Subject: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Stanislav Kinsbursky on Fri, 14 Sep 2012 14:25:47 GMT
View Forum Message <> Reply to Message

This is a bug fix for https://bugzilla.redhat.com/show_bug.cgi?id=830862.

The problem is that with NFSv4 mount in container (with separated mount
namesapce) and active lock on it, dying child reaped of this container will
try to umount NFS and doing this will try to create RPC client to send
unmonitor request to statd.
But creation of RCP client requires valid current->nsproxy (for operation with
utsname()) and during umount on child reaper exit it's equal to zero.

Proposed solution is to introduce refrence-counter per-net NSM client, which
is created on fist monitor call and destroyed after the lst monitor call.

The following series implements...

Stanislav Kinsbursky (3):
    lockd: use rpc client's cl_nodename for id encoding
    lockd: per-net NSM client creation and destruction helpers introduced
    lockd: create and use per-net NSM RPC clients on MON/UNMON requests

 fs/lockd/mon.c | 91 ++++++

```
--- a/fs/lockd/mon.c
+++ b/fs/lockd/mon.c
@@ -426,11 +426,12 @@ static void encode_mon_name(struct xdr_stream *xdr, const struct
nsm_args *argp)
  * (via the NLMPROC_SM_NOTIFY call) that the state of host "mon_name"
  * has changed.
  */
-static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp)
+static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp,
+    char *nodename)
 {
  __be32 *p;

- encode_nsm_string(xdr, utsname()->nodename);
+ encode_nsm_string(xdr, nodename);
  p = xdr_reserve_space(xdr, 4 + 4 + 4);
  *p++ = cpu_to_be32(argp->prog);
  *p++ = cpu_to_be32(argp->vers);
@@ -441,10 +442,11 @@ static void encode_my_id(struct xdr_stream *xdr, const struct
nsm_args *argp)
  * The "mon_id" argument specifies the non-private arguments
  * of an NSMPROC_MON or NSMPROC_UNMON call.
  */
-static void encode_mon_id(struct xdr_stream *xdr, const struct nsm_args *argp)
+static void encode_mon_id(struct xdr_stream *xdr, const struct nsm_args *argp,
+     char *nodename)
 {
  encode_mon_name(xdr, argp);
- encode_my_id(xdr, argp);
+ encode_my_id(xdr, argp, nodename);
 }

 /*
@@ -463,14 +465,14 @@ static void encode_priv(struct xdr_stream *xdr, const struct nsm_args
*argp)
 static void nsm_xdr_enc_mon(struct rpc_rqst *req, struct xdr_stream *xdr,
        const struct nsm_args *argp)
 {
- encode_mon_id(xdr, argp);
+ encode_mon_id(xdr, argp, req->rq_task->tk_client->cl_nodename);
  encode_priv(xdr, argp);
 }

 static void nsm_xdr_enc_unmon(struct rpc_rqst *req, struct xdr_stream *xdr,
         const struct nsm_args *argp)
 {
- encode_mon_id(xdr, argp);
+ encode_mon_id(xdr, argp, req->rq_task->tk_client->cl_nodename);
```

}

static int nsm_xdr_dec_stat_res(struct rpc_rqst *rqstp,

Subject: [PATCH 2/3] lockd: per-net NSM client creation and destruction helpers introduced
Posted by Stanislav Kinsbursky on Fri, 14 Sep 2012 14:26:00 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 introduces reference counted NFS RPC clients creation and
destruction helpers (similar to RPCBIND RPC clients).

