Subject: [PATCH 2.6.24-rc8-mm1 01/15] IPC/semaphores: code factorisation
Posted by Pierre Peiffer on Tue, 29 Jan 2008 16:02:30 GMT
View Forum Message <> Reply to Message

From: Pierre Peiffer <pierre.peiffer@bull.net>

Trivial patch which adds some small locking functions and makes use of them
to factorize some part of code and makes it cleaner.

Signed-off-by: Pierre Peiffer <pierre.peiffer@bull.net>
Acked-by: Serge Hallyn <serue@us.ibm.com>
---

 ipc/sem.c | 61 ++++++++++++++++++++++++++++++++++++++----------------------
 1 file changed, 31 insertions(+), 30 deletions(-)

Index: b/ipc/sem.c
===
--- a/ipc/sem.c
+++ b/ipc/sem.c
@@ -181,6 +181,25 @@ static inline struct sem_array *sem_lock
  return container_of(ipcp, struct sem_array, sem_perm);
 }

+static inline void sem_lock_and_putref(struct sem_array *sma)
+{
+ ipc_lock_by_ptr(&sma->sem_perm);
+ ipc_rcu_putref(sma);
+}
+
+static inline void sem_getref_and_unlock(struct sem_array *sma)
+{
+ ipc_rcu_getref(sma);
+ ipc_unlock(&(sma)->sem_perm);
+}
+
+static inline void sem_putref(struct sem_array *sma)
+{
+ ipc_lock_by_ptr(&sma->sem_perm);
+ ipc_rcu_putref(sma);
+ ipc_unlock(&(sma)->sem_perm);
+}
+
 static inline void sem_rmid(struct ipc_namespace *ns, struct sem_array *s)
 {
  ipc_rmid(&sem_ids(ns), &s->sem_perm);
@@ -700,19 +719,15 @@ static int semctl_main(struct ipc_namesp
   int i;

```
  if(nsems > SEMMSL_FAST) {
-   ipc_rcu_getref(sma);
-   sem_unlock(sma);
+   sem_getref_and_unlock(sma);

    sem_io = ipc_alloc(sizeof(ushort)*nsems);
    if(sem_io == NULL) {
-    ipc_lock_by_ptr(&sma->sem_perm);
-    ipc_rcu_putref(sma);
-    sem_unlock(sma);
+    sem_putref(sma);
     return -ENOMEM;
    }

-   ipc_lock_by_ptr(&sma->sem_perm);
-   ipc_rcu_putref(sma);
+   sem_lock_and_putref(sma);
    if (sma->sem_perm.deleted) {
     sem_unlock(sma);
     err = -EIDRM;
@@ -733,38 +748,30 @@ static int semctl_main(struct ipc_namesp
   int i;
   struct sem_undo *un;

-  ipc_rcu_getref(sma);
-  sem_unlock(sma);
+  sem_getref_and_unlock(sma);

   if(nsems > SEMMSL_FAST) {
    sem_io = ipc_alloc(sizeof(ushort)*nsems);
    if(sem_io == NULL) {
-    ipc_lock_by_ptr(&sma->sem_perm);
-    ipc_rcu_putref(sma);
-    sem_unlock(sma);
+    sem_putref(sma);
     return -ENOMEM;
    }
   }

   if (copy_from_user (sem_io, arg.array, nsems*sizeof(ushort))) {
-   ipc_lock_by_ptr(&sma->sem_perm);
-   ipc_rcu_putref(sma);
-   sem_unlock(sma);
+   sem_putref(sma);
    err = -EFAULT;
    goto out_free;
   }
```

```
   for (i = 0; i < nsems; i++) {
    if (sem_io[i] > SEMVMX) {
-    ipc_lock_by_ptr(&sma->sem_perm);
-    ipc_rcu_putref(sma);
-    sem_unlock(sma);
+    sem_putref(sma);
     err = -ERANGE;
     goto out_free;
    }
   }
-  ipc_lock_by_ptr(&sma->sem_perm);
-  ipc_rcu_putref(sma);
+  sem_lock_and_putref(sma);
   if (sma->sem_perm.deleted) {
    sem_unlock(sma);
    err = -EIDRM;
@@ -1044,14 +1051,11 @@ static struct sem_undo *find_undo(struct
   return ERR_PTR(PTR_ERR(sma));

  nsems = sma->sem_nsems;
- ipc_rcu_getref(sma);
- sem_unlock(sma);
+ sem_getref_and_unlock(sma);

  new = kzalloc(sizeof(struct sem_undo) + sizeof(short)*nsems, GFP_KERNEL);
  if (!new) {
-  ipc_lock_by_ptr(&sma->sem_perm);
-  ipc_rcu_putref(sma);
-  sem_unlock(sma);
+  sem_putref(sma);
   return ERR_PTR(-ENOMEM);
  }
  new->semadj = (short *) &new[1];
@@ -1062,13 +1066,10 @@ static struct sem_undo *find_undo(struct
  if (un) {
   spin_unlock(&ulp->lock);
   kfree(new);
-  ipc_lock_by_ptr(&sma->sem_perm);
-  ipc_rcu_putref(sma);
-  sem_unlock(sma);
+  sem_putref(sma);
   goto out;
  }
- ipc_lock_by_ptr(&sma->sem_perm);
- ipc_rcu_putref(sma);
+ sem_lock_and_putref(sma);
  if (sma->sem_perm.deleted) {
```

```
  sem_unlock(sma);
  spin_unlock(&ulp->lock);
```

--
Pierre Peiffer

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers