Subject: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by Stanislav Kinsbursky on Thu, 27 Oct 2011 18:11:28 GMT
View Forum Message <> Reply to Message

v3:
1) First two patches from previous version were squashed.

This patch-set was created in context of clone of git branch:
git://git.linux-nfs.org/projects/trondmy/nfs-2.6.git
and rebased on tag "v3.1".

This patch-set virtualizes rpcbind clients per network namespace context. IOW,
each network namespace will have its own pair of rpcbind clients (if they would
be created by request).

Note:
1) this patch-set depends on "SUNRPC: make rpcbind clients allocated and
destroyed dynamically" patch-set which has been send earlier.
2) init_net pointer is still used instead of current->nsproxy->net_ns,
because I'm not sure yet about how to virtualize services. I.e. NFS callback
services will be per netns. NFSd service will be per netns too from my pow. But
Lockd can be per netns or one for all. And also we have NFSd file system, which
is not virtualized yet.

The following series consists of:

Stanislav Kinsbursky (3):
    SUNRPC: move rpcbind internals to sunrpc part of network namespace context
    SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
    SUNRPC: optimize net_ns dereferencing in rpcbind registering calls

 net/sunrpc/netns.h | 5 ++
 net/sunrpc/rpcb_clnt.c | 103 ++++++++++++++++++++++++++++++++----------------
 2 files changed, 61 insertions(+), 47 deletions(-)

--
Signature

Subject: [PATCH v3 1/3] SUNRPC: move rpcbind internals to sunrpc part of network namespace context
Posted by Stanislav Kinsbursky on Thu, 27 Oct 2011 18:11:39 GMT
View Forum Message <> Reply to Message

This patch makes rpcbind logic works in network namespace context. IOW each

network namespace will have it's own unique rpcbind internals (clients and
friends) which is required for registering svc services per network namespace.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

---
 net/sunrpc/netns.h | 5 ++++
 net/sunrpc/rpcb_clnt.c | 64 ++++++++++++++++++++++++++++++----------------
 2 files changed, 40 insertions(+), 29 deletions(-)

diff --git a/net/sunrpc/netns.h b/net/sunrpc/netns.h
index d013bf2..83eede3 100644
--- a/net/sunrpc/netns.h
+++ b/net/sunrpc/netns.h
@@ -9,6 +9,11 @@ struct cache_detail;
 struct sunrpc_net {
  struct proc_dir_entry *proc_net_rpc;
  struct cache_detail *ip_map_cache;
+
+ struct rpc_clnt *rpcb_local_clnt;
+ struct rpc_clnt *rpcb_local_clnt4;
+ spinlock_t rpcb_clnt_lock;
+ unsigned int rpcb_users;
 };

 extern int sunrpc_net_id;
diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index 983b74f..59bd1fb 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -23,12 +23,15 @@
 #include <linux/errno.h>
 #include <linux/mutex.h>
 #include <linux/slab.h>
+#include <linux/nsproxy.h>
 #include <net/ipv6.h>

 #include <linux/sunrpc/clnt.h>
 #include <linux/sunrpc/sched.h>
 #include <linux/sunrpc/xprtsock.h>

