
Subject: [PATCH v2] Lockd: pass network namespace to creation and destruction routines

Posted by Stanislav Kinsbursky on Thu, 29 Mar 2012 14:54:33 GMT

[View Forum Message](#) <> [Reply to Message](#)

v2: dereference of most probably already released nlm_host removed in
nlmclnt_done() and reclaimer().

These routines are called from locks reclaimer() kernel thread. This thread
works in "init_net" network context and currently relays on persence on lockd
thread and it's per-net resources. Thus lockd_up() and lockd_down() can't relay
on current network context. So let's pass corrent one into them.

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

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

```
fs/lockd/clntlock.c      | 13 ++++++-----
fs/lockd/svc.c          |  7 +-----
fs/nfsd/nfssvc.c        |  6 +---
include/linux/lockd/bind.h|  4 +--
4 files changed, 16 insertions(+), 14 deletions(-)
```

```
diff --git a/fs/lockd/clntlock.c b/fs/lockd/clntlock.c
index ba1dc2e..ca0a080 100644
--- a/fs/lockd/clntlock.c
+++ b/fs/lockd/clntlock.c
@@ -56,7 +56,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
    u32 nlm_version = (nlm_init->nfs_version == 2) ? 1 : 4;
    int status;

-    status = lockd_up();
+    status = lockd_up(nlm_init->net);
    if (status < 0)
        return ERR_PTR(status);

@@ -65,7 +65,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
    nlm_init->hostname, nlm_init->noresvport,
    nlm_init->net);
    if (host == NULL) {
-        lockd_down();
+        lockd_down(nlm_init->net);
        return ERR_PTR(-ENOLCK);
    }

@@ -80,8 +80,10 @@ EXPORT_SYMBOL_GPL(nlmclnt_init);
 */
```

```

void nlmclnt_done(struct nlm_host *host)
{
+ struct net *net = host->net;
+
nlmclnt_release_host(host);
- lockd_down();
+ lockd_down(net);
}
EXPORT_SYMBOL_GPL(nlmclnt_done);

@@ -220,11 +222,12 @@ reclaimer(void *ptr)
    struct nlm_wait *block;
    struct file_lock *fl, *next;
    u32 nsmstate;
+ struct net *net = host->net;

    allow_signal(SIGKILL);

    down_write(&host->h_rwsem);
- lockd_up(); /* note: this cannot fail as lockd is already running */
+ lockd_up(net); /* note: this cannot fail as lockd is already running */

    dprintk("lockd: reclaiming locks for host %s\n", host->h_name);

@@ -275,6 +278,6 @@ restart:

/* Release host handle after use */
nlmclnt_release_host(host);
- lockd_down();
+ lockd_down(net);
return 0;
}
diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
index b34100e..ce4c80e 100644
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -295,11 +295,10 @@ static void lockd_down_net(struct net *net)
/*
 * Bring up the lockd process if it's not already up.
 */
-int lockd_up(void)
+int lockd_up(struct net *net)
{
    struct svc_serv *serv;
    int error = 0;
- struct net *net = current->nsproxy->net_ns;

    mutex_lock(&nlmsvc_mutex);