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
Cc: <stable@vger.kernel.org>
---
fs/lockd/mon.c | 47 ++++++

```
+ spin_lock(&ln->nsm_clnt_lock);
+ if (ln->nsm_users) {
+  ln->nsm_users++;
+  clnt = ln->nsm_clnt;
+  spin_unlock(&ln->nsm_clnt_lock);
+  goto out;
+ }
+ spin_unlock(&ln->nsm_clnt_lock);
+
+ mutex_lock(&nsm_create_mutex);
+ clnt = nsm_create(net);
+ if (!IS_ERR(clnt)) {
+  ln->nsm_clnt = clnt;
+  smp_wmb();
+  ln->nsm_users = 1;
+ }
+ mutex_unlock(&nsm_create_mutex);
+out:
+ return clnt;
+}
+
+__maybe_unused 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;
+ int shutdown = 0;
+
+ spin_lock(&ln->nsm_clnt_lock);
+ if (ln->nsm_users) {
+  if (--ln->nsm_users)
+   ln->nsm_clnt = NULL;
+  shutdown = !ln->nsm_users;
+ }
+ spin_unlock(&ln->nsm_clnt_lock);
+
+ if (shutdown)
+  rpc_shutdown_client(clnt);
+}
+
 static int nsm_mon_unmon(struct nsm_handle *nsm, u32 proc, struct nsm_res *res,
     struct net *net)
 {
diff --git a/fs/lockd/netns.h b/fs/lockd/netns.h
index 4eee248..5010b55 100644
--- a/fs/lockd/netns.h
+++ b/fs/lockd/netns.h
@@ -12,6 +12,10 @@ struct lockd_net {
  struct delayed_work grace_period_end;
```

```
  struct lock_manager lockd_manager;
  struct list_head grace_list;
+
+ spinlock_t nsm_clnt_lock;
+ unsigned int nsm_users;
+ struct rpc_clnt *nsm_clnt;
 };

 extern int lockd_net_id;
diff --git a/fs/lockd/svc.c b/fs/lockd/svc.c
index 31a63f8..7e35587 100644
--- a/fs/lockd/svc.c
+++ b/fs/lockd/svc.c
@@ -596,6 +596,7 @@ static int lockd_init_net(struct net *net)

  INIT_DELAYED_WORK(&ln->grace_period_end, grace_ender);
  INIT_LIST_HEAD(&ln->grace_list);
+ spin_lock_init(&ln->nsm_clnt_lock);
  return 0;
 }
```

