
Subject: [PATCH v2] NFSd: fix locking in nfsd_forget_delegations()
Posted by [Stanislav Kinsbursky](#) on Fri, 25 May 2012 14:38:50 GMT
[View Forum Message](#) <> [Reply to Message](#)

v2: dl_recall_lru list is used for delegations collect because it's modified both in unhash_delegation() and nfsd_break_one_deleg().

This patch adds recall_lock hold to nfsd_forget_delegations() to protect nfsd_process_n_delegations() call.

Also, looks like it would be better to collect delegations to some local on-stack list, and then unhash collected list. This split allows to simplify locking, because delegation traversing is protected by recall_lock, when delegation unhash is protected by client_mutex.

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

```
---
fs/nfsd/nfs4state.c | 21 ++++++++-----
1 files changed, 17 insertions(+), 4 deletions(-)

diff --git a/fs/nfsd/nfs4state.c b/fs/nfsd/nfs4state.c
index 21266c7..3d6848b 100644
--- a/fs/nfsd/nfs4state.c
+++ b/fs/nfsd/nfs4state.c
@@ -4694,7 +4694,7 @@ void nfsd_forget_openowners(u64 num)
    printk(KERN_INFO "NFSD: Forgot %d open owners", count);
}

-int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct nfs4_delegation *))
+int nfsd_process_n_delegations(u64 num, struct list_head *list)
{
    int i, count = 0;
    struct nfs4_file *fp, *fnext;
@@ -4703,7 +4703,7 @@ int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct
nfs4_delegatio
    for (i = 0; i < FILE_HASH_SIZE; i++) {
        list_for_each_entry_safe(fp, fnext, &file_hashtbl[i], fi_hash) {
            list_for_each_entry_safe(dp, dnext, &fp->fi_delegations, dl_perfile) {
-                deleg_func(dp);
+                list_move(&dp->dl_recall_lru, list);
                if (++count == num)
                    return count;
            }
        }
@@ -4716,9 +4716,16 @@ int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct
nfs4_delegatio
    void nfsd_forget_delegations(u64 num)
    {
        unsigned int count;
+        LIST_HEAD(victims);
```

```

+ struct nfs4_delegation *dp, *dnext;
+
+ spin_lock(&recall_lock);
+ count = nfsd_process_n_delegations(num, &victims);
+ spin_unlock(&recall_lock);

nfs4_lock_state();
- count = nfsd_process_n_delegations(num, unhash_delegation);
+ list_for_each_entry_safe(dp, dnext, &victims, dl_recall_lru)
+ unhash_delegation(dp);
nfs4_unlock_state();

printk(KERN_INFO "NFSD: Forgot %d delegations", count);
@@ -4727,10 +4734,16 @@ void nfsd_forget_delegations(u64 num)
void nfsd_recall_delegations(u64 num)
{
    unsigned int count;
+ LIST_HEAD(victims);
+ struct nfs4_delegation *dp, *dnext;

nfs4_lock_state();
spin_lock(&recall_lock);
- count = nfsd_process_n_delegations(num, nfsd_break_one_deleg);
+ count = nfsd_process_n_delegations(num, &victims);
+ list_for_each_entry_safe(dp, dnext, &victims, dl_recall_lru) {
+ list_del(&dp->dl_recall_lru);
+ nfsd_break_one_deleg(dp);
+ }
spin_unlock(&recall_lock);
nfs4_unlock_state();

```

Subject: Re: [PATCH v2] NFSD: fix locking in nfsd_forget_delegations()

Posted by [bfields](#) on Thu, 12 Jul 2012 15:43:31 GMT

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

On Fri, May 25, 2012 at 06:38:50PM +0400, Stanislav Kinsbursky wrote:

> v2: dl_recall_lru list is used for delegations collect because it's modified

> both in unhash_delegation() and nfsd_break_one_deleg().

>

> This patch adds recall_lock hold to nfsd_forget_delegations() to protect

> nfsd_process_n_delegations() call.

> Also, looks like it would be better to collect delegations to some local

> on-stack list, and then unhash collected list. This split allows to

> simplify locking, because delegation traversing is protected by recall_lock,

> when delegation unhash is protected by client_mutex.

Thanks, applying for 3.6 (assuming Bryan doesn't have any objection).

