the net_namespace.c
Posted by Pavel Emelianov on Wed, 31 Oct 2007 18:27:00 GMT
View Forum Message <> Reply to Message

The namespace creation/destruction code is never called
if the CONFIG_NET_NS is n, so it's OK to move it under
appropriate ifdef.

The copy_net_ns() in the "n" case checks for flags and
returns -EINVAL when new net ns is requested. In a perfect
world this stub must be in net_namespace.h, but this
function need to know the CLONE_NEWNET value and thus
requires sched.h. On the other hand this header is to be
injected into almost every .c file in the networking code,
and making all this code depend on the sched.h is a
suicidal attempt.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/net/core/net_namespace.c b/net/core/net_namespace.c
index 4e52921..d5bf8b2 100644
--- a/net/core/net_namespace.c
+++ b/net/core/net_namespace.c
@@ -22,65 +22,6 @@ static struct kmem_cache *net_cachep;
 struct net init_net;
 EXPORT_SYMBOL_GPL(init_net);

-static struct net *net_alloc(void)
-{
- return kmem_cache_zalloc(net_cachep, GFP_KERNEL);
-}
-
-static void net_free(struct net *net)
-{
- if (!net)
-  return;
-
- if (unlikely(atomic_read(&net->use_count) != 0)) {
-  printk(KERN_EMERG "network namespace not free! Usage: %d\n",
-    atomic_read(&net->use_count));
-  return;
- }
-
- kmem_cache_free(net_cachep, net);
-}
-
-static void cleanup_net(struct work_struct *work)
-{
- struct pernet_operations *ops;
- struct net *net;
```

```
-
- net = container_of(work, struct net, work);
-
- mutex_lock(&net_mutex);
-
- /* Don't let anyone else find us. */
- rtnl_lock();
- list_del(&net->list);
- rtnl_unlock();
-
- /* Run all of the network namespace exit methods */
- list_for_each_entry_reverse(ops, &pernet_list, list) {
-  if (ops->exit)
-   ops->exit(net);
- }
-
- mutex_unlock(&net_mutex);
-
- /* Ensure there are no outstanding rcu callbacks using this
-  * network namespace.
-  */
- rcu_barrier();
-
- /* Finally it is safe to free my network namespace structure */
- net_free(net);
-}
-
-
-void __put_net(struct net *net)
-{
- /* Cleanup the network namespace in process context */
- INIT_WORK(&net->work, cleanup_net);
- schedule_work(&net->work);
-}
-EXPORT_SYMBOL_GPL(__put_net);
-
 /*
  * setup_net runs the initializers for the network namespace object.
  */
@@ -117,6 +58,12 @@ out_undo:
  goto out;
 }

+#ifdef CONFIG_NET_NS
+static struct net *net_alloc(void)
+{
+ return kmem_cache_zalloc(net_cachep, GFP_KERNEL);
+}
```

```
+
 struct net *copy_net_ns(unsigned long flags, struct net *old_net)
 {
  struct net *new_net = NULL;
@@ -127,10 +74,6 @@ struct net *copy_net_ns(unsigned long flags, struct net *old_net)
  if (!(flags & CLONE_NEWNET))
   return old_net;

-#ifndef CONFIG_NET_NS
- return ERR_PTR(-EINVAL);
-#endif
-
  err = -ENOMEM;
  new_net = net_alloc();
  if (!new_net)
@@ -157,6 +100,68 @@ out:
  return new_net;
 }

+static void net_free(struct net *net)
+{
+ if (!net)
+  return;
+
+ if (unlikely(atomic_read(&net->use_count) != 0)) {
+  printk(KERN_EMERG "network namespace not free! Usage: %d\n",
+    atomic_read(&net->use_count));
+  return;
+ }
+
+ kmem_cache_free(net_cachep, net);
+}
+
+static void cleanup_net(struct work_struct *work)
+{
+ struct pernet_operations *ops;
+ struct net *net;
+
+ net = container_of(work, struct net, work);
+
+ mutex_lock(&net_mutex);
+
+ /* Don't let anyone else find us. */
+ rtnl_lock();
+ list_del(&net->list);
+ rtnl_unlock();
+
+ /* Run all of the network namespace exit methods */
```

