author	Dominique Martinet <asmadeus@codewreck.org>	2022-09-04 20:17:49 +0900
committer	Greg Kroah-Hartman <gregkh@linuxfoundation.org>	2022-11-26 09:24:51 +0100
commit	43bbadb7e4636dc02f6a283c2a39e6438e6173cd (patch)	
tree	a427a7ef63eba97c7fffab57ca08e812f4291af7	
parent	9357fca9dad7e76994dec4c3c997269997c94101 (diff)	
download	linux-43bbadb7e4636dc02f6a283c2a39e6438e6173cd.tar.gz	

net/9p: use a dedicated spinlock for trans_fd

```
commit 296ab4a813841ba1d5f40b03190fd1bd8f25aab0 upstream.

Shamelessly copying the explanation from Tetsuo Handa's suggested
patch[1] (slightly reworded):
syzbot is reporting inconsistent lock state in p9_req_put()[2],
for p9_tag_remove() from p9_req_put() from IRQ context is using
spin_lock_irqsave() on "struct p9_client"->lock but trans_fd
(not from IRQ context) is using spin_lock().

Since the locks actually protect different things in client.c and in
trans_fd.c, just replace trans_fd.c's lock by a new one specific to the
transport (client.c's protect the idr for fid/tag allocations,
while trans_fd.c's protects its own req list and request status field
that acts as the transport's state machine)

Link: https://lore.kernel.org/r/20220904112928.1308799-1-asmadeus@codewreck.org
Link: https://lkml.kernel.org/r/2470e028-9b05-2013-7198-1fdad071d999@I-love.SAKURA.ne.jp [1]
Link: https://syzkaller.appspot.com/bug?extid=2f20b523930c32c160cc [2]
Reported-by: syzbot <syzbot+2f20b523930c32c160cc@syzkaller.appspotmail.com>
Reported-by: Tetsuo Handa <penguin-kernel@I-love.SAKURA.ne.jp>
Reviewed-by: Christian Schoenebeck <linux_oss@crudebyte.com>
Signed-off-by: Dominique Martinet <asmadeus@codewreck.org>
Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
```