--b.

```
>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
> ---
> fs/nfsd/nfs4state.c | 21 ++++++-----
> 1 files changed, 17 insertions(+), 4 deletions(-)
>
> diff --git a/fs/nfsd/nfs4state.c b/fs/nfsd/nfs4state.c
> index 21266c7..3d6848b 100644
> --- a/fs/nfsd/nfs4state.c
> +++ b/fs/nfsd/nfs4state.c
> @@ -4694,7 +4694,7 @@ void nfsd_forget_openowners(u64 num)
>   printk(KERN_INFO "NFSD: Forgot %d open owners", count);
> }
>
> -int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct nfs4_delegation *))
> +int nfsd_process_n_delegations(u64 num, struct list_head *list)
> {
>   int i, count = 0;
>   struct nfs4_file *fp, *fnext;
> @@ -4703,7 +4703,7 @@ int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct
nfs4_delegatio
>   for (i = 0; i < FILE_HASH_SIZE; i++) {
>     list_for_each_entry_safe(fp, fnext, &file_hashtbl[i], fi_hash) {
>       list_for_each_entry_safe(dp, dnext, &fp->fi_delegations, dl_perfile) {
> -       deleg_func(dp);
> +       list_move(&dp->dl_recall_lru, list);
>       if (++count == num)
>         return count;
>     }
> @@ -4716,9 +4716,16 @@ int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct
nfs4_delegatio
>   void nfsd_forget_delegations(u64 num)
>   {
>     unsigned int count;
>     + LIST_HEAD(victims);
>     + struct nfs4_delegation *dp, *dnext;
>     +
>     + spin_lock(&recall_lock);
>     + count = nfsd_process_n_delegations(num, &victims);
>     + spin_unlock(&recall_lock);
>
>     + nfs4_lock_state();
>     - count = nfsd_process_n_delegations(num, unhash_delegation);
>     + list_for_each_entry_safe(dp, dnext, &victims, dl_recall_lru)
>     +   unhash_delegation(dp);
```

```

> nfs4_unlock_state();
>
> printk(KERN_INFO "NFSD: Forgot %d delegations", count);
> @@ -4727,10 +4734,16 @@ void nfsd_forget_delegations(u64 num)
> void nfsd_recall_delegations(u64 num)
> {
> unsigned int count;
> + LIST_HEAD(victims);
> + struct nfs4_delegation *dp, *dnext;
>
> nfs4_lock_state();
> spin_lock(&recall_lock);
> - count = nfsd_process_n_delegations(num, nfsd_break_one_deleg);
> + count = nfsd_process_n_delegations(num, &victims);
> + list_for_each_entry_safe(dp, dnext, &victims, dl_recall_lru) {
> + list_del(&dp->dl_recall_lru);
> + nfsd_break_one_deleg(dp);
> + }
> spin_unlock(&recall_lock);
> nfs4_unlock_state();
>
>
> --
> 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 v2] NFSD: fix locking in nfsd_forget_delegations()

Posted by [Bryan Schumaker](#) on Thu, 12 Jul 2012 16:02:38 GMT

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

On 07/12/2012 11:43 AM, J. Bruce Fields wrote:

```

> On Fri, May 25, 2012 at 06:38:50PM +0400, Stanislav Kinsbursky wrote:
>> v2: dl_recall_lru list is used for delegations collect because it's modified
>> both in unhash_delegation() and nfsd_break_one_deleg().
>>
>> This patch adds recall_lock hold to nfsd_forget_delegations() to protect
>> nfsd_process_n_delegations() call.
>> Also, looks like it would be better to collect delegations to some local
>> on-stack list, and then unhash collected list. This split allows to
>> simplify locking, because delegation traversing is protected by recall_lock,
>> when delegation unhash is protected by client_mutex.
>
> Thanks, applying for 3.6 (assuming Bryan doesn't have any objection).

```

Nope, I don't have any objections.

- Bryan

```
>
> --b.
>
>>
>> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>> ---
>> fs/nfsd/nfs4state.c | 21 ++++++++-----
>> 1 files changed, 17 insertions(+), 4 deletions(-)
>>
>> diff --git a/fs/nfsd/nfs4state.c b/fs/nfsd/nfs4state.c
>> index 21266c7..3d6848b 100644
>> --- a/fs/nfsd/nfs4state.c
>> +++ b/fs/nfsd/nfs4state.c
>> @@ -4694,7 +4694,7 @@ void nfsd_forget_openowners(u64 num)
>>  printk(KERN_INFO "NFSD: Forgot %d open owners", count);
>> }
>>
>> -int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct nfs4_delegation *))
>> +int nfsd_process_n_delegations(u64 num, struct list_head *list)
>> {
>>  int i, count = 0;
>>  struct nfs4_file *fp, *fnext;
>> @@ -4703,7 +4703,7 @@ int nfsd_process_n_delegations(u64 num, void (*deleg_func)(struct
nfs4_delegatio
>>  for (i = 0; i < FILE_HASH_SIZE; i++) {
>>  list_for_each_entry_safe(fp, fnext, &file_hashtbl[i], fi_hash) {
>>  list_for_each_entry_safe(dp, dnext, &fp->fi_delegations, dl_perfile) {
>> -   deleg_func(dp);
>> +   list_move(&dp->dl_recall_lru, list);
>>  if (++count == num)
>>  return count;
>>  }
>> @@ -4716,9 +4716,16 @@ int nfsd_process_n_delegations(u64 num, void
(*deleg_func)(struct nfs4_delegatio
>>  void nfsd_forget_delegations(u64 num)
>>  {
>>  unsigned int count;
>>  + LIST_HEAD(victims);
>>  + struct nfs4_delegation *dp, *dnext;
>>  +
>>  + spin_lock(&recall_lock);
>>  + count = nfsd_process_n_delegations(num, &victims);
>>  + spin_unlock(&recall_lock);
>>
>>  nfs4_lock_state();
>>  - count = nfsd_process_n_delegations(num, unhash_delegation);
```

```

>> + list_for_each_entry_safe(dp, dnext, &victims, dl_recall_lru)
>> + unhash_delegation(dp);
>> nfs4_unlock_state();
>>
>> printk(KERN_INFO "NFSD: Forgot %d delegations", count);
>> @@ -4727,10 +4734,16 @@ void nfsd_forget_delegations(u64 num)
>> void nfsd_recall_delegations(u64 num)
>> {
>> unsigned int count;
>> + LIST_HEAD(victims);
>> + struct nfs4_delegation *dp, *dnext;
>>
>> nfs4_lock_state();
>> spin_lock(&recall_lock);
>> - count = nfsd_process_n_delegations(num, nfsd_break_one_deleg);
>> + count = nfsd_process_n_delegations(num, &victims);
>> + list_for_each_entry_safe(dp, dnext, &victims, dl_recall_lru) {
>> + list_del(&dp->dl_recall_lru);
>> + nfsd_break_one_deleg(dp);
>> + }
>> spin_unlock(&recall_lock);
>> nfs4_unlock_state();
>>
>>
>> --
>> 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

```