```
+ list_for_each_entry_reverse(ops, &pernet_list, list) {
+  if (ops->exit)
+   ops->exit(net);
+ }
+
+ mutex_unlock(&net_mutex);
+
+ /* Ensure there are no outstanding rcu callbacks using this
+  * network namespace.
+  */
+ rcu_barrier();
+
+ /* Finally it is safe to free my network namespace structure */
+ net_free(net);
+}
+
+void __put_net(struct net *net)
+{
+ /* Cleanup the network namespace in process context */
+ INIT_WORK(&net->work, cleanup_net);
+ schedule_work(&net->work);
+}
+EXPORT_SYMBOL_GPL(__put_net);
+
+#else
+struct net *copy_net_ns(unsigned long flags, struct net *old_net)
+{
+ if (flags & CLONE_NEWNET)
+  return ERR_PTR(-EINVAL);
+ return old_net;
+}
+#endif
+
 static int __init net_ns_init(void)
 {
  int err;
--
1.5.3.4
```

Subject: [PATCH 4/5] Mark the setup_net as __net_init
Posted by Pavel Emelianov on Wed, 31 Oct 2007 18:29:45 GMT
View Forum Message <> Reply to Message

The setup_net is called for the init net namespace
only (int the CONFIG_NET_NS=n of course) from the __init
function, so mark it as __net_init to disappear with the
caller after the boot.

Yet again, in the perfect world this has to be under
#ifdef CONFIG_NET_NS, but it isn't guaranteed that every
subsystem is registered *after* the init_net_ns is set
up. After we are sure, that we don't start registering
them before the init net setup, we'll be able to move
this code under the ifdef.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/net/core/net_namespace.c b/net/core/net_namespace.c
index d5bf8b2..a044e2d 100644
--- a/net/core/net_namespace.c
+++ b/net/core/net_namespace.c
@@ -25,7 +25,7 @@ EXPORT_SYMBOL_GPL(init_net);
 /*
  * setup_net runs the initializers for the network namespace object.
  */
-static int setup_net(struct net *net)
+static __net_init int setup_net(struct net *net)
 {
  /* Must be called with net_mutex held */
  struct pernet_operations *ops;
--
1.5.3.4
```

Subject: [PATCH 5/5] Hide the net_ns kmem cache
Posted by Pavel Emelianov on Wed, 31 Oct 2007 18:30:53 GMT
View Forum Message <> Reply to Message

This cache is only required to create new namespaces,
but we won't have them in CONFIG_NET_NS=n case.

Hide it under the appropriate ifdef.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

```
diff --git a/net/core/net_namespace.c b/net/core/net_namespace.c
index a044e2d..e9f0964 100644
--- a/net/core/net_namespace.c
+++ b/net/core/net_namespace.c
@@ -17,8 +17,6 @@ static DEFINE_MUTEX(net_mutex);
```

```
 LIST_HEAD(net_namespace_list);

-static struct kmem_cache *net_cachep;
-
 struct net init_net;
 EXPORT_SYMBOL_GPL(init_net);

@@ -59,6 +57,8 @@ out_undo:
 }

 #ifdef CONFIG_NET_NS
+static struct kmem_cache *net_cachep;
+
 static struct net *net_alloc(void)
 {
  return kmem_cache_zalloc(net_cachep, GFP_KERNEL);
@@ -167,9 +167,11 @@ static int __init net_ns_init(void)
  int err;

  printk(KERN_INFO "net_namespace: %zd bytes\n", sizeof(struct net));
+#ifdef CONFIG_NET_NS
  net_cachep = kmem_cache_create("net_namespace", sizeof(struct net),
      SMP_CACHE_BYTES,
      SLAB_PANIC, NULL);
+#endif
  mutex_lock(&net_mutex);
  err = setup_net(&init_net);

--
1.5.3.4
```

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by Eric Dumazet on Wed, 31 Oct 2007 18:49:24 GMT
View Forum Message <> Reply to Message

On Wed, 31 Oct 2007 22:19:43 +0300
Pavel Emelyanov <xemul@openvz.org> wrote:

> Currently we have the NET_NS config option, but the only change it
> makes is just return ERR_PTR(-EINVAL) inside the cloning call thus
> introducing a bunch of a dead code and making the reference counting
> unneeded. This is not very good.
>
> So clean the net_namespace.c to fix this.
>
> I have sent a set of patches to Andrew to make similar thing for
> other namespaces, which introduces the NAMESPACES option to turn

> all the namespaces off at once (to make embedded people suffer
> less). So after that stuff is in, there will be some more patches
> to tie all this together.
>
> What is to be done after this set is to make the register_pernet_xxx
> stuff smaller. Currently this code weights approximately 500 bytes,
> so it worths reducing it, but I haven't found a good solution yet.