+#include "netns.h"
+
 #ifdef RPC_DEBUG
 # define RPCDBG_FACILITY RPCDBG_BIND
 #endif
@@ -111,12 +114,6 @@ static void   rpcb_getport_done(struct rpc_task *, void *);
 static void   rpcb_map_release(void *data);

```
 static struct rpc_program rpcb_program;

-static struct rpc_clnt * rpcb_local_clnt;
-static struct rpc_clnt * rpcb_local_clnt4;
-
-DEFINE_SPINLOCK(rpcb_clnt_lock);
-unsigned int   rpcb_users;
-
 struct rpcbind_args {
  struct rpc_xprt * r_xprt;

@@ -167,29 +164,31 @@ static void rpcb_map_release(void *data)
 static int rpcb_get_local(void)
 {
  int cnt;
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

- spin_lock(&rpcb_clnt_lock);
- if (rpcb_users)
-  rpcb_users++;
- cnt = rpcb_users;
- spin_unlock(&rpcb_clnt_lock);
+ spin_lock(&sn->rpcb_clnt_lock);
+ if (sn->rpcb_users)
+  sn->rpcb_users++;
+ cnt = sn->rpcb_users;
+ spin_unlock(&sn->rpcb_clnt_lock);

  return cnt;
 }

 void rpcb_put_local(void)
 {
- struct rpc_clnt *clnt = rpcb_local_clnt;
- struct rpc_clnt *clnt4 = rpcb_local_clnt4;
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);
+ struct rpc_clnt *clnt = sn->rpcb_local_clnt;
+ struct rpc_clnt *clnt4 = sn->rpcb_local_clnt4;
  int shutdown;

- spin_lock(&rpcb_clnt_lock);
- if (--rpcb_users == 0) {
-  rpcb_local_clnt = NULL;
-  rpcb_local_clnt4 = NULL;
+ spin_lock(&sn->rpcb_clnt_lock);
+ if (--sn->rpcb_users == 0) {
+  sn->rpcb_local_clnt = NULL;
+  sn->rpcb_local_clnt4 = NULL;
```

```
  }
- shutdown = !rpcb_users;
- spin_unlock(&rpcb_clnt_lock);
+ shutdown = !sn->rpcb_users;
+ spin_unlock(&sn->rpcb_clnt_lock);

  if (shutdown) {
   /*
@@ -204,14 +203,16 @@ void rpcb_put_local(void)

 static void rpcb_set_local(struct rpc_clnt *clnt, struct rpc_clnt *clnt4)
 {
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);
+
  /* Protected by rpcb_create_local_mutex */
- rpcb_local_clnt = clnt;
- rpcb_local_clnt4 = clnt4;
+ sn->rpcb_local_clnt = clnt;
+ sn->rpcb_local_clnt4 = clnt4;
  smp_wmb();
- rpcb_users = 1;
+ sn->rpcb_users = 1;
  dprintk("RPC:       created new rpcb local clients (rpcb_local_clnt: "
-   "%p, rpcb_local_clnt4: %p)\n", rpcb_local_clnt,
-   rpcb_local_clnt4);
+   "%p, rpcb_local_clnt4: %p)\n", sn->rpcb_local_clnt,
+   sn->rpcb_local_clnt4);
 }

 /*
@@ -431,6 +432,7 @@ int rpcb_register(u32 prog, u32 vers, int prot, unsigned short port)
  struct rpc_message msg = {
   .rpc_argp = &map,
  };
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  dprintk("RPC:       %sregistering (%u, %u, %d, %u) with local "
    "rpcbind\n", (port ? "" : "un"),
@@ -440,7 +442,7 @@ int rpcb_register(u32 prog, u32 vers, int prot, unsigned short port)
  if (port)
   msg.rpc_proc = &rpcb_procedures2[RPCBPROC_SET];

- return rpcb_register_call(rpcb_local_clnt, &msg);
+ return rpcb_register_call(sn->rpcb_local_clnt, &msg);
 }

 /*
@@ -453,6 +455,7 @@ static int rpcb_register_inet4(const struct sockaddr *sap,
```

```
  struct rpcbind_args *map = msg->rpc_argp;
  unsigned short port = ntohs(sin->sin_port);
  int result;
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  map->r_addr = rpc_sockaddr2uaddr(sap);

@@ -465,7 +468,7 @@ static int rpcb_register_inet4(const struct sockaddr *sap,
  if (port)
   msg->rpc_proc = &rpcb_procedures4[RPCBPROC_SET];

- result = rpcb_register_call(rpcb_local_clnt4, msg);
+ result = rpcb_register_call(sn->rpcb_local_clnt4, msg);
  kfree(map->r_addr);
  return result;
 }
@@ -480,6 +483,7 @@ static int rpcb_register_inet6(const struct sockaddr *sap,
  struct rpcbind_args *map = msg->rpc_argp;
  unsigned short port = ntohs(sin6->sin6_port);
  int result;
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  map->r_addr = rpc_sockaddr2uaddr(sap);

@@ -492,7 +496,7 @@ static int rpcb_register_inet6(const struct sockaddr *sap,
  if (port)
   msg->rpc_proc = &rpcb_procedures4[RPCBPROC_SET];

- result = rpcb_register_call(rpcb_local_clnt4, msg);
+ result = rpcb_register_call(sn->rpcb_local_clnt4, msg);
  kfree(map->r_addr);
  return result;
 }
@@ -500,6 +504,7 @@ static int rpcb_register_inet6(const struct sockaddr *sap,
 static int rpcb_unregister_all_protofamilies(struct rpc_message *msg)
 {
  struct rpcbind_args *map = msg->rpc_argp;
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  dprintk("RPC:       unregistering [%u, %u, '%s'] with "
   "local rpcbind\n",
@@ -508,7 +513,7 @@ static int rpcb_unregister_all_protofamilies(struct rpc_message *msg)
  map->r_addr = "";
  msg->rpc_proc = &rpcb_procedures4[RPCBPROC_UNSET];

- return rpcb_register_call(rpcb_local_clnt4, msg);
+ return rpcb_register_call(sn->rpcb_local_clnt4, msg);
 }
```

```
 /**
@@ -566,8 +571,9 @@ int rpcb_v4_register(const u32 program, const u32 version,
  struct rpc_message msg = {
   .rpc_argp = &map,
  };
+ struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

- if (rpcb_local_clnt4 == NULL)
+ if (sn->rpcb_local_clnt4 == NULL)
   return -EPROTONOSUPPORT;

  if (address == NULL)
```