Diffstat

-rw-r--r-- net/9p/trans_fd.c 41

1 files changed, 25 insertions, 16 deletions

```
diff --git a/net/9p/trans_fd.c b/net/9p/trans_fd.c
index dfce201d988418..a8c1f742148cba 100644
--- a/net/9p/trans_fd.c
+++ b/net/9p/trans_fd.c
@@ -93,6 +93,7 @@ struct p9_poll_wait {
  * @mux_list: list link for mux to manage multiple connections (?)
  * @client: reference to client instance for this connection
  * @err: error state
+ * @req_lock: lock protecting req_list and requests statuses
  * @req_list: accounting for requests which have been sent
  * @unsent_req_list: accounting for requests that haven't been sent
  * @rreq: read request
@@ -116,6 +117,7 @@ struct p9_conn {
```

```
        struct list_head mux_list;
        struct p9_client *client;
        int err;
+       spinlock_t req_lock;
        struct list_head req_list;
        struct list_head unsent_req_list;
        struct p9_req_t *rreq;
@@ -191,10 +193,10 @@ static void p9_conn_cancel(struct p9_conn *m, int err)

        p9_debug(P9_DEBUG_ERROR, "mux %p err %d\n", m, err);

-       spin_lock(&m->client->lock);
+       spin_lock(&m->req_lock);

        if (m->err) {
-               spin_unlock(&m->client->lock);
+               spin_unlock(&m->req_lock);
                return;
        }

@@ -207,7 +209,7 @@ static void p9_conn_cancel(struct p9_conn *m, int err)
                list_move(&req->req_list, &cancel_list);
        }

-       spin_unlock(&m->client->lock);
+       spin_unlock(&m->req_lock);

        list_for_each_entry_safe(req, rtmp, &cancel_list, req_list) {
                p9_debug(P9_DEBUG_ERROR, "call back req %p\n", req);
@@ -362,7 +364,7 @@ static void p9_read_work(struct work_struct *work)
        if ((m->rreq) && (m->rc.offset == m->rc.capacity)) {
                p9_debug(P9_DEBUG_TRANS, "got new packet\n");
                m->rreq->rc.size = m->rc.offset;
-               spin_lock(&m->client->lock);
+               spin_lock(&m->req_lock);
                if (m->rreq->status == REQ_STATUS_SENT) {
                        list_del(&m->rreq->req_list);
                        p9_client_cb(m->client, m->rreq, REQ_STATUS_RCVD);
@@ -371,14 +373,14 @@ static void p9_read_work(struct work_struct *work)
                        p9_debug(P9_DEBUG_TRANS,
                                 "Ignore replies associated with a cancelled request\n");
                } else {
-                       spin_unlock(&m->client->lock);
+                       spin_unlock(&m->req_lock);
                        p9_debug(P9_DEBUG_ERROR,
                                 "Request tag %d errored out while we were reading the reply\n",
                                 m->rc.tag);
                        err = -EIO;
                        goto error;
                }
-               spin_unlock(&m->client->lock);
+               spin_unlock(&m->req_lock);
                m->rc.sdata = NULL;
                m->rc.offset = 0;
                m->rc.capacity = 0;
@@ -456,10 +458,10 @@ static void p9_write_work(struct work_struct *work)
        }

        if (!m->wsize) {
-               spin_lock(&m->client->lock);
+               spin_lock(&m->req_lock);
                if (list_empty(&m->unsent_req_list)) {
```

```
                        clear_bit(Wworksched, &m->wsched);
-                       spin_unlock(&m->client->lock);
+                       spin_unlock(&m->req_lock);
                        return;
                }

@@ -474,7 +476,7 @@ static void p9_write_work(struct work_struct *work)
                m->wpos = 0;
                p9_req_get(req);
                m->wreq = req;
-               spin_unlock(&m->client->lock);
+               spin_unlock(&m->req_lock);
        }

        p9_debug(P9_DEBUG_TRANS, "mux %p pos %d size %d\n",
@@ -591,6 +593,7 @@ static void p9_conn_create(struct p9_client *client)
        INIT_LIST_HEAD(&m->mux_list);
        m->client = client;

+       spin_lock_init(&m->req_lock);
        INIT_LIST_HEAD(&m->req_list);
        INIT_LIST_HEAD(&m->unsent_req_list);
        INIT_WORK(&m->rq, p9_read_work);
@@ -672,10 +675,10 @@ static int p9_fd_request(struct p9_client *client, struct p9_req_t *req)
        if (m->err < 0)
                return m->err;

-       spin_lock(&client->lock);
+       spin_lock(&m->req_lock);
        req->status = REQ_STATUS_UNSENT;
        list_add_tail(&req->req_list, &m->unsent_req_list);
-       spin_unlock(&client->lock);
+       spin_unlock(&m->req_lock);

        if (test_and_clear_bit(Wpending, &m->wsched))
                n = EPOLLOUT;
@@ -690,11 +693,13 @@ static int p9_fd_request(struct p9_client *client, struct p9_req_t *req)

 static int p9_fd_cancel(struct p9_client *client, struct p9_req_t *req)
 {
+       struct p9_trans_fd *ts = client->trans;
+       struct p9_conn *m = &ts->conn;
        int ret = 1;

        p9_debug(P9_DEBUG_TRANS, "client %p req %p\n", client, req);

-       spin_lock(&client->lock);
+       spin_lock(&m->req_lock);

        if (req->status == REQ_STATUS_UNSENT) {
                list_del(&req->req_list);
@@ -702,21 +707,24 @@ static int p9_fd_cancel(struct p9_client *client, struct p9_req_t *req)
                p9_req_put(client, req);
                ret = 0;
        }
-       spin_unlock(&client->lock);
+       spin_unlock(&m->req_lock);

        return ret;
 }

 static int p9_fd_cancelled(struct p9_client *client, struct p9_req_t *req)
```

```
 {
+	struct p9_trans_fd *ts = client->trans;
+	struct p9_conn *m = &ts->conn;
+
 	p9_debug(P9_DEBUG_TRANS, "client %p req %p\n", client, req);

-	spin_lock(&client->lock);
+	spin_lock(&m->req_lock);
 	/* Ignore cancelled request if message has been received
 	 * before lock.
 	 */
 	if (req->status == REQ_STATUS_RCVD) {
-		spin_unlock(&client->lock);
+		spin_unlock(&m->req_lock);
 		return 0;
 	}

@@ -725,7 +733,8 @@ static int p9_fd_cancelled(struct p9_client *client, struct p9_req_t *req)
 	 */
 	list_del(&req->req_list);
 	req->status = REQ_STATUS_FLSHD;
-	spin_unlock(&client->lock);
+	spin_unlock(&m->req_lock);
+
 	p9_req_put(client, req);

 	return 0;
```