Definitly wanted here. Thank you.
One more refcounting on each socket creation/deletion was expensive.

Maybe we can add a macro to get nd_net from a "struct net_device"
so that every instance of

```
if (dev->nd_net != &init_net)
    goto drop;
```

can also be optimized away if !CONFIG_NET_NS

```
extern inline netdev_get_ns(struct netdevice *dev)
{
#ifdef CONFIG_NET_NS
 return dev->nd_net;
#else
 return &init_net;
#endif
}

...

if (netdev_get_ns(dev) != &init_net)
 goto drop;
```

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by Daniel Lezcano on Wed, 31 Oct 2007 21:35:51 GMT
View Forum Message <> Reply to Message

Eric Dumazet wrote:
> On Wed, 31 Oct 2007 22:19:43 +0300
> Pavel Emelyanov <xemul@openvz.org> wrote:
>
>> Currently we have the NET_NS config option, but the only change it
>> makes is just return ERR_PTR(-EINVAL) inside the cloning call thus
>> introducing a bunch of a dead code and making the reference counting
>> unneeded. This is not very good.
>>
>> So clean the net_namespace.c to fix this.

>>
>> I have sent a set of patches to Andrew to make similar thing for
>> other namespaces, which introduces the NAMESPACES option to turn
>> all the namespaces off at once (to make embedded people suffer
>> less). So after that stuff is in, there will be some more patches
>> to tie all this together.
>>
>> What is to be done after this set is to make the register_pernet_xxx
>> stuff smaller. Currently this code weights approximately 500 bytes,
>> so it worths reducing it, but I haven't found a good solution yet.
>
> Definitly wanted here. Thank you.
> One more refcounting on each socket creation/deletion was expensive.
>
> Maybe we can add a macro to get nd_net from a "struct net_device"
> so that every instance of
>
> if (dev->nd_net != &init_net)
>   goto drop;
>
> can also be optimized away if !CONFIG_NET_NS
>
> extern inline netdev_get_ns(struct netdevice *dev)
> {
> #ifdef CONFIG_NET_NS
> return dev->nd_net;
> #else
> return &init_net;
> #endif
> }

Or something like:

```
#ifdef CONFIG_NET_NS
static inline int init_net_dev(struct net_device *dev)
{
 return dev->nd_net == &init_net;
}
#else
static inline int init_net_dev(struct net_device *dev)
{
 return 1;
}
#endif
```

By the way, this kind of test will disappear when the network namespace
will be complete and take into account the differents protocols.

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by Daniel Lezcano on Wed, 31 Oct 2007 21:37:11 GMT
View Forum Message <> Reply to Message

Pavel Emelyanov wrote:
> Currently we have the NET_NS config option, but the only change it
> makes is just return ERR_PTR(-EINVAL) inside the cloning call thus
> introducing a bunch of a dead code and making the reference counting
> unneeded. This is not very good.
>
> So clean the net_namespace.c to fix this.
>
> I have sent a set of patches to Andrew to make similar thing for
> other namespaces, which introduces the NAMESPACES option to turn
> all the namespaces off at once (to make embedded people suffer
> less). So after that stuff is in, there will be some more patches
> to tie all this together.
>
> What is to be done after this set is to make the register_pernet_xxx
> stuff smaller. Currently this code weights approximately 500 bytes,
> so it worths reducing it, but I haven't found a good solution yet.

Did you had time to check the impact of your patch with the rest of the network namespaces not yet included in mainline, belonging to Eric's git tree ?

ps: can you cc' emails concerning the network namespace to the containers mailing list too ? thx.

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by ebiederm on Wed, 31 Oct 2007 22:05:46 GMT
View Forum Message <> Reply to Message

Eric Dumazet <dada1@cosmosbay.com> writes:

> Definitly wanted here. Thank you.
> One more refcounting on each socket creation/deletion was expensive.

Really? Have you actually measured that? If the overhead is measurable and expensive we may want to look at per cpu counters or something like that. So far I don't have any numbers that say any of the network namespace work inherently has any overhead.