```

```

/*
@@ -377,12 +376,12 @@ EXPORT_SYMBOL_GPL(lockd_up);
 * Decrement the user count and bring down lockd if we're the last.
 */
void
-lockd_down(void)
+lockd_down(struct net *net)
{
    mutex_lock(&nlmsvc_mutex);
    if (nlmsvc_users) {
        if (--nlmsvc_users) {
-            lockd_down_net(current->nsproxy->net_ns);
+            lockd_down_net(net);
            goto out;
        }
    } else {
diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
index fce472f..0f3e35b 100644
--- a/fs/nfsd/nfssvc.c
+++ b/fs/nfsd/nfssvc.c
@@ -220,7 +220,7 @@ static int nfsd_startup(unsigned short port, int nrsvrs)
    ret = nfsd_init_socks(port);
    if (ret)
        goto out_racache;
-    ret = lockd_up();
+    ret = lockd_up(&init_net);
    if (ret)
        goto out_racache;
    ret = nfs4_state_start();
@@ -229,7 +229,7 @@ static int nfsd_startup(unsigned short port, int nrsvrs)
    nfsd_up = true;
    return 0;
out_lockd:
-    lockd_down();
+    lockd_down(&init_net);
out_racache:
    nfsd_racache_shutdown();
    return ret;
@@ -246,7 +246,7 @@ static void nfsd_shutdown(void)
    if (!nfsd_up)
        return;
    nfs4_state_shutdown();
-    lockd_down();
+    lockd_down(&init_net);
    nfsd_racache_shutdown();
    nfsd_up = false;
}
diff --git a/include/linux/lockd/bind.h b/include/linux/lockd/bind.h
```

```
index 11a966e..4d24d64 100644
--- a/include/linux/lockd/bind.h
+++ b/include/linux/lockd/bind.h
@@ -54,7 +54,7 @@ extern void nlmclnt_done(struct nlm_host *host);

extern int nlmclnt_proc(struct nlm_host *host, int cmd,
    struct file_lock *fl);
-extern int lockd_up(void);
-extern void lockd_down(void);
+extern int lockd_up(struct net *net);
+extern void lockd_down(struct net *net);

#endif /* LINUX_LOCKD_BIND_H */
```

Subject: Re: [PATCH v2] Lockd: pass network namespace to creation and destruction routines

Posted by [bfields](#) on Wed, 11 Apr 2012 16:11:14 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Thu, Mar 29, 2012 at 06:54:33PM +0400, Stanislav Kinsbursky wrote:
> v2: dereference of most probably already released nlm_host removed in
> nlmclnt_done() and reclaimer().

Did you want this in Trond's tree or mine?

--b.

>
> These routines are called from locks reclaimer() kernel thread. This thread
> works in "init_net" network context and currently relays on persence on lockd
> thread and it's per-net resources. Thus lockd_up() and lockd_down() can't relay
> on current network context. So let's pass corrent one into them.
>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>
> ---
> fs/lockd/clntlock.c | 13 ++++++-----
> fs/lockd/svc.c | 7 +----
> fs/nfsd/nfssvc.c | 6 +---
> include/linux/lockd/bind.h | 4 +--
> 4 files changed, 16 insertions(+), 14 deletions(-)
>
> diff --git a/fs/lockd/clntlock.c b/fs/lockd/clntlock.c
> index ba1dc2e..ca0a080 100644
> --- a/fs/lockd/clntlock.c

```

> +++
> b/fs/lockd/clntlock.c
> @@ -56,7 +56,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
>   u32 nlm_version = (nlm_init->nfs_version == 2) ? 1 : 4;
>   int status;
>
> - status = lockd_up();
> + status = lockd_up(nlm_init->net);
>   if (status < 0)
>     return ERR_PTR(status);
>
> @@ -65,7 +65,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
>   nlm_init->hostname, nlm_init->noresvport,
>   nlm_init->net);
>   if (host == NULL) {
> -   lockd_down();
> +   lockd_down(nlm_init->net);
>   return ERR_PTR(-ENOLCK);
> }
>
> @@ -80,8 +80,10 @@ EXPORT_SYMBOL_GPL(nlmclnt_init);
> */
> void nlmclnt_done(struct nlm_host *host)
> {
> + struct net *net = host->net;
> +
>   nlmclnt_release_host(host);
> - lockd_down();
> + lockd_down(net);
> }
> EXPORT_SYMBOL_GPL(nlmclnt_done);
>
> @@ -220,11 +222,12 @@ reclamer(void *ptr)
>   struct nlm_wait *block;
>   struct file_lock *fl, *next;
>   u32 nsystate;
> + struct net *net = host->net;
>
>   allow_signal(SIGKILL);
>
>   down_write(&host->h_rwsem);
> - lockd_up(); /* note: this cannot fail as lockd is already running */
> + lockd_up(net); /* note: this cannot fail as lockd is already running */
>
>   dprintk("lockd: reclaiming locks for host %s\n", host->h_name);
>
> @@ -275,6 +278,6 @@ restart:
>
> /* Release host handle after use */

