Subject: [PATCH 0/2] NFS: callback shutdown panic fix
Posted by Stanislav Kinsbursky on Fri, 01 Jun 2012 11:17:34 GMT
View Forum Message <> Reply to Message

This patch set is the back-port of "SUNRPC: separate per-net data creation
from service" patch set and "NFS: hard-code init_net for NFS callback
transports" patch for 3.5 kernel.

It fixes NULL pointer dereference in svc_destroy(), reported by Dave Jones in
"3.4. sunrpc oops during shutdown" e-mail.

The following series implements...

Stanislav Kinsbursky (2):
    SUNRPC: new svc_bind() routine introduced
    SUNRPC: move per-net operations from svc_destroy()

```
 fs/lockd/svc.c            |   33 ++++++++++++++++++++++------------
 fs/nfs/callback.c         |   16 ++++++++++++---
 fs/nfsd/nfsctl.c          |   12 ++++++++++---
 fs/nfsd/nfssvc.c          |   23 +++++++++++++++++++++++
 include/linux/sunrpc/svc.h |    1 +
 net/sunrpc/rpcb_clnt.c    |   12 +++++++-----
 net/sunrpc/svc.c          |   23 +++++++++++++----------
 7 files changed, 84 insertions(+), 36 deletions(-)
```

Subject: [PATCH 1/2] SUNRPC: new svc_bind() routine introduced
Posted by Stanislav Kinsbursky on Fri, 01 Jun 2012 11:17:42 GMT
View Forum Message <> Reply to Message

This patch was back-ported from 3.5 kernel.

New routine is responsible for service registration in specified network
context.
The idea is to separate service creation from per-net operations.
Since registering service with svc_bind() can fail, then service will be
destroyed and during destruction it will try to unregister itself from
rpcbind. In this case unregister have to be skipped.