> Maybe we can add a macro to get nd_net from a "struct net_device"
> so that every instance of
>

> if (dev->nd_net != &init_net)
>   goto drop;
>
> can also be optimized away if !CONFIG_NET_NS

Well that extra check should be removed once we finish converting
those code paths. So I'm not too worried.

If this becomes a big issue I can dig up my old code that
replaced struct net * with a net_t typedef and used functions
for all of the comparisons and allowed everything to be compiled
away.

Trouble was it was sufficiently different that it was just enough
different that people could not immediately understand the code.

Eric

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by Eric Dumazet on Wed, 31 Oct 2007 22:40:59 GMT
View Forum Message <> Reply to Message

> Eric Dumazet <dada1@cosmosbay.com> writes:
>
>
>> Definitly wanted here. Thank you.
>> One more refcounting on each socket creation/deletion was expensive.
>
> Really? Have you actually measured that? If the overhead is
> measurable and expensive we may want to look at per cpu counters or
> something like that. So far I don't have any numbers that say any
> of the network namespace work inherently has any overhead.

It seems that on some old opterons (two 246 for example),
"if (atomic_dec_and_test(&net->count))" is rather expensive yes :(

I am not sure per cpu counters help : I tried this and got no speedup. (This
was on net_device refcnt at that time)

(on this machines, the access through fs/gs selector seems expensive too)

Maybe a lazy mode could be done, ie only do a atomic_dec(), as done in dev_put() ?

Also, "count" sits in a cache line that contains mostly read and shared
fields, you might want to put it in a separate cache line in SMP, to avoid
cache line ping-pongs.

>
>> Maybe we can add a macro to get nd_net from a "struct net_device"
>> so that every instance of
>>
>> if (dev->nd_net != &init_net)
>>   goto drop;
>>
>> can also be optimized away if !CONFIG_NET_NS
>
> Well that extra check should be removed once we finish converting
> those code paths. So I'm not too worried.

OK. Since the conditional test can be predicted by cpu, it certainly doesnt
matter.

>
> If this becomes a big issue I can dig up my old code that
> replaced struct net * with a net_t typedef and used functions
> for all of the comparisons and allowed everything to be compiled
> away.

>
> Trouble was it was sufficiently different that it was just enough
> different that people could not immediately understand the code.
>

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by davem on Wed, 31 Oct 2007 23:31:46 GMT
View Forum Message <> Reply to Message

From: Eric Dumazet <dada1@cosmosbay.com>
Date: Wed, 31 Oct 2007 23:40:59 +0100

> > Eric Dumazet <dada1@cosmosbay.com> writes:
> >
> >
> >> Definitly wanted here. Thank you.
> >> One more refcounting on each socket creation/deletion was expensive.
> >
> > Really? Have you actually measured that? If the overhead is
> > measurable and expensive we may want to look at per cpu counters or
> > something like that. So far I don't have any numbers that say any

> > of the network namespace work inherently has any overhead.
>
> It seems that on some old opterons (two 246 for example),
> "if (atomic_dec_and_test(&net->count))" is rather expensive yes :(

P4 chips are generally very poor at mispredicted branches and
atomics. So every atomic you remove from the socket paths
gives a noticable improvement on them.

Network device reference counting is such a stupid problem. There has
to be a way to get rid of it on the packet side.

I think we could get rid of all of the device refcounting from packets
if we:

1) Formalize "SKB roots". This is every place a packet
could sit in the transmit path.

2) On device unregister:

a) wait for RCU quiesce period
b) stop_machine_run(skb_walk_roots, netdev, NR_CPUS);

skb_walk_roots is a function that walks all the places in
#1, rewriting the packet to point to loopback or whatever
instead of 'netdev' which we are trying to unregister.

This gives us two things.

First, we no longer would need to rectount net devices
for packet references.

Second, we have a debugging framework for all those dreaded SKB leaks
that keep devices from being unloadable. As we walk the roots
we'll see where all packets referencing a device actually are.

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by ebiederm on Thu, 01 Nov 2007 00:51:54 GMT
View Forum Message <> Reply to Message

Eric Dumazet <dada1@cosmosbay.com> writes:

>> Eric Dumazet <dada1@cosmosbay.com> writes:
>>
>>
>>> Definitly wanted here. Thank you.

