Subject: [PATCH 1/2][INET] (resend) Fix potential kfree on vmalloc-ed area of request_sock_queue
Posted by Pavel Emelianov on Thu, 15 Nov 2007 08:41:37 GMT
View Forum Message <> Reply to Message

The request_sock_queue's listen_opt is either vmalloc-ed or
kmalloc-ed depending on the number of table entries. Thus it
is expected to be handled properly on free, which is done in
the reqsk_queue_destroy().

However the error path in inet_csk_listen_start() calls
the lite version of reqsk_queue_destroy, called
__reqsk_queue_destroy, which calls the kfree unconditionally.

Fix this and move the __reqsk_queue_destroy into a .c file as
it looks too big to be inline.

As David also noticed, this is an error recovery path only,
so no locking is required and the lopt is known to be not NULL.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>

---

```
diff --git a/include/net/request_sock.h b/include/net/request_sock.h
index 7aed02c..0a954ee 100644
--- a/include/net/request_sock.h
+++ b/include/net/request_sock.h
@@ -136,11 +136,7 @@ static inline struct listen_sock *reqsk_queue_yank_listen_sk(struct
request_sock
  return lopt;
 }

-static inline void __reqsk_queue_destroy(struct request_sock_queue *queue)
-{
- kfree(reqsk_queue_yank_listen_sk(queue));
-}
-
+extern void __reqsk_queue_destroy(struct request_sock_queue *queue);
 extern void reqsk_queue_destroy(struct request_sock_queue *queue);

 static inline struct request_sock *
diff --git a/net/core/request_sock.c b/net/core/request_sock.c
index 5f0818d..dd78b85 100644
--- a/net/core/request_sock.c
+++ b/net/core/request_sock.c
@@ -71,6 +71,28 @@ int reqsk_queue_alloc(struct request_sock_queue *queue,
```

```
 EXPORT_SYMBOL(reqsk_queue_alloc);

+void __reqsk_queue_destroy(struct request_sock_queue *queue)
+{
+ struct listen_sock *lopt;
+ size_t lopt_size;
+
+ /*
+  * this is an error recovery path only
+  * no locking needed and the lopt is not NULL
+  */
+
+ lopt = queue->listen_opt;
+ lopt_size = sizeof(struct listen_sock) +
+  lopt->nr_table_entries * sizeof(struct request_sock *);
+
+ if (lopt_size > PAGE_SIZE)
+  vfree(lopt);
+ else
+  kfree(lopt);
+}
+
+EXPORT_SYMBOL(__reqsk_queue_destroy);
+
 void reqsk_queue_destroy(struct request_sock_queue *queue)
 {
  /* make all the listen_opt local to us */
--
1.5.3.4
```