Subject: [PATCH v3 2/3] SUNRPC: optimize net_ns dereferencing in rpcbind creation calls

Posted by Stanislav Kinsbursky on Thu, 27 Oct 2011 18:11:58 GMT

View Forum Message <> Reply to Message

Static rpcbind creation functions can be parametrized by network namespace pointer, calculated only once, instead of using init_net pointer (or taking it from current when virtualization will be comleted) in many places.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

```
---
 net/sunrpc/rpcb_clnt.c |   35 ++++++++++++++++++++++++++++++++---------
 1 files changed, 19 insertions(+), 16 deletions(-)

diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index 59bd1fb..8c873a8 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -161,10 +161,10 @@ static void rpcb_map_release(void *data)
  kfree(map);
 }

-static int rpcb_get_local(void)
+static int rpcb_get_local(struct net *net)
 {
  int cnt;
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

  spin_lock(&sn->rpcb_clnt_lock);
  if (sn->rpcb_users)
@@ -201,9 +201,10 @@ void rpcb_put_local(void)
```

```
 }
}

-static void rpcb_set_local(struct rpc_clnt *clnt, struct rpc_clnt *clnt4)
+static void rpcb_set_local(struct net *net, struct rpc_clnt *clnt,
+   struct rpc_clnt *clnt4)
 {
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);

  /* Protected by rpcb_create_local_mutex */
  sn->rpcb_local_clnt = clnt;
@@ -211,22 +212,23 @@ static void rpcb_set_local(struct rpc_clnt *clnt, struct rpc_clnt *clnt4)
  smp_wmb();
  sn->rpcb_users = 1;
  dprintk("RPC:       created new rpcb local clients (rpcb_local_clnt: "
-   "%p, rpcb_local_clnt4: %p)\n", sn->rpcb_local_clnt,
-   sn->rpcb_local_clnt4);
+   "%p, rpcb_local_clnt4: %p) for net %p%s\n",
+   sn->rpcb_local_clnt, sn->rpcb_local_clnt4,
+   net, (net == &init_net) ? " (init_net)" : "");
 }

 /*
  * Returns zero on success, otherwise a negative errno value
  * is returned.
  */
-static int rpcb_create_local_unix(void)
+static int rpcb_create_local_unix(struct net *net)
 {
  static const struct sockaddr_un rpcb_localaddr_rpcbind = {
   .sun_family  = AF_LOCAL,
   .sun_path  = RPCBIND_SOCK_PATHNAME,
  };
  struct rpc_create_args args = {
-  .net  = &init_net,
+  .net  = net,
   .protocol = XPRT_TRANSPORT_LOCAL,
   .address = (struct sockaddr *)&rpcb_localaddr_rpcbind,
   .addrsize = sizeof(rpcb_localaddr_rpcbind),
@@ -259,7 +261,7 @@ static int rpcb_create_local_unix(void)
   clnt4 = NULL;
  }

- rpcb_set_local(clnt, clnt4);
+ rpcb_set_local(net, clnt, clnt4);

 out:
```

```
  return result;
@@ -269,7 +271,7 @@ out:
   * Returns zero on success, otherwise a negative errno value
   * is returned.
   */
-static int rpcb_create_local_net(void)
+static int rpcb_create_local_net(struct net *net)
 {
   static const struct sockaddr_in rpcb_inaddr_loopback = {
     .sin_family  = AF_INET,
@@ -277,7 +279,7 @@ static int rpcb_create_local_net(void)
     .sin_port  = htons(RPCBIND_PORT),
   };
   struct rpc_create_args args = {
-   .net  = &init_net,
+   .net  = net,
     .protocol = XPRT_TRANSPORT_TCP,
     .address = (struct sockaddr *)&rpcb_inaddr_loopback,
     .addrsize = sizeof(rpcb_inaddr_loopback),
@@ -311,7 +313,7 @@ static int rpcb_create_local_net(void)
     clnt4 = NULL;
   }

- rpcb_set_local(clnt, clnt4);
+ rpcb_set_local(net, clnt, clnt4);

 out:
   return result;
@@ -325,16 +327,17 @@ int rpcb_create_local(void)
 {
   static DEFINE_MUTEX(rpcb_create_local_mutex);
   int result = 0;
+ struct net *net = &init_net;

- if (rpcb_get_local())
+ if (rpcb_get_local(net))
     return result;

   mutex_lock(&rpcb_create_local_mutex);
- if (rpcb_get_local())
+ if (rpcb_get_local(net))
     goto out;

- if (rpcb_create_local_unix() != 0)
-   result = rpcb_create_local_net();
+ if (rpcb_create_local_unix(net) != 0)
+   result = rpcb_create_local_net(net);
```

```
 out:
  mutex_unlock(&rpcb_create_local_mutex);
```

