Subject: [PATCH 4/4] namespace containers: implement enter into existing container

Posted by serue on Mon, 19 Feb 2007 22:16:34 GMT

View Forum Message <> Reply to Message

From: Serge E. Hallyn <serue@us.ibm.com>
Subject: [PATCH 4/4] namespace containers: implement enter into existing container

Implement ns container subsys.can_attach(). Remove the constraint
in can_attach() that the destination container must be unpopulated.

When entering a container, if the container's nsproxy has not been
set, set it to the parent container's. This will eventually support
more complicated creation by composing namespaces from several
nsproxies.

Finally set the task's nsproxy to the container's to effect the
namespace transition.

Signed-off-by: Serge E. Hallyn <serue@us.ibm.com>

```
 kernel/ns_container.c |   30 ++++++++++++++++++++++++++----
 1 files changed, 26 insertions(+), 4 deletions(-)

fa70868cb500ac6e3319f6a5c04dac5e419cefbf
diff --git a/kernel/ns_container.c b/kernel/ns_container.c
index 1cc9cea..76044ad 100644
--- a/kernel/ns_container.c
+++ b/kernel/ns_container.c
@@ -23,6 +23,17 @@ static inline struct nscont *container_n
 	struct nscont, css);
 }

+/* for now we just take the parnet container nsproxy.
+ * eventually we will construct them based on file link activity */
+static void ns_create_nsproxy(struct nscont *ns, struct container *cont)
+{
+ struct container *parent = cont->parent;
+ struct nscont *parentns;
+ parentns = container_nscont(parent);
+ ns->nsproxy = parentns->nsproxy;
+ get_nsproxy(ns->nsproxy);
+}
+
 int ns_container_clone(struct task_struct *tsk, struct nsproxy *nsproxy)
 {
  swap_nsproxies(tsk, nsproxy);
@@ -52,9 +63,6 @@ int ns_can_attach(struct container_subsy
    return -EPERM;
  }

- if (atomic_read(&cont->count) != 0)
-  return -EPERM;
-
  c = task_container(tsk, &ns_subsys);
  if (c && c != cont->parent)
   return -EPERM;
@@ -90,6 +98,20 @@ void ns_init_from_task(struct container
  get_nsproxy(ns->nsproxy);
 }

+void ns_attach(struct container_subsys *ss, struct container *cont,
+     struct container *old_cont, struct task_struct *tsk)
+{
+ struct nscont *ns = container_nscont(cont);
+ if (!ns->nsproxy) {
+  spin_lock(&ns->lock);
+  if (!ns->nsproxy)
+   ns_create_nsproxy(ns, cont);
+  spin_unlock(&ns->lock);
+ }
+ if (tsk->nsproxy != ns->nsproxy)
+  swap_nsproxies(tsk, ns->nsproxy);
+}
+
 static void ns_destroy(struct container_subsys *ss,
         struct container *cont)
 {
@@ -104,7 +126,7 @@ static struct container_subsys ns_subsys
  .create = ns_create,
  .destroy  = ns_destroy,
  .can_attach = ns_can_attach,
- //.attach = ns_attach,
+ .attach = ns_attach,
  //.post_attach = ns_post_attach,
  //.populate = ns_populate,
  .init_from_task = ns_init_from_task,
--
1.1.6
```