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);
```

```
 }
}
```

Subject: Re: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Chuck Lever on Fri, 14 Sep 2012 17:01:21 GMT
View Forum Message <> Reply to Message

What happens if statd is restarted?

Sent from my iPhonespam SPAMSPAM

On Sep 14, 2012, at 10:25 AM, Stanislav Kinsbursky <skinsbursky@parallels.com> wrote:

> This is a bug fix for https://bugzilla.redhat.com/show_bug.cgi?id=830862.
>
> The problem is that with NFSv4 mount in container (with separated mount
> namesapce) and active lock on it, dying child reaped of this container will
> try to umount NFS and doing this will try to create RPC client to send
> unmonitor request to statd.
> But creation of RCP client requires valid current->nsproxy (for operation with
> utsname()) and during umount on child reaper exit it's equal to zero.
>
> Proposed solution is to introduce refrence-counter per-net NSM client, which
> is created on fist monitor call and destroyed after the lst monitor call.
>
> The following series implements...
>
> ---
>
> Stanislav Kinsbursky (3):
> lockd: use rpc client's cl_nodename for id encoding
> lockd: per-net NSM client creation and destruction helpers introduced
> lockd: create and use per-net NSM RPC clients on MON/UNMON requests
>
>
> fs/lockd/mon.c | 91 ++----------
> fs/lockd/netns.h | 4 ++
> fs/lockd/svc.c | 1 +
> 3 files changed, 77 insertions(+), 19 deletions(-)
>
>
> --
> To unsubscribe from this list: send the line "unsubscribe linux-nfs" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

Subject: Re: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Myklebust, Trond on Fri, 14 Sep 2012 17:38:18 GMT
View Forum Message <> Reply to Message

On Fri, 2012-09-14 at 13:01 -0400, Chuck Lever wrote:
> What happens if statd is restarted?

Nothing unusual. Why?

> Sent from my iPhonespam SPAMSPAM
>
> On Sep 14, 2012, at 10:25 AM, Stanislav Kinsbursky <skinsbursky@parallels.com> wrote:
>
> > This is a bug fix for https://bugzilla.redhat.com/show_bug.cgi?id=830862.
> >
> > The problem is that with NFSv4 mount in container (with separated mount
> > namesapce) and active lock on it, dying child reaped of this container will
> > try to umount NFS and doing this will try to create RPC client to send
> > unmonitor request to statd.
> > But creation of RCP client requires valid current->nsproxy (for operation with
> > utsname()) and during umount on child reaper exit it's equal to zero.
> >
> > Proposed solution is to introduce refrence-counter per-net NSM client, which
> > is created on fist monitor call and destroyed after the lst monitor call.
> >
> > The following series implements...
> >
> > ---
> >
> > Stanislav Kinsbursky (3):
> >   lockd: use rpc client's cl_nodename for id encoding
> >   lockd: per-net NSM client creation and destruction helpers introduced
> >   lockd: create and use per-net NSM RPC clients on MON/UNMON requests
> >
> >
> > fs/lockd/mon.c | 91 ++++++

NetApp
Trond.Myklebust@netapp.com
www.netapp.com

Subject: Re: [PATCH 1/3] lockd: use rpc client's cl_nodename for id encoding
Posted by Chuck Lever on Fri, 14 Sep 2012 19:08:42 GMT
View Forum Message <> Reply to Message

Hi-

On Sep 14, 2012, at 10:25 AM, Stanislav Kinsbursky wrote:

> Taking hostname from uts namespace if not safe, because this cuold be
> performind during umount operation on child reaper death. And in this case
> current->nsproxy is NULL already.
>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
> Cc: <stable@vger.kernel.org>
> ---
> fs/lockd/mon.c | 14 ++++++++------
> 1 files changed, 8 insertions(+), 6 deletions(-)
>
> diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
> index 7ef14b3..c6186fb 100644
> --- a/fs/lockd/mon.c
> +++ b/fs/lockd/mon.c
> @@ -426,11 +426,12 @@ static void encode_mon_name(struct xdr_stream *xdr, const struct nsm_args *argp)
> * (via the NLMPROC_SM_NOTIFY call) that the state of host "mon_name"
> * has changed.
> */
> -static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp)
> +static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp,
> + char *nodename)
> {
> __be32 *p;
>
> - encode_nsm_string(xdr, utsname()->nodename);
> + encode_nsm_string(xdr, nodename);
> p = xdr_reserve_space(xdr, 4 + 4 + 4);
> *p++ = cpu_to_be32(argp->prog);
> *p++ = cpu_to_be32(argp->vers);
> @@ -441,10 +442,11 @@ static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp)
> * The "mon_id" argument specifies the non-private arguments
> * of an NSMPROC_MON or NSMPROC_UNMON call.

> */
> -static void encode_mon_id(struct xdr_stream *xdr, const struct nsm_args *argp)
> +static void encode_mon_id(struct xdr_stream *xdr, const struct nsm_args *argp,