Subject: [PATCH v3 3/3] SUNRPC: optimize net_ns dereferencing in rpcbind registering calls
Posted by Stanislav Kinsbursky on Thu, 27 Oct 2011 18:30:16 GMT
View Forum Message <> Reply to Message

Static rpcbind registering functions can be parametrized by network namespace
pointer, calculated only once, instead of using init_net pointer (or taking it
from current when virtualization will be comleted) in many places.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

```
---
 net/sunrpc/rpcb_clnt.c |   18 +++++++++---------
 1 files changed, 9 insertions(+), 9 deletions(-)

diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index 8c873a8..cc0f402 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -451,14 +451,14 @@ int rpcb_register(u32 prog, u32 vers, int prot, unsigned short port)
 /*
  * Fill in AF_INET family-specific arguments to register
  */
-static int rpcb_register_inet4(const struct sockaddr *sap,
+static int rpcb_register_inet4(struct sunrpc_net *sn,
+          const struct sockaddr *sap,
          struct rpc_message *msg)
 {
  const struct sockaddr_in *sin = (const struct sockaddr_in *)sap;
  struct rpcbind_args *map = msg->rpc_argp;
  unsigned short port = ntohs(sin->sin_port);
  int result;
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  map->r_addr = rpc_sockaddr2uaddr(sap);

@@ -479,14 +479,14 @@ static int rpcb_register_inet4(const struct sockaddr *sap,
 /*
  * Fill in AF_INET6 family-specific arguments to register
  */
-static int rpcb_register_inet6(const struct sockaddr *sap,
+static int rpcb_register_inet6(struct sunrpc_net *sn,
+          const struct sockaddr *sap,
          struct rpc_message *msg)
```

```
 {
  const struct sockaddr_in6 *sin6 = (const struct sockaddr_in6 *)sap;
  struct rpcbind_args *map = msg->rpc_argp;
  unsigned short port = ntohs(sin6->sin6_port);
  int result;
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  map->r_addr = rpc_sockaddr2uaddr(sap);

@@ -504,10 +504,10 @@ static int rpcb_register_inet6(const struct sockaddr *sap,
  return result;
 }

-static int rpcb_unregister_all_protofamilies(struct rpc_message *msg)
+static int rpcb_unregister_all_protofamilies(struct sunrpc_net *sn,
+          struct rpc_message *msg)
 {
  struct rpcbind_args *map = msg->rpc_argp;
- struct sunrpc_net *sn = net_generic(&init_net, sunrpc_net_id);

  dprintk("RPC:       unregistering [%u, %u, '%s'] with "
   "local rpcbind\n",
@@ -580,13 +580,13 @@ int rpcb_v4_register(const u32 program, const u32 version,
   return -EPROTONOSUPPORT;

  if (address == NULL)
-  return rpcb_unregister_all_protofamilies(&msg);
+  return rpcb_unregister_all_protofamilies(sn, &msg);

  switch (address->sa_family) {
  case AF_INET:
-  return rpcb_register_inet4(address, &msg);
+  return rpcb_register_inet4(sn, address, &msg);
  case AF_INET6:
-  return rpcb_register_inet6(address, &msg);
+  return rpcb_register_inet6(sn, address, &msg);
  }

  return -EAFNOSUPPORT;
```