```

```

> nlmclnt_release_host(host);
> - lockd_down();
> + lockd_down(net);
> return 0;
> }
> diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
> index b34100e..ce4c80e 100644
> --- a/fs/lockd/svc.c
> +++ b/fs/lockd/svc.c
> @@ -295,11 +295,10 @@ static void lockd_down_net(struct net *net)
> /*
> * Bring up the lockd process if it's not already up.
> */
> -int lockd_up(void)
> +int lockd_up(struct net *net)
> {
>     struct svc_serv *serv;
>     int error = 0;
> - struct net *net = current->nsproxy->net_ns;
>
>     mutex_lock(&nlmsvc_mutex);
> /*
> @@ -377,12 +376,12 @@ EXPORT_SYMBOL_GPL(lockd_up);
> * Decrement the user count and bring down lockd if we're the last.
> */
> void
> -lockd_down(void)
> +lockd_down(struct net *net)
> {
>     mutex_lock(&nlmsvc_mutex);
>     if (nlmsvc_users) {
>         if (--nlmsvc_users) {
> -     lockd_down_net(current->nsproxy->net_ns);
> +     lockd_down_net(net);
>         goto out;
>     }
>     } else {
>         diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
> index fce472f..0f3e35b 100644
> --- a/fs/nfsd/nfssvc.c
> +++ b/fs/nfsd/nfssvc.c
> @@ -220,7 +220,7 @@ static int nfsd_startup(unsigned short port, int nrsvr)
>     ret = nfsd_init_socks(port);
>     if (ret)
>         goto out_racache;
> - ret = lockd_up();
> + ret = lockd_up(&init_net);
>     if (ret)

```

```

>     goto out_racache;
>     ret = nfs4_state_start();
> @@ -229,7 +229,7 @@ static int nfsd_startup(unsigned short port, int nrsvs)
>     nfsd_up = true;
>     return 0;
> out_lockd:
> - lockd_down();
> + lockd_down(&init_net);
> out_racache:
>     nfsd_racache_shutdown();
>     return ret;
> @@ -246,7 +246,7 @@ static void nfsd_shutdown(void)
>     if (!nfsd_up)
>     return;
>     nfs4_state_shutdown();
> - lockd_down();
> + lockd_down(&init_net);
>     nfsd_racache_shutdown();
>     nfsd_up = false;
> }
> diff --git a/include/linux/lockd/bind.h b/include/linux/lockd/bind.h
> index 11a966e..4d24d64 100644
> --- a/include/linux/lockd/bind.h
> +++ b/include/linux/lockd/bind.h
> @@ -54,7 +54,7 @@ extern void nlmclnt_done(struct nlm_host *host);
>
> extern int nlmclnt_proc(struct nlm_host *host, int cmd,
>     struct file_lock *fl);
> -extern int lockd_up(void);
> -extern void lockd_down(void);
> +extern int lockd_up(struct net *net);
> +extern void lockd_down(struct net *net);
>
> #endif /* LINUX_LOCKD_BIND_H */
>
```

Subject: Re: [PATCH v2] Lockd: pass network namespace to creation and destruction routines

Posted by [Stanislav Kinsbursky](#) on Wed, 11 Apr 2012 16:12:04 GMT

[View Forum Message](#) <> [Reply to Message](#)

```

> On Thu, Mar 29, 2012 at 06:54:33PM +0400, Stanislav Kinsbursky wrote:
>> v2: dereference of most probably already released nlm_host removed in
>> nlmclnt_done() and reclaimer().
>
> Did you want this in Trond's tree or mine?
```

>

Your tree is preferred since I'm working with it.

