Subject: [PATCH 0/4] Fix race between sk_filter reassign and sk_clone()
Posted by Pavel Emelianov on Wed, 17 Oct 2007 09:45:54 GMT
View Forum Message <> Reply to Message

The race can result in that some sock will get an sk_filter
pointer set to kfree-d memory. Look

```
CPU1:                          CPU2:
sk_clone():                     sk_attach_filter():
  new_sk = sk_alloc(...);
  sock_copy(new_sk, sk);
  /* copies the filter ptr */
  ...
  filter = new_sk->sk_filter;
  if (filter)
                                old_fp = sk->sk_filter;
                                ...
                                sk_filter_release(old_fp);
                                  if (atomic_dec_and_test(&old_fp->refcnt))
    atomic_inc(&filter->refcnt);
                                     /* true */
                                     call_rcu(&fp->rcu, kfree);
```

that's it - after a quiescent state pass the new_sk will have
a pointer on kfree-d filter.

The same problem exists for detaching filter (SO_DETACH_FILTER).

The proposed fix consists of 3 preparation patches and the fix itself.

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