Subject: Re: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by bfields on Thu, 27 Oct 2011 20:25:14 GMT

View Forum Message <> Reply to Message

On Thu, Oct 27, 2011 at 10:10:43PM +0300, Stanislav Kinsbursky wrote:
> v3:
> 1) First two patches from previous version were squashed.

>
> This patch-set was created in context of clone of git branch:
> git://git.linux-nfs.org/projects/trondmy/nfs-2.6.git
> and rebased on tag "v3.1".
>
> This patch-set virtualizes rpcbind clients per network namespace context. IOW,
> each network namespace will have its own pair of rpcbind clients (if they would
> be created by request).
>
> Note:
> 1) this patch-set depends on "SUNRPC: make rpcbind clients allocated and
> destroyed dynamically" patch-set which has been send earlier.
> 2) init_net pointer is still used instead of current->nsproxy->net_ns,
> because I'm not sure yet about how to virtualize services. I.e. NFS callback
> services will be per netns. NFSd service will be per netns too from my pow. But
> Lockd can be per netns or one for all.

I'm not sure what you mean by that; could you explain?

--b.

> And also we have NFSd file system, which
> is not virtualized yet.
>
> The following series consists of:
>
> ---
>
> Stanislav Kinsbursky (3):
>   SUNRPC: move rpcbind internals to sunrpc part of network namespace context
>   SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
>   SUNRPC: optimize net_ns dereferencing in rpcbind registering calls
>
>
> net/sunrpc/netns.h | 5 ++
> net/sunrpc/rpcb_clnt.c | 103 ++++++++++++++++++++++++++++++++++++++----------
> 2 files changed, 61 insertions(+), 47 deletions(-)
>
> --
> Signature

Subject: Re: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by Stanislav Kinsbursky on Fri, 28 Oct 2011 09:24:45 GMT
View Forum Message <> Reply to Message

> On Thu, Oct 27, 2011 at 10:10:43PM +0300, Stanislav Kinsbursky wrote:

>> v3:
>> 1) First two patches from previous version were squashed.
>>
>> This patch-set was created in context of clone of git branch:
>> git://git.linux-nfs.org/projects/trondmy/nfs-2.6.git
>> and rebased on tag "v3.1".
>>
>> This patch-set virtualizes rpcbind clients per network namespace context. IOW,
>> each network namespace will have its own pair of rpcbind clients (if they would
>> be created by request).
>>
>> Note:
>> 1) this patch-set depends on "SUNRPC: make rpcbind clients allocated and
>> destroyed dynamically" patch-set which has been send earlier.
>> 2) init_net pointer is still used instead of current->nsproxy->net_ns,
>> because I'm not sure yet about how to virtualize services. I.e. NFS callback
>> services will be per netns. NFSd service will be per netns too from my pow. But
>> Lockd can be per netns or one for all.
>
> I'm not sure what you mean by that; could you explain?
>

This patch-set was created before you've sent your NFSd plan and we disacussed
Lockd per netns.
So, this sentence: "NFSd service will be per netns too from my pow" is obsolete.
And Lockd will be one for all.
Or you are asking about something else?

