Subject: [PATCH] Fix memory leak in inet_hashtables.h when NUMA is on
Posted by Pavel Emelianov on Fri, 23 Nov 2007 16:13:11 GMT
View Forum Message <> Reply to Message

The inet_ehash_locks_alloc() looks like this:

```
#ifdef CONFIG_NUMA
 if (size > PAGE_SIZE)
  x = vmalloc(...);
 else
#endif
  x = kmalloc(...);
```

Unlike it, the inet_ehash_locks_alloc() looks like this:

```
#ifdef CONFIG_NUMA
 if (size > PAGE_SIZE)
  vfree(x);
 else
#else
  kfree(x);
#endif
```

The error is obvious - if the NUMA is on and the size
is less than the PAGE_SIZE we leak the pointer (kfree is
inside the #else branch).

Compiler doesn't warn us because after the kfree(x) there's
a "x = NULL" assignment, so here's another (minor?) bug: we
don't set x to NULL under certain circumstances.

Boring explanation, I know... Patch explains it better.

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

```
diff --git a/include/net/inet_hashtables.h b/include/net/inet_hashtables.h
index 469216d..37f6cb1 100644
--- a/include/net/inet_hashtables.h
+++ b/include/net/inet_hashtables.h
@@ -186,9 +186,8 @@ static inline void inet_ehash_locks_free(struct inet_hashinfo *hashinfo)
   if (size > PAGE_SIZE)
    vfree(hashinfo->ehash_locks);
   else
-#else
-  kfree(hashinfo->ehash_locks);
 #endif
```

```
+  kfree(hashinfo->ehash_locks);
   hashinfo->ehash_locks = NULL;
  }
 }
```