>>> One more refcounting on each socket creation/deletion was expensive.
>>
>> Really? Have you actually measured that? If the overhead is
>> measurable and expensive we may want to look at per cpu counters or
>> something like that. So far I don't have any numbers that say any
>> of the network namespace work inherently has any overhead.
>
> It seems that on some old opterons (two 246 for example),
> "if (atomic_dec_and_test(&net->count))" is rather expensive yes :(

I won't argue that atomic_dec_and_test is costly. My gut feel is that
socket creation/destruction is sufficiently rare that such a test
would be lost in the noise. Doing anything more sophisticated is
likely to be less readable, and unless we can measure some overhead
my preference right now is to keep the code stupid and simple. Which
usually has a good icache footprint.

> I am not sure per cpu counters help : I tried this and got no speedup. (This was
> on net_device refcnt at that time)
>
> (on this machines, the access through fs/gs selector seems expensive too)
>
> Maybe a lazy mode could be done, ie only do a atomic_dec(), as done in dev_put()
> ?
>
> Also, "count" sits in a cache line that contains mostly read and shared fields,
> you might want to put it in a separate cache line in SMP, to avoid cache line
> ping-pongs.

As for cache lines I could reverse the order 'list' and 'work' which
should split the read-only and the writable fields in practice for
that part of the structure.

Eric

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by ebiederm on Thu, 01 Nov 2007 00:58:28 GMT
View Forum Message <> Reply to Message

David Miller <davem@davemloft.net> writes:

> From: Eric Dumazet <dada1@cosmosbay.com>
> Date: Wed, 31 Oct 2007 23:40:59 +0100
>

>> > Eric Dumazet <dada1@cosmosbay.com> writes:
>> >

>> >
>> >> Definitly wanted here. Thank you.
>> >> One more refcounting on each socket creation/deletion was expensive.
>> >
>> > Really? Have you actually measured that? If the overhead is
>> > measurable and expensive we may want to look at per cpu counters or
>> > something like that. So far I don't have any numbers that say any
>> > of the network namespace work inherently has any overhead.
>>
>> It seems that on some old opterons (two 246 for example),
>> "if (atomic_dec_and_test(&net->count))" is rather expensive yes :(
>
> P4 chips are generally very poor at mispredicted branches and
> atomics. So every atomic you remove from the socket paths
> gives a noticable improvement on them.

Interesting.

> Network device reference counting is such a stupid problem. There has
> to be a way to get rid of it on the packet side.
>
> I think we could get rid of all of the device refcounting from packets
> if we:
>
> 1) Formalize "SKB roots". This is every place a packet
>   could sit in the transmit path.

Yes there are very few of these, and I think they are generally
in interrupt or at least bottom half context aren't they?

I think the OpenVz version of network namespaces may have
already identified all of these.

> 2) On device unregister:
>
> a) wait for RCU quiesce period
> b) stop_machine_run(skb_walk_roots, netdev, NR_CPUS);

RCU sounds sufficient but possibly overkill to achieve what
we need to do here.

>   skb_walk_roots is a function that walks all the places in
>   #1, rewriting the packet to point to loopback or whatever
>   instead of 'netdev' which we are trying to unregister.
>
> This gives us two things.
>
> First, we no longer would need to rectount net devices

> for packet references.
>
> Second, we have a debugging framework for all those dreaded SKB leaks
> that keep devices from being unloadable. As we walk the roots
> we'll see where all packets referencing a device actually are.

Sounds quite useful. Grrr. The brain cache locality that gets us
to rewrite things while we are refactoring them to have more
functionality.... It just keeps the problem from being straigh
forward ;)

Eric

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by Eric Dumazet on Thu, 01 Nov 2007 06:58:59 GMT
View Forum Message <> Reply to Message

> Eric Dumazet <dada1@cosmosbay.com> writes:
>

>>> Eric Dumazet <dada1@cosmosbay.com> writes:
>>>
>>>
>>>> Definitly wanted here. Thank you.
>>>> One more refcounting on each socket creation/deletion was expensive.
>>> Really? Have you actually measured that? If the overhead is
>>> measurable and expensive we may want to look at per cpu counters or
>>> something like that. So far I don't have any numbers that say any
>>> of the network namespace work inherently has any overhead.
>> It seems that on some old opterons (two 246 for example),
>> "if (atomic_dec_and_test(&net->count))" is rather expensive yes :(
>
> I won't argue that atomic_dec_and_test is costly. My gut feel is that
> socket creation/destruction is sufficiently rare that such a test
> would be lost in the noise. Doing anything more sophisticated is
> likely to be less readable, and unless we can measure some overhead
> my preference right now is to keep the code stupid and simple. Which
> usually has a good icache footprint.