> --b.
>
>> And also we have NFSd file system, which
>> is not virtualized yet.
>>
>> The following series consists of:
>>
>> ---
>>
>> Stanislav Kinsbursky (3):
>>   SUNRPC: move rpcbind internals to sunrpc part of network namespace context
>>   SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
>>   SUNRPC: optimize net_ns dereferencing in rpcbind registering calls
>>
>>
>> net/sunrpc/netns.h | 5 ++
>> net/sunrpc/rpcb_clnt.c | 103 ++++++++++++++++++++++++++++++++----------------
>> 2 files changed, 61 insertions(+), 47 deletions(-)
>>
>> --

>> Signature

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by bfields on Fri, 28 Oct 2011 09:30:36 GMT
View Forum Message <> Reply to Message

On Fri, Oct 28, 2011 at 01:24:45PM +0400, Stanislav Kinsbursky wrote:
> This patch-set was created before you've sent your NFSd plan and we
> disacussed Lockd per netns.
> So, this sentence: "NFSd service will be per netns too from my pow"
> is obsolete. And Lockd will be one for all.

I believe lockd should be pert-netns--at least that's what the server
needs.

(The single lockd thread may handle requests from all netns, but it
should behave like a different service depending on netns, so its data
structures, etc. will need to be per-ns.

--b.

> Or you are asking about something else?
>
> >--b.
> >
> >>And also we have NFSd file system, which
> >>is not virtualized yet.
> >>
> >>The following series consists of:
> >>
> >>---
> >>
> >>Stanislav Kinsbursky (3):
> >>   SUNRPC: move rpcbind internals to sunrpc part of network namespace context
> >>   SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
> >>   SUNRPC: optimize net_ns dereferencing in rpcbind registering calls
> >>
> >>
> >> net/sunrpc/netns.h | 5 ++
> >> net/sunrpc/rpcb_clnt.c | 103 ++++++++++++++++++++++++++++++++++++++----------
> >> 2 files changed, 61 insertions(+), 47 deletions(-)
> >>

> >>--
> >>Signature
>
>
> --
> Best regards,
> Stanislav Kinsbursky

Subject: Re: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by Stanislav Kinsbursky on Fri, 28 Oct 2011 09:41:44 GMT
View Forum Message <> Reply to Message

> On Fri, Oct 28, 2011 at 01:24:45PM +0400, Stanislav Kinsbursky wrote:
>> This patch-set was created before you've sent your NFSd plan and we
>> disacussed Lockd per netns.
>> So, this sentence: "NFSd service will be per netns too from my pow"
>> is obsolete. And Lockd will be one for all.
>
> I believe lockd should be pert-netns--at least that's what the server
> needs.
>
> (The single lockd thread may handle requests from all netns, but it
> should behave like a different service depending on netns, so its data
> structures, etc. will need to be per-ns.
>

Sure. Looks like we have misunderstanding here. When I said, that Lockd should be one for all, I meaned, that we will have only one kthread for all ns (not one per ns). Private data will be per net ns, of course.

BTW, Bruce, please, have a brief look at my e-mail to linux-nfs@vger.kernel.org named "SUNRPC: non-exclusive pipe creation".
I've done a lot in "RPC pipefs per net ns" task, and going to send first patches soon. But right now I'm really confused will this non-exclusive pipes creation and almost ready so remove this functionality. But I'm afraid, that I've missed something. Would be greatly appreciate for your opinion about my question.