> --b.
>
>>
>> These routines are called from locks reclaimer() kernel thread. This thread
>> works in "init_net" network context and currently relays on persence on lockd
>> thread and it's per-net resources. Thus lockd_up() and lockd_down() can't relay
>> on current network context. So let's pass corrent one into them.
>>
>> Signed-off-by: Stanislav Kinsbursky<skinsbursky@parallels.com>
>>
>> Signed-off-by: Stanislav Kinsbursky<skinsbursky@parallels.com>
>>
>> ---
>> fs/lockd/clntlock.c | 13 ++++++-----
>> fs/lockd/svc.c | 7 +----
>> fs/nfsd/nfssvc.c | 6 +----
>> include/linux/lockd/bind.h | 4 +--
>> 4 files changed, 16 insertions(+), 14 deletions(-)
>>
>> diff --git a/fs/lockd/clntlock.c b/fs/lockd/clntlock.c
>> index ba1dc2e..ca0a080 100644
>> --- a/fs/lockd/clntlock.c
>> +++ b/fs/lockd/clntlock.c
>> @@ -56,7 +56,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
>> u32 nlm_version = (nlm_init->nfs_version == 2) ? 1 : 4;
>> int status;
>>
>> - status = lockd_up();
>> + status = lockd_up(nlm_init->net);
>> if (status < 0)
>> return ERR_PTR(status);
>>
>> @@ -65,7 +65,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
>> nlm_init->hostname, nlm_init->noresvport,
>> nlm_init->net);
>> if (host == NULL) {
>> - lockd_down();
>> + lockd_down(nlm_init->net);
>> return ERR_PTR(-ENOLCK);
>> }
>>
>> @@ -80,8 +80,10 @@ EXPORT_SYMBOL_GPL(nlmclnt_init);
>> */
>> void nlmclnt_done(struct nlm_host *host)

```

>> {
>> + struct net *net = host->net;
>> +
>>   nlmclnt_release_host(host);
>> - lockd_down();
>> + lockd_down(net);
>> }
>> EXPORT_SYMBOL_GPL(nlmclnt_done);
>>
>> @@ -220,11 +222,12 @@ reclamer(void *ptr)
>>   struct nlm_wait *block;
>>   struct file_lock *fl, *next;
>>   u32 nsmstate;
>> + struct net *net = host->net;
>>
>>   allow_signal(SIGKILL);
>>
>>   down_write(&host->h_rwsem);
>> - lockd_up(); /* note: this cannot fail as lockd is already running */
>> + lockd_up(net); /* note: this cannot fail as lockd is already running */
>>
>>   dprintk("lockd: reclaiming locks for host %s\n", host->h_name);
>>
>> @@ -275,6 +278,6 @@ restart:
>>
>>   /* Release host handle after use */
>>   nlmclnt_release_host(host);
>> - lockd_down();
>> + lockd_down(net);
>>   return 0;
>> }
>> diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
>> index b34100e..ce4c80e 100644
>> --- a/fs/lockd/svc.c
>> +++ b/fs/lockd/svc.c
>> @@ -295,11 +295,10 @@ static void lockd_down_net(struct net *net)
>> /*
>>   * Bring up the lockd process if it's not already up.
>> */
>> -int lockd_up(void)
>> +int lockd_up(struct net *net)
>> {
>>   struct svc_serv *serv;
>>   int error = 0;
>> - struct net *net = current->nsproxy->net_ns;
>>
>>   mutex_lock(&nlmsvc_mutex);
>> /*

```

```

>> @@ -377,12 +376,12 @@ EXPORT_SYMBOL_GPL(lockd_up);
>>   * Decrement the user count and bring down lockd if we're the last.
>>   */
>> void
>> -lockd_down(void)
>> +lockd_down(struct net *net)
>> {
>>   mutex_lock(&nlmsvc_mutex);
>>   if (nlmsvc_users) {
>>     if (--nlmsvc_users) {
>> -   lockd_down_net(current->nproxy->net_ns);
>> +   lockd_down_net(net);
>>     goto out;
>>   }
>> } else {
>> diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
>> index fce472f..0f3e35b 100644
>> --- a/fs/nfsd/nfssvc.c
>> +++ b/fs/nfsd/nfssvc.c
>> @@ -220,7 +220,7 @@ static int nfsd_startup(unsigned short port, int nrsvs)
>>   ret = nfsd_init_socks(port);
>>   if (ret)
>>     goto out_racache;
>> - ret = lockd_up();
>> + ret = lockd_up(&init_net);
>>   if (ret)
>>     goto out_racache;
>>   ret = nfs4_state_start();
>> @@ -229,7 +229,7 @@ static int nfsd_startup(unsigned short port, int nrsvs)
>>   nfsd_up = true;
>>   return 0;
>> out_lockd:
>> - lockd_down();
>> + lockd_down(&init_net);
>> out_racache:
>>   nfsd_racache_shutdown();
>>   return ret;
>> @@ -246,7 +246,7 @@ static void nfsd_shutdown(void)
>>   if (!nfsd_up)
>>     return;
>>   nfs4_state_shutdown();
>> - lockd_down();
>> + lockd_down(&init_net);
>>   nfsd_racache_shutdown();
>>   nfsd_up = false;
>> }
>> diff --git a/include/linux/lockd/bind.h b/include/linux/lockd/bind.h
>> index 11a966e..4d24d64 100644