I agree with you that with current state, this atomic_inc/atomic_dec_and_test
wont come in profiles unless a trivial bench is writen

for(;;){close(socket(....));}

If David or another dev can eliminate the atomic inc/dec on device refcount
cost for each packet traveling, the socket creation/destruction would

certainly raise.

Other contention points is the mnt_count (yet another refcount) in "struct
vfsmount", a truly useless refcount as I never had (and nobody had) to
un-mount sock_mnt :)

Subject: Re: [PATCH 0/5] Make nicer CONFIG_NET_NS=n case code
Posted by davem on Thu, 01 Nov 2007 07:02:38 GMT
View Forum Message <> Reply to Message

From: Eric Dumazet <dada1@cosmosbay.com>
Date: Thu, 01 Nov 2007 07:58:59 +0100

> I agree with you that with current state, this atomic_inc/atomic_dec_and_test
> wont come in profiles unless a trivial bench is writen
>
> for(;;){close(socket(....));}

Just add one packet send and one packet receive in there and you have
a transaction workload. It's really not that unrealistic.

Subject: Re: [PATCH 1/5][NETNS] Make the init/exit hooks checks outside the loop
Posted by davem on Thu, 01 Nov 2007 07:43:07 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Wed, 31 Oct 2007 22:23:35 +0300

> When the new pernet something (subsys, device or operations) is
> being registered, the init callback is to be called for each
> namespace, that currently exitst in the system. During the
> unregister, the same is to be done with the exit callback.
>
> However, not every pernet something has both calls, but the
> check for the appropriate pointer to be not NULL is performed
> inside the for_each_net() loop.
>
> This is (at least) strange, so tune this.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH 2/5] Relax the reference counting of init_net_ns
Posted by davem on Thu, 01 Nov 2007 07:43:58 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Wed, 31 Oct 2007 22:25:18 +0300

> When the CONFIG_NET_NS is n there's no need in refcounting
> the initial net namespace. So relax this code by making a
> stupid stubs for the "n" case.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH 3/5] Hide the dead code in the net_namespace.c
Posted by davem on Thu, 01 Nov 2007 07:45:06 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Wed, 31 Oct 2007 22:28:51 +0300

> The namespace creation/destruction code is never called
> if the CONFIG_NET_NS is n, so it's OK to move it under
> appropriate ifdef.
>
> The copy_net_ns() in the "n" case checks for flags and
> returns -EINVAL when new net ns is requested. In a perfect
> world this stub must be in net_namespace.h, but this
> function need to know the CLONE_NEWNET value and thus
> requires sched.h. On the other hand this header is to be
> injected into almost every .c file in the networking code,
> and making all this code depend on the sched.h is a
> suicidal attempt.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH 4/5] Mark the setup_net as __net_init
Posted by davem on Thu, 01 Nov 2007 07:46:16 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Wed, 31 Oct 2007 22:31:46 +0300

> The setup_net is called for the init net namespace
> only (int the CONFIG_NET_NS=n of course) from the __init
> function, so mark it as __net_init to disappear with the
> caller after the boot.
>
> Yet again, in the perfect world this has to be under
> #ifdef CONFIG_NET_NS, but it isn't guaranteed that every
> subsystem is registered *after* the init_net_ns is set
> up. After we are sure, that we don't start registering
> them before the init net setup, we'll be able to move
> this code under the ifdef.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.

Subject: Re: [PATCH 5/5] Hide the net_ns kmem cache
Posted by davem on Thu, 01 Nov 2007 07:47:05 GMT
View Forum Message <> Reply to Message

From: Pavel Emelyanov <xemul@openvz.org>
Date: Wed, 31 Oct 2007 22:32:55 +0300

> This cache is only required to create new namespaces,
> but we won't have them in CONFIG_NET_NS=n case.
>
> Hide it under the appropriate ifdef.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

Applied.