> +   char *nodename)
> {
> encode_mon_name(xdr, argp);
> - encode_my_id(xdr, argp);
> + encode_my_id(xdr, argp, nodename);
> }
>
> /*
> @@ -463,14 +465,14 @@ static void encode_priv(struct xdr_stream *xdr, const struct
nsm_args *argp)
> static void nsm_xdr_enc_mon(struct rpc_rqst *req, struct xdr_stream *xdr,
>   const struct nsm_args *argp)
> {
> - encode_mon_id(xdr, argp);
> + encode_mon_id(xdr, argp, req->rq_task->tk_client->cl_nodename);

IMO you should get the cl_nodename in nsm_mon_unmon() from clnt->cl_nodename, and pass it in as part of *argp . The choice of which nodename to use is clearly a decision for an "upper layer" not a choice for the XDR functions.

Long ago I had patches which fixed this layering violation for a very similar purpose as yours, but they were never applied.

> encode_priv(xdr, argp);
> }
>
> static void nsm_xdr_enc_unmon(struct rpc_rqst *req, struct xdr_stream *xdr,
>   const struct nsm_args *argp)
> {
> - encode_mon_id(xdr, argp);
> + encode_mon_id(xdr, argp, req->rq_task->tk_client->cl_nodename);

Ditto.

> }
>
> static int nsm_xdr_dec_stat_res(struct rpc_rqst *rqstp,
>
> --
> To unsubscribe from this list: send the line "unsubscribe linux-nfs" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html

--
Chuck Lever

Subject: Re: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Chuck Lever on Fri, 14 Sep 2012 19:10:02 GMT
View Forum Message <> Reply to Message

On Sep 14, 2012, at 1:38 PM, Myklebust, Trond wrote:

> On Fri, 2012-09-14 at 13:01 -0400, Chuck Lever wrote:
>> What happens if statd is restarted?
>
> Nothing unusual. Why?

The NSM upcall transport is a potential application for TCP + softconn, now that a persistent rpc_clnt is used. It just depends on what failure mode we'd like to optimize for.

>
>> Sent from my iPhonespam SPAMSPAM
>>
>> On Sep 14, 2012, at 10:25 AM, Stanislav Kinsbursky <skinsbursky@parallels.com> wrote:
>>
>>> This is a bug fix for https://bugzilla.redhat.com/show_bug.cgi?id=830862.
>>>
>>> The problem is that with NFSv4 mount in container (with separated mount
>>> namesapce) and active lock on it, dying child reaped of this container will
>>> try to umount NFS and doing this will try to create RPC client to send
>>> unmonitor request to statd.
>>> But creation of RCP client requires valid current->nsproxy (for operation with
>>> utsname()) and during umount on child reaper exit it's equal to zero.
>>>
>>> Proposed solution is to introduce refrence-counter per-net NSM client, which
>>> is created on fist monitor call and destroyed after the lst monitor call.
>>>
>>> The following series implements...
>>>
>>> ---
>>>
>>> Stanislav Kinsbursky (3):
>>>   lockd: use rpc client's cl_nodename for id encoding
>>>   lockd: per-net NSM client creation and destruction helpers introduced
>>>   lockd: create and use per-net NSM RPC clients on MON/UNMON requests
>>>
>>>
>>> fs/lockd/mon.c | 91 ++++++

>>>
>>>
>>> --
>>> To unsubscribe from this list: send the line "unsubscribe linux-nfs" in
>>> the body of a message to majordomo@vger.kernel.org
>>> More majordomo info at http://vger.kernel.org/majordomo-info.html
>
> --
> Trond Myklebust
> Linux NFS client maintainer
>
> NetApp
> Trond.Myklebust@netapp.com
> www.netapp.com

--
Chuck Lever
chuck[dot]lever[at]oracle[dot]com

Subject: Re: [PATCH 1/3] lockd: use rpc client's cl_nodename for id encoding
Posted by Stanislav Kinsbursky on Mon, 17 Sep 2012 10:48:05 GMT
View Forum Message <> Reply to Message

> Hi-
>
> On Sep 14, 2012, at 10:25 AM, Stanislav Kinsbursky wrote:
>
>> Taking hostname from uts namespace if not safe, because this cuold be
>> performind during umount operation on child reaper death. And in this case
>> current->nsproxy is NULL already.
>>
>> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>> Cc: <stable@vger.kernel.org>
>> ---
>> fs/lockd/mon.c | 14 ++++++++------
>> 1 files changed, 8 insertions(+), 6 deletions(-)
>>
>> diff --git a/fs/lockd/mon.c b/fs/lockd/mon.c
>> index 7ef14b3..c6186fb 100644
>> --- a/fs/lockd/mon.c
>> +++ b/fs/lockd/mon.c
>> @@ -426,11 +426,12 @@ static void encode_mon_name(struct xdr_stream *xdr, const struct
nsm_args *argp)
>>   * (via the NLMPROC_SM_NOTIFY call) that the state of host "mon_name"
>>   * has changed.
>>   */

>> -static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp)
>> +static void encode_my_id(struct xdr_stream *xdr, const struct nsm_args *argp,
>> +   char *nodename)
>> {
>> __be32 *p;
>>
>> - encode_nsm_string(xdr, utsname()->nodename);
>> + encode_nsm_string(xdr, nodename);
>> p = xdr_reserve_space(xdr, 4 + 4 + 4);
>> *p++ = cpu_to_be32(argp->prog);
>> *p++ = cpu_to_be32(argp->vers);