> --b.
>
>> Or you are asking about something else?
>>
>>> --b.
>>>
>>>> And also we have NFSd file system, which
>>>> is not virtualized yet.
>>>>

>>>> The following series consists of:
>>>>
>>>> ---
>>>>
>>>> Stanislav Kinsbursky (3):
>>>>   SUNRPC: move rpcbind internals to sunrpc part of network namespace context
>>>>   SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
>>>>   SUNRPC: optimize net_ns dereferencing in rpcbind registering calls
>>>>
>>>>
>>>> net/sunrpc/netns.h | 5 ++
>>>> net/sunrpc/rpcb_clnt.c | 103 ++++++++++++++++++++++++++++++++----------------
>>>> 2 files changed, 61 insertions(+), 47 deletions(-)
>>>>
>>>> --
>>>> Signature
>>
>>
>> --
>> Best regards,
>> Stanislav Kinsbursky

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by bfields on Fri, 04 Nov 2011 22:10:45 GMT
View Forum Message <> Reply to Message

On Fri, Oct 28, 2011 at 01:41:49PM +0400, Stanislav Kinsbursky wrote:

> >On Fri, Oct 28, 2011 at 01:24:45PM +0400, Stanislav Kinsbursky wrote:
> >>This patch-set was created before you've sent your NFSd plan and we
> >>disacussed Lockd per netns.
> >>So, this sentence: "NFSd service will be per netns too from my pow"
> >>is obsolete. And Lockd will be one for all.
> >
> >I believe lockd should be pert-netns--at least that's what the server
> >needs.
> >
> >(The single lockd thread may handle requests from all netns, but it
> >should behave like a different service depending on netns, so its data
> >structures, etc. will need to be per-ns.
> >
>

> Sure. Looks like we have misunderstanding here. When I said, that
> Lockd should be one for all, I meaned, that we will have only one
> kthread for all ns (not one per ns). Private data will be per net
> ns, of course.
>
> BTW, Bruce, please, have a brief look at my e-mail to
> linux-nfs@vger.kernel.org named "SUNRPC: non-exclusive pipe
> creation".
> I've done a lot in "RPC pipefs per net ns" task, and going to send
> first patches soon. But right now I'm really confused will this
> non-exclusive pipes creation and almost ready so remove this
> functionality. But I'm afraid, that I've missed something. Would be
> greatly appreciate for your opinion about my question.

Sorry for the delay--it looks reasonable to me on a quick skim, but I'm
assuming it's Trond that will need to review this.

--b.

>
> >--b.
> >
> >>Or you are asking about something else?
> >>
> >>>--b.
> >>>
> >>>>And also we have NFSd file system, which
> >>>>is not virtualized yet.
> >>>>
> >>>>The following series consists of:
> >>>>
> >>>>---
> >>>>
> >>>>Stanislav Kinsbursky (3):
> >>>>   SUNRPC: move rpcbind internals to sunrpc part of network namespace context
> >>>>   SUNRPC: optimize net_ns dereferencing in rpcbind creation calls
> >>>>   SUNRPC: optimize net_ns dereferencing in rpcbind registering calls
> >>>>
> >>>>
> >>>> net/sunrpc/netns.h | 5 ++
> >>>> net/sunrpc/rpcb_clnt.c | 103 +++++++++++++++++++++++++++++++++------------------------
> >>>> 2 files changed, 61 insertions(+), 47 deletions(-)
> >>>>
> >>>>--
> >>>>Signature
> >>
> >>
> >>--

> >>Best regards,
> >>Stanislav Kinsbursky
>
>
> --
> Best regards,
> Stanislav Kinsbursky

Subject: Re: [PATCH v3 0/3] SUNRPC: rcbind clients virtualization
Posted by Stanislav Kinsbursky on Mon, 07 Nov 2011 08:02:51 GMT
View Forum Message <> Reply to Message

>> BTW, Bruce, please, have a brief look at my e-mail to
>> linux-nfs@vger.kernel.org named "SUNRPC: non-exclusive pipe
>> creation".
>> I've done a lot in "RPC pipefs per net ns" task, and going to send
>> first patches soon. But right now I'm really confused will this
>> non-exclusive pipes creation and almost ready so remove this
>> functionality. But I'm afraid, that I've missed something. Would be
>> greatly appreciate for your opinion about my question.
>
> Sorry for the delay--it looks reasonable to me on a quick skim, but I'm
> assuming it's Trond that will need to review this.
>

Thanks for your review.
And, yep, still waiting for Trond answer too...

--
Best regards,
Stanislav Kinsbursky