```

```
>> --- a/include/linux/lockd/bind.h
>> +++ b/include/linux/lockd/bind.h
>> @@ -54,7 +54,7 @@ extern void nlmclnt_done(struct nlm_host *host);
>>
>> extern int nlmclnt_proc(struct nlm_host *host, int cmd,
>>     struct file_lock *fl);
>> -extern int lockd_up(void);
>> -extern void lockd_down(void);
>> +extern int lockd_up(struct net *net);
>> +extern void lockd_down(struct net *net);
>>
>> #endif /* LINUX_LOCKD_BIND_H */
>>
```

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH v2] Lockd: pass network namespace to creation and destruction routines

Posted by [bfields](#) on Wed, 11 Apr 2012 21:32:45 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Wed, Apr 11, 2012 at 08:12:04PM +0400, Stanislav Kinsbursky wrote:

```
> >On Thu, Mar 29, 2012 at 06:54:33PM +0400, Stanislav Kinsbursky wrote:
> >>v2: dereference of most probably already released nlm_host removed in
> >>nlmclnt_done() and reclaimer().
> >
> >Did you want this in Trond's tree or mine?
> >
>
> Your tree is preferred since I'm working with it.
```

OK, applying.--b.

```
>
> >--b.
> >
> >>
> >>These routines are called from locks reclaimer() kernel thread. This thread
> >>works in "init_net" network context and currently relays on persence on lockd
> >>thread and it's per-net resources. Thus lockd_up() and lockd_down() can't relay
> >>on current network context. So let's pass corrent one into them.
> >>
> >>Signed-off-by: Stanislav Kinsbursky<skinsbursky@parallels.com>
```

```

> >>
> >>Signed-off-by: Stanislav Kinsbursky<skinsbursky@parallels.com>
> >>
> >>---
> >> fs/lockd/clntlock.c      | 13 ++++++-----
> >> fs/lockd/svc.c          |  7 +++++-
> >> fs/nfsd/nfssvc.c        |  6 +++++
> >> include/linux/lockd/bind.h |  4 +---
> >> 4 files changed, 16 insertions(+), 14 deletions(-)
> >>
> >>diff --git a/fs/lockd/clntlock.c b/fs/lockd/clntlock.c
> >>index ba1dc2e..ca0a080 100644
> >>--- a/fs/lockd/clntlock.c
> >>+++ b/fs/lockd/clntlock.c
> >>@@ -56,7 +56,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
> >> u32 nlm_version = (nlm_init->nfs_version == 2) ? 1 : 4;
> >> int status;
> >>
> >>- status = lockd_up();
> >>+ status = lockd_up(nlm_init->net);
> >> if (status< 0)
> >> return ERR_PTR(status);
> >>
> >>@@ -65,7 +65,7 @@ struct nlm_host *nlmclnt_init(const struct nlmclnt_initdata *nlm_init)
> >>     nlm_init->hostname, nlm_init->noresvport,
> > >     nlm_init->net);
> >> if (host == NULL) {
> >>- lockd_down();
> >>+ lockd_down(nlm_init->net);
> >> return ERR_PTR(-ENOLCK);
> >> }
> >>
> >>@@ -80,8 +80,10 @@ EXPORT_SYMBOL_GPL(nlmclnt_init);
> >> */
> >> void nlmclnt_done(struct nlm_host *host)
> >> {
> >>+ struct net *net = host->net;
> >>+
> >> nlmclnt_release_host(host);
> >>- lockd_down();
> >>+ lockd_down(net);
> >> }
> >> EXPORT_SYMBOL_GPL(nlmclnt_done);
> >>
> >>@@ -220,11 +222,12 @@ reclamer(void *ptr)
> >> struct nlm_wait *block;
> >> struct file_lock *fl, *next;
> >> u32 nsmstate;

