Subject: [PATCH 3/3] lockd: create and use per-net NSM RPC clients on MON/UNMON requests
Posted by Stanislav Kinsbursky on Fri, 14 Sep 2012 14:26:05 GMT
View Forum Message <> Reply to Message

NSM RPC client can be required on NFSv3 umount, when child reaper is dying
(and destroying it's mount namespace). It means, that current nsproxy is set
to NULL already, but creation of RPC client requires UTS namespace for gaining
hostname string.

This patch creates reference-counted per-net NSM client on first monitor
request and destroys it after last unmonitor request.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
Cc: <stable@vger.kernel.org>
---
 fs/lockd/mon.c | 34 ++++++++++++++++++++++++++----------
 1 files changed, 19 insertions(+), 15 deletions(-)

```
diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
index 77e07fe..c233ed5 100644
--- a/fs/lockd/mon.c
+++ b/fs/lockd/mon.c
@@ -85,7 +85,7 @@ static struct rpc_clnt *nsm_create(struct net *net)
  return rpc_create(&args);
 }

-__maybe_unused static struct rpc_clnt *nsm_client_get(struct net *net)
+static struct rpc_clnt *nsm_client_get(struct net *net)
 {
  static DEFINE_MUTEX(nsm_create_mutex);
  struct rpc_clnt *clnt;
@@ -112,7 +112,7 @@ out:
  return clnt;
 }

-__maybe_unused static void nsm_client_put(struct net *net)
+static void nsm_client_put(struct net *net)
 {
  struct lockd_net *ln = net_generic(net, lockd_net_id);
  struct rpc_clnt *clnt = ln->nsm_clnt;
@@ -131,9 +131,8 @@ __maybe_unused static void nsm_client_put(struct net *net)
 }

 static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct nsm_res *res,
-    struct net *net)
+     struct rpc_clnt *clnt)
 {
```

```
- struct rpc_clnt *clnt;
  int  status;
  struct nsm_args args = {
   .priv  = &nsm->sm_priv,
@@ -147,13 +146,7 @@ static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct
nsm_res *res,
   .rpc_resp = res,
  };

- clnt = nsm_create(net);
- if (IS_ERR(clnt)) {
-  status = PTR_ERR(clnt);
-  dprintk("lockd: failed to create NSM upcall transport, "
-    "status=%d\n", status);
-  goto out;
- }
+ BUG_ON(clnt == NULL);

  memset(res, 0, sizeof(*res));

@@ -164,8 +157,6 @@ static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct
nsm_res *res,
     status);
  else
   status = 0;
- rpc_shutdown_client(clnt);
- out:
  return status;
 }

@@ -185,6 +176,7 @@ int nsm_monitor(const struct nlm_host *host)
  struct nsm_handle *nsm = host->h_nsmhandle;
  struct nsm_res res;
  int  status;
+ struct rpc_clnt *clnt;

  dprintk("lockd: nsm_monitor(%s)\n", nsm->sm_name);

@@ -197,7 +189,15 @@ int nsm_monitor(const struct nlm_host *host)
   */
  nsm->sm_mon_name = nsm_use_hostnames ? nsm->sm_name : nsm->sm_addrbuf;

- status = nsm_mon_unmon(nsm, NSMPROC_MON, &res, host->net);
+ clnt = nsm_client_get(host->net);
+ if (IS_ERR(clnt)) {
+  status = PTR_ERR(clnt);
+  dprintk("lockd: failed to create NSM upcall transport, "
+    "status=%d, net=%p\n", status, host->net);
```

```
+  return status;
+ }
+
+ status = nsm_mon_unmon(nsm, NSMPROC_MON, &res, clnt);
  if (unlikely(res.status != 0))
   status = -EIO;
  if (unlikely(status < 0)) {
@@ -229,9 +229,11 @@ void nsm_unmonitor(const struct nlm_host *host)

  if (atomic_read(&nsm->sm_count) == 1
   && nsm->sm_monitored && !nsm->sm_sticky) {
+  struct lockd_net *ln = net_generic(host->net, lockd_net_id);
+
   dprintk("lockd: nsm_unmonitor(%s)\n", nsm->sm_name);

-  status = nsm_mon_unmon(nsm, NSMPROC_UNMON, &res, host->net);
+  status = nsm_mon_unmon(nsm, NSMPROC_UNMON, &res, ln->nsm_clnt);
   if (res.status != 0)
    status = -EIO;
   if (status < 0)
@@ -239,6 +241,8 @@ void nsm_unmonitor(const struct nlm_host *host)
      nsm->sm_name);
   else
    nsm->sm_monitored = 0;
+
+  nsm_client_put(host->net);
  }
 }
```