Subject: [PATCH] eCryptfs: Fix refs to pid and user_ns
Posted by Michael Halcrow on Thu, 17 Apr 2008 17:03:31 GMT
View Forum Message <> Reply to Message

On Thu, Apr 17, 2008 at 10:34:06AM -0500, Serge E. Hallyn wrote:
```
> Quoting Michael Halcrow (mhalcrow@us.ibm.com):
> > @@ -206,6 +210,7 @@ ecryptfs_spawn_daemon(struct ecryptfs_daemon **daemon, uid_t
euid, pid_t pid)
> >    goto out;
> >   }
> >   (*daemon)->euid = euid;
> > + (*daemon)->user_ns = user_ns;
> >   (*daemon)->pid = pid;
>
> You'll want to do a get_pid() here, no?
>
> And get_user_ns().
>

> It's not because you particulary need them to stick around, but just
> to ensure no wraparound causes another daemon with the same struct
> pid or user_namespace to be spawned.  Pretty gosh-darned unlikely,
> but still...
> ...
> > @@ -372,12 +383,24 @@ int ecryptfs_process_response(struct ecryptfs_message *msg,
uid_t euid,
> >   msg_ctx = &ecryptfs_msg_ctx_arr[msg->index];
> >   mutex_lock(&msg_ctx->mux);
> >   mutex_lock(&ecryptfs_daemon_hash_mux);
> > - rc = ecryptfs_find_daemon_by_euid(&daemon, msg_ctx->task->euid);
> > + rcu_read_lock();
