Subject: Re: [ckrm-tech] [RFC][PATCH 4/7] UBC: syscalls (user interface)
Posted by Matt Helsley on Sat, 19 Aug 2006 02:43:02 GMT
View Forum Message <> Reply to Message

On Fri, 2006-08-18 at 15:45 +0400, Kirill Korotaev wrote:

```
> Matt Helsley wrote:
>
> [... snip ...]
> >>--- ./kernel/ub/sys.c.ubsys 2006-07-28 18:52:18.000000000 +0400
> >>+++ ./kernel/ub/sys.c 2006-08-03 16:14:23.000000000 +0400
> >>@@ -0,0 +1,126 @@

<snip>

> >>+#else /* CONFIG_USER_RESOURCE */
> >>+
> >>+/*
> >>+ * The (rather boring) getluid syscall
> >>+ */
> >>+asmlinkage long sys_getluid(void)
> >>+{
> >>+ struct user_beancounter *ub;
> >>+
> >>+ ub = get_exec_ub();
> >>+ if (ub == NULL)
> >>+  return -EINVAL;
> >>+
> >>+ return ub->ub_uid;
> >>+}
> >>+
> >>+/*
> >>+ * The setluid syscall
> >>+ */
> >>+asmlinkage long sys_setluid(uid_t uid)
> >>+{
> >>+ int error;
> >>+ struct user_beancounter *ub;
> >>+ struct task_beancounter *task_bc;
> >>+
> >>+ task_bc = &current->task_bc;
> >>+
> >>+ /* You may not disown a setluid */
> >>+ error = -EINVAL;
> >>+ if (uid == (uid_t)-1)
> >>+  goto out;
> >>+
> >>+ /* You may only set an ub as root */
> >>+ error = -EPERM;
> >>+ if (!capable(CAP_SETUID))
> >>+  goto out;
> >
> >
```

> > With resource groups you don't necessarily have to be root -- just the
> > owner of the group and task.
> the question is - who is the owner of group?

Whoever is made the 'owner' of the directory is the owner of the group.
If you own both then you can add your task to your group.

> user, user group or who?
> Both are bad, since the same user can run inside the container and thus
> container will be potentially controllable/breakable from inside.

No, that's not a problem. The way shares work is you get a "portion" of
the parent group's resources and if the parent has limited your portion
you cannot exceed that. At the same time you can control how your
portion is dealt out within the child group.

> > Filesystems and appropriate share representations offer a way to give
> > regular users the ability to manage their resources without requiring
> > CAP_FOO.
> not sure what you propose...

A filesystem interface.

> we can introduce the following rules:
>
> containers (UB) can be created by process with SETUID cap only.
> subcontainers (SUB) can be created by any process.

Can subsubcontainers be created?

> what do you think?

I think a filesystem interface would work better. ;)

```
>
> >>+ /* Ok - set up a beancounter entry for this user */
> >>+ error = -ENOBUFS;
> >>+ ub = beancounter_findcreate(uid, NULL, UB_ALLOC);
> >>+ if (ub == NULL)
> >>+  goto out;
> >>+
> >>+ /* install bc */
> >>+ put_beancounter(task_bc->exec_ub);
> >>+ task_bc->exec_ub = ub;
> >>+ put_beancounter(task_bc->fork_sub);
> >>+ task_bc->fork_sub = get_beancounter(ub);
> >>+ error = 0;
> >>+out:
> >>+ return error;
> >>+}
> >>+
> >>+/*
> >>+ * The setbeanlimit syscall
> >>+ */
> >>+asmlinkage long sys_setublimit(uid_t uid, unsigned long resource,
> >>+  unsigned long *limits)
> >>+{
> >>+ int error;
> >>+ unsigned long flags;
> >>+ struct user_beancounter *ub;
> >>+ unsigned long new_limits[2];
> >>+
> >>+ error = -EPERM;
> >>+ if(!capable(CAP_SYS_RESOURCE))
> >>+  goto out;
> >
> >
```

> > Again, a filesystem interface would give us more flexibility when it
> > comes to allowing users to manage their resources while still preventing
> > them from exceeding limits.
> we can have 2 different root users with uid = 0 in 2 different containers.

You shouldn't need to have the 2 containers to give resource control to other users. In other words you shouldn't need to use containers in order to do resource management. The container model is by no means the only way to model resource management.

> > I doubt you really want to give owners of a container CAP_SYS_RESOURCE
> > and CAP_USER (i.e. total control over resource management) just to allow
> > them to manage their subset of the resources.
> The origin idea is that administator of the node can manage user
> resources only. Users can't, since otherwise they can increase the limits.

The user may wish to manage the resource usage of her applications within restrictions imposed by an administrator. If the user has a portion of resources then you only need to ensure that the sum of her resources does not exceed the administrator-provided limit.

> But we can allow them to manage sub beancoutners imho...

And subsubbeancounters?

<snip>

Cheers,
 -Matt Helsley