>> @@ -441,10 +442,11 @@ static void encode_my_id(struct xdr_stream *xdr, const struct
nsm_args *argp)
>>   * The "mon_id" argument specifies the non-private arguments
>>   * of an NSMPROC_MON or NSMPROC_UNMON call.
>>   */
>> -static void encode_mon_id(struct xdr_stream *xdr, const struct nsm_args *argp)
>> +static void encode_mon_id(struct xdr_stream *xdr, const struct nsm_args *argp,
>> +   char *nodename)
>> {
>> encode_mon_name(xdr, argp);
>> - encode_my_id(xdr, argp);
>> + encode_my_id(xdr, argp, nodename);
>> }
>>
>> /*
>> @@ -463,14 +465,14 @@ static void encode_priv(struct xdr_stream *xdr, const struct
nsm_args *argp)
>> static void nsm_xdr_enc_mon(struct rpc_rqst *req, struct xdr_stream *xdr,
>>   const struct nsm_args *argp)
>> {
>> - encode_mon_id(xdr, argp);
>> + encode_mon_id(xdr, argp, req->rq_task->tk_client->cl_nodename);
>
> IMO you should get the cl_nodename in nsm_mon_unmon() from clnt->cl_nodename, and pass it in as part of *argp . The choice of which nodename to use is clearly a decision for an "upper layer" not a choice for the XDR functions.
>
> Long ago I had patches which fixed this layering violation for a very similar purpose as yours, but they were never applied.
>

I like it. Thanks.

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Stanislav Kinsbursky on Mon, 17 Sep 2012 10:49:54 GMT
View Forum Message <> Reply to Message

>
> On Sep 14, 2012, at 1:38 PM, Myklebust, Trond wrote:
>
>> On Fri, 2012-09-14 at 13:01 -0400, Chuck Lever wrote:
>>> What happens if statd is restarted?
>>
>> Nothing unusual. Why?
>
> The NSM upcall transport is a potential application for TCP + softconn, now that a persistent rpc_clnt is used. It just depends on what failure mode we'd like to optimize for.
>

I don't understand, where the problem is.
Could you be more specific, please?

--
Best regards,
Stanislav Kinsbursky

Subject: Re: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Chuck Lever on Mon, 17 Sep 2012 15:10:30 GMT
View Forum Message <> Reply to Message

On Sep 17, 2012, at 6:49 AM, Stanislav Kinsbursky wrote:

>>
>> On Sep 14, 2012, at 1:38 PM, Myklebust, Trond wrote:
>>
>>> On Fri, 2012-09-14 at 13:01 -0400, Chuck Lever wrote:
>>>> What happens if statd is restarted?
>>>
>>> Nothing unusual. Why?
>>
>> The NSM upcall transport is a potential application for TCP + softconn, now that a persistent rpc_clnt is used. It just depends on what failure mode we'd like to optimize for.
>>
>
> I don't understand, where the problem is.
> Could you be more specific, please?

I'm suggesting an enhancement.

The change is to use TCP for the NSM upcall transport, and set RPC_TASK_SOFTCONN on the individual RPCs. The advantage of this is that the kernel could discover when statd is not running and fail the upcall immediately, rather than waiting possibly many seconds for each upcall RPC to time out.

The client already has a check in the mount.nfs command to see that statd is running, likely to avoid this lengthly timeout. Since the client already has long-standing logic to avoid it, I think the benefit would be mostly on the server side.

But this change can be done at some later point.

--
Chuck Lever
chuck[dot]lever[at]oracle[dot]com

Subject: Re: [PATCH 0/3] lockd: use per-net refrence-counted NSM clients
Posted by Stanislav Kinsbursky on Mon, 17 Sep 2012 15:23:49 GMT
View Forum Message <> Reply to Message

>
> On Sep 17, 2012, at 6:49 AM, Stanislav Kinsbursky wrote:
>

>>>
>>> On Sep 14, 2012, at 1:38 PM, Myklebust, Trond wrote:
>>>
>>>> On Fri, 2012-09-14 at 13:01 -0400, Chuck Lever wrote:
>>>>> What happens if statd is restarted?
>>>>
>>>> Nothing unusual. Why?
>>>
>>> The NSM upcall transport is a potential application for TCP + softconn, now that a persistent rpc_clnt is used. It just depends on what failure mode we'd like to optimize for.
>>>
>>
>> I don't understand, where the problem is.
>> Could you be more specific, please?
>
> I'm suggesting an enhancement.
>
> The change is to use TCP for the NSM upcall transport, and set RPC_TASK_SOFTCONN on the individual RPCs. The advantage of this is that the kernel could discover when statd is not running and fail the upcall immediately, rather than waiting possibly many seconds for each upcall RPC to time out.
>

> The client already has a check in the mount.nfs command to see that statd is running, likely to avoid this lengthly timeout. Since the client already has long-standing logic to avoid it, I think the benefit would be mostly on the server side.
>
> But this change can be done at some later point.
>

Ok, thanks.
Sounds reasonable to me.
I'll do so.

--
Best regards,
Stanislav Kinsbursky
