Subject: Re: [PATCH] namespaces: introduce sys_hijack (v4)
Posted by Cedric Le Goater on Tue, 16 Oct 2007 08:51:04 GMT
View Forum Message <> Reply to Message

>> hmm, I'm wondering how this is going to work for a process which
>> would have unshared its device (pts) namespace. How are we going
>> to link the pts living in different namespaces if the stdios of the
>> hijacked process is using them ? like in the case of a shell, which
>> is certainly something we would like to hijacked.
>>
>> it looks like a challenge for me. maybe I'm wrong.
>
> Might be a problem, but tough to address that until we actually
> have a dev ns or devpts ns and established semantics.
>
> Note the filestruct comes from current, not the hijack target, so
> presumably we can work around the tty issue in any case by
> keeping an open file across the hijack?
>
> For instance, use the attached modified version of hijack.c
> which puts a writeable fd for /tmp/helloworld in fd 5, then
> does hijack, then from the resulting shell do
>
> echo ab >&5
>
> So we should easily be able to work around it.

yes. it should.

> Or am i missing something?

I guess we need to work a little more on the pts/device namespace
to see how it interacts.

>>> The effect is a sort of namespace enter. The following program
>>> uses sys_hijack to 'enter' all namespaces of the specified pid.
>>> For instance in one terminal, do
>>>
>>> mount -t cgroup -ons /cgroup
>>> hostname
>>>   qemu
>>> ns_exec -u /bin/sh
>>>   hostname serge
>>>   echo $$
>>>   1073
>>>   cat /proc/$$/cgroup
>>>   ns:/node_1073
>> Is there a reason to have the 'node_' prefix ? couldn't we just

>> use $pid ?
>
> Good question. It's just how the ns-cgroup does it... If you want to
> send in a patch to change that, I'll ack it.

just below.

I gave a quick look to the ns subsystem and didn't see how the node_$pid
was destroyed. do we have to do a rmdir ?

Thanks,

C.

Signed-off-by: Cedric Le Goater <clg@fr.ibm.com>
---
 kernel/cgroup.c | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

Index: 2.6.23-mm1/kernel/cgroup.c
===
--- 2.6.23-mm1.orig/kernel/cgroup.c
+++ 2.6.23-mm1/kernel/cgroup.c
@@ -2604,7 +2604,7 @@ int cgroup_clone(struct task_struct *tsk
        cg = tsk->cgroups;
        parent = task_cgroup(tsk, subsys->subsys_id);

-       snprintf(nodename, MAX_CGROUP_TYPE_NAMELEN, "node_%d", tsk->pid);
+   snprintf(nodename, MAX_CGROUP_TYPE_NAMELEN, "%d", tsk->pid);

        /* Pin the hierarchy */
        atomic_inc(&parent->root->sb->s_active);

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