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

```
 fs/lockd/svc.c            |    6 ++++++
 fs/nfs/callback.c         |   13 ++++++++++---
 fs/nfsd/nfssvc.c          |    9 +++++++++
```

```
 include/linux/sunrpc/svc.h |    1 +
 net/sunrpc/rpcb_clnt.c     |   12 +++++++-----
 net/sunrpc/svc.c           |   19 ++++++++++---------
 6 files changed, 43 insertions(+), 17 deletions(-)

diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
index f49b9af..b1d0708 100644
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -325,6 +325,12 @@ int lockd_up(void)
   goto out;
  }

+ error = svc_bind(serv, net);
+ if (error < 0) {
+  printk(KERN_WARNING "lockd_up: bind service failed\n");
+  goto destroy_and_out;
+ }
+
  error = make_socks(serv, net);
  if (error < 0)
   goto destroy_and_out;
diff --git a/fs/nfs/callback.c b/fs/nfs/callback.c
index eb95f50..0563237 100644
--- a/fs/nfs/callback.c
+++ b/fs/nfs/callback.c
@@ -106,7 +106,7 @@ nfs4_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
 {
  int ret;

- ret = svc_create_xprt(serv, "tcp", xprt->xprt_net, PF_INET,
+ ret = svc_create_xprt(serv, "tcp", &init_net, PF_INET,
     nfs_callback_set_tcpport, SVC_SOCK_ANONYMOUS);
  if (ret <= 0)
   goto out_err;
@@ -114,7 +114,7 @@ nfs4_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
  dprintk("NFS: Callback listener port = %u (af %u)\n",
    nfs_callback_tcpport, PF_INET);

- ret = svc_create_xprt(serv, "tcp", xprt->xprt_net, PF_INET6,
+ ret = svc_create_xprt(serv, "tcp", &init_net, PF_INET6,
     nfs_callback_set_tcpport, SVC_SOCK_ANONYMOUS);
  if (ret > 0) {
   nfs_callback_tcpport6 = ret;
@@ -183,7 +183,7 @@ nfs41_callback_up(struct svc_serv *serv, struct rpc_xprt *xprt)
   * fore channel connection.
   * Returns the input port (0) and sets the svc_serv bc_xprt on success
   */
```

```
- ret = svc_create_xprt(serv, "tcp-bc", xprt->xprt_net, PF_INET, 0,
+ ret = svc_create_xprt(serv, "tcp-bc", &init_net, PF_INET, 0,
          SVC_SOCK_ANONYMOUS);
  if (ret < 0) {
   rqstp = ERR_PTR(ret);
@@ -253,6 +253,7 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
  char svc_name[12];
  int ret = 0;
  int minorversion_setup;
+ struct net *net = &init_net;

  mutex_lock(&nfs_callback_mutex);
  if (cb_info->users++ || cb_info->task != NULL) {
@@ -265,6 +266,12 @@ int nfs_callback_up(u32 minorversion, struct rpc_xprt *xprt)
   goto out_err;
  }

+ ret = svc_bind(serv, net);
+ if (ret < 0) {
+  printk(KERN_WARNING "NFS: bind callback service failed\n");
+  goto out_err;
+ }
+
  minorversion_setup =  nfs_minorversion_callback_svc_setup(minorversion,
      serv, xprt, &rqstp, &callback_svc);
  if (!minorversion_setup) {
diff --git a/fs/nfsd/nfssvc.c b/fs/nfsd/nfssvc.c
index 28dfad3..a6461f3 100644
--- a/fs/nfsd/nfssvc.c
+++ b/fs/nfsd/nfssvc.c
@@ -11,6 +11,7 @@
 #include <linux/module.h>
 #include <linux/fs_struct.h>
 #include <linux/swap.h>
+#include <linux/nsproxy.h>

 #include <linux/sunrpc/stats.h>
 #include <linux/sunrpc/svcsock.h>
@@ -330,6 +331,8 @@ static int nfsd_get_default_max_blksize(void)

 int nfsd_create_serv(void)
 {
+ int error;
+
  WARN_ON(!mutex_is_locked(&nfsd_mutex));
  if (nfsd_serv) {
   svc_get(nfsd_serv);
@@ -343,6 +346,12 @@ int nfsd_create_serv(void)
```

```
  if (nfsd_serv == NULL)
   return -ENOMEM;

+ error = svc_bind(nfsd_serv, current->nsproxy->net_ns);
+ if (error < 0) {
+  svc_destroy(nfsd_serv);
+  return error;
+ }
+
  set_max_drc();
  do_gettimeofday(&nfssvc_boot);  /* record boot time */
  return 0;
diff --git a/include/linux/sunrpc/svc.h b/include/linux/sunrpc/svc.h
index 51b29ac..2b43e02 100644
--- a/include/linux/sunrpc/svc.h
+++ b/include/linux/sunrpc/svc.h
@@ -416,6 +416,7 @@ struct svc_procedure {
  */
 int svc_rpcb_setup(struct svc_serv *serv, struct net *net);
 void svc_rpcb_cleanup(struct svc_serv *serv, struct net *net);
+int svc_bind(struct svc_serv *serv, struct net *net);
 struct svc_serv *svc_create(struct svc_program *, unsigned int,
        void (*shutdown)(struct svc_serv *, struct net *net));
 struct svc_rqst *svc_prepare_thread(struct svc_serv *serv,
diff --git a/net/sunrpc/rpcb_clnt.c b/net/sunrpc/rpcb_clnt.c
index 78ac39f..4c38b33 100644
--- a/net/sunrpc/rpcb_clnt.c
+++ b/net/sunrpc/rpcb_clnt.c
@@ -180,14 +180,16 @@ void rpcb_put_local(struct net *net)
  struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
  struct rpc_clnt *clnt = sn->rpcb_local_clnt;
  struct rpc_clnt *clnt4 = sn->rpcb_local_clnt4;
- int shutdown;
+ int shutdown = 0;

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

```
  if (shutdown) {
diff --git a/net/sunrpc/svc.c b/net/sunrpc/svc.c
index 4153846..e6d542c 100644
--- a/net/sunrpc/svc.c
+++ b/net/sunrpc/svc.c
@@ -407,6 +407,14 @@ static int svc_uses_rpcbind(struct svc_serv *serv)
  return 0;
 }

+int svc_bind(struct svc_serv *serv, struct net *net)
+{
+ if (!svc_uses_rpcbind(serv))
+  return 0;
+ return svc_rpcb_setup(serv, net);
+}
+EXPORT_SYMBOL_GPL(svc_bind);
+
 /*
  * Create an RPC service
  */
@@ -471,15 +479,8 @@ __svc_create(struct svc_program *prog, unsigned int bufsize, int
npools,
   spin_lock_init(&pool->sp_lock);
  }

- if (svc_uses_rpcbind(serv)) {
-  if (svc_rpcb_setup(serv, current->nsproxy->net_ns) < 0) {
-   kfree(serv->sv_pools);
-   kfree(serv);
-   return NULL;
-  }
-  if (!serv->sv_shutdown)
-   serv->sv_shutdown = svc_rpcb_cleanup;
- }
+ if (svc_uses_rpcbind(serv) && (!serv->sv_shutdown))
+  serv->sv_shutdown = svc_rpcb_cleanup;

  return serv;
 }
```