```

```

> >>+ struct net *net = host->net;
> >>
> >> allow_signal(SIGKILL);
> >>
> >> down_write(&host->h_rwsem);
> >>- lockd_up() /* note: this cannot fail as lockd is already running */
> >>+ lockd_up(net) /* note: this cannot fail as lockd is already running */
> >>
> >> dprintk("lockd: reclaiming locks for host %s\n", host->h_name);
> >>
> >>@@ -275,6 +278,6 @@ restart:
> >>
> >> /* Release host handle after use */
> >> nlmclnt_release_host(host);
> >>- lockd_down();
> >>+ lockd_down(net);
> >> return 0;
> >> }
> >>diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
> >>index b34100e..ce4c80e 100644
> >>--- a/fs/lockd/svc.c
> >>+++ b/fs/lockd/svc.c
> >>@@ -295,11 +295,10 @@ static void lockd_down_net(struct net *net)
> >> /*
> >> * Bring up the lockd process if it's not already up.
> >> */
> >>-int lockd_up(void)
> >>+int lockd_up(struct net *net)
> >> {
> >>   struct svc_serv *serv;
> >>   int error = 0;
> >>- struct net *net = current->nsproxy->net_ns;
> >>
> >>   mutex_lock(&nlmsvc_mutex);
> >> /*
> >>@@ -377,12 +376,12 @@ EXPORT_SYMBOL_GPL(lockd_up);
> >> * Decrement the user count and bring down lockd if we're the last.
> >> */
> >> void
> >>-lockd_down(void)
> >>+lockd_down(struct net *net)
> >> {
> >>   mutex_lock(&nlmsvc_mutex);
> >>   if (nlmsvc_users) {
> >>     if (--nlmsvc_users) {
> >>-   lockd_down_net(current->nsproxy->net_ns);
> >>+   lockd_down_net(net);
> >>     goto out;

```

```

> >>   }
> >> } else {
> >>diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
> >>index fce472f..0f3e35b 100644
> >>--- a/fs/nfsd/nfssvc.c
> >>+++ b/fs/nfsd/nfssvc.c
> >>@@ -220,7 +220,7 @@ static int nfsd_startup(unsigned short port, int nrsvs)
> >>    ret = nfsd_init_socks(port);
> >>    if (ret)
> >>        goto out_racache;
> >>-    ret = lockd_up();
> >>+    ret = lockd_up(&init_net);
> >>    if (ret)
> >>        goto out_racache;
> >>    ret = nfs4_state_start();
> >>@@ -229,7 +229,7 @@ static int nfsd_startup(unsigned short port, int nrsvs)
> >>    nfsd_up = true;
> >>    return 0;
> >> out_lockd:
> >>-    lockd_down();
> >>+    lockd_down(&init_net);
> >> out_racache:
> >>    nfsd_racache_shutdown();
> >>    return ret;
> >>@@ -246,7 +246,7 @@ static void nfsd_shutdown(void)
> >>    if (!nfsd_up)
> >>        return;
> >>    nfs4_state_shutdown();
> >>-    lockd_down();
> >>+    lockd_down(&init_net);
> >>    nfsd_racache_shutdown();
> >>    nfsd_up = false;
> >> }
> >>diff --git a/include/linux/lockd/bind.h b/include/linux/lockd/bind.h
> >>index 11a966e..4d24d64 100644
> >>--- a/include/linux/lockd/bind.h
> >>+++ b/include/linux/lockd/bind.h
> >>@@ -54,7 +54,7 @@ extern void nlmclnt_done(struct nlm_host *host);
> >>
> >>    extern int nlmclnt_proc(struct nlm_host *host, int cmd,
> >>        struct file_lock *fl);
> >>-extern int lockd_up(void);
> >>-extern void lockd_down(void);
> >>+extern int lockd_up(struct net *net);
> >>+extern void lockd_down(struct net *net);
> >>
> >> #endif /* LINUX_LOCKD_BIND_H */
> >>

```

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