Subject: [PATCH] namespaces: fix exit race by splitting exit
Posted by serue on Fri, 26 Jan 2007 05:26:59 GMT
View Forum Message <> Reply to Message

Ok, could you verify that the following patch at least solves
the oopsing?

(I can't reproduce the oops with Daniel's test prog)

thanks,
-serge

From: Serge E. Hallyn <serue@us.ibm.com>
Subject: [PATCH] namespaces: fix exit race by splitting exit

Fix exit race by splitting the nsproxy putting into two pieces.
First piece reduces the nsproxy refcount. If we dropped the last
reference, then it puts the mnt_ns, and returns the nsproxy as a
hint to the caller. Else it returns NULL. The second piece of
exiting task namespaces sets tsk->nsproxy to NULL, and drops the
references to other namespaces and frees the nsproxy only if an
nsproxy was passed in.

A little awkward and should probably be reworked, but hopefully
it fixes the NFS oops.

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

---

 include/linux/nsproxy.h | 30 ++++++++++++++++++++----------
 kernel/exit.c | 6 ++++--
 kernel/fork.c | 4 ++--
 kernel/nsproxy.c | 16 ++++++++++++++-
 4 files changed, 40 insertions(+), 16 deletions(-)

ab969afa3624aba0bc26dc237d27178137c05d46
diff --git a/include/linux/nsproxy.h b/include/linux/nsproxy.h
index 0b9f0dc..678e1d3 100644
--- a/include/linux/nsproxy.h
+++ b/include/linux/nsproxy.h
@@ -35,22 +35,30 @@ struct nsproxy *dup_namespaces(struct ns
 int copy_namespaces(int flags, struct task_struct *tsk);
 void get_task_namespaces(struct task_struct *tsk);
 void free_nsproxy(struct nsproxy *ns);
+struct nsproxy *put_nsproxy(struct nsproxy *ns);

-static inline void put_nsproxy(struct nsproxy *ns)

```
+static inline void finalize_put_nsproxy(struct nsproxy *ns)
 {
- if (atomic_dec_and_test(&ns->count)) {
+ if (ns)
   free_nsproxy(ns);
- }
 }

-static inline void exit_task_namespaces(struct task_struct *p)
+static inline void put_and_finalize_nsproxy(struct nsproxy *ns)
 {
- struct nsproxy *ns = p->nsproxy;
- if (ns) {
-  task_lock(p);
-  p->nsproxy = NULL;
-  task_unlock(p);
-  put_nsproxy(ns);
- }
+ finalize_put_nsproxy(put_nsproxy(ns));
+}
+
+static inline struct nsproxy *preexit_task_namespaces(struct task_struct *p)
+{
+ return put_nsproxy(p->nsproxy);
+}
+
+static inline void exit_task_namespaces(struct task_struct *p,
+       struct nsproxy *ns)
+{
+ task_lock(p);
+ p->nsproxy = NULL;
+ task_unlock(p);
+ finalize_put_nsproxy(ns);
 }
 #endif
diff --git a/kernel/exit.c b/kernel/exit.c
index 3540172..a5bf532 100644
--- a/kernel/exit.c
+++ b/kernel/exit.c
@@ -396,7 +396,7 @@ void daemonize(const char *name, ...)
  current->fs = fs;
  atomic_inc(&fs->count);

- exit_task_namespaces(current);
+ put_and_finalize_nsproxy(current->nsproxy);
  current->nsproxy = init_task.nsproxy;
  get_task_namespaces(current);
```

```
@@ -853,6 +853,7 @@ static void exit_notify(struct task_stru
 fastcall NORET_TYPE void do_exit(long code)
 {
  struct task_struct *tsk = current;
+ struct nsproxy *ns;
  int group_dead;

  profile_task_exit(tsk);
@@ -938,8 +939,9 @@ fastcall NORET_TYPE void do_exit(long co

  tsk->exit_code = code;
  proc_exit_connector(tsk);
+ ns = preexit_task_namespaces(tsk);
  exit_notify(tsk);
- exit_task_namespaces(tsk);
+ exit_task_namespaces(tsk, ns);
 #ifdef CONFIG_NUMA
  mpol_free(tsk->mempolicy);
  tsk->mempolicy = NULL;
diff --git a/kernel/fork.c b/kernel/fork.c
index fc723e5..4cf8684 100644
--- a/kernel/fork.c
+++ b/kernel/fork.c
@@ -1265,7 +1265,7 @@ static struct task_struct *copy_process(
  return p;

 bad_fork_cleanup_namespaces:
- exit_task_namespaces(p);
+ put_and_finalize_nsproxy(p->nsproxy);
 bad_fork_cleanup_keys:
  exit_keys(p);
 bad_fork_cleanup_mm:
@@ -1711,7 +1711,7 @@ asmlinkage long sys_unshare(unsigned lon
  }

  if (new_nsproxy)
-  put_nsproxy(new_nsproxy);
+  put_and_finalize_nsproxy(new_nsproxy);

 bad_unshare_cleanup_ipc:
  if (new_ipc)
diff --git a/kernel/nsproxy.c b/kernel/nsproxy.c
index f5b9ee6..7b05bce 100644
--- a/kernel/nsproxy.c
+++ b/kernel/nsproxy.c
@@ -117,7 +117,7 @@ int copy_namespaces(int flags, struct ta
   goto out_pid;
```

```
 out:
- put_nsproxy(old_ns);
+ put_and_finalize_nsproxy(old_ns);
  return err;

 out_pid:
@@ -135,6 +135,20 @@ out_ns:
  goto out;
 }

+struct nsproxy *put_nsproxy(struct nsproxy *ns)
+{
+ if (ns) {
+  if (atomic_dec_and_test(&ns->count)) {
+   if (ns->mnt_ns) {
+    put_mnt_ns(ns->mnt_ns);
+    ns->mnt_ns = NULL;
+   }
+   return ns;
+  }
+ }
+ return NULL;
+}
+
 void free_nsproxy(struct nsproxy *ns)
 {
  if (ns->mnt_ns)
--
1.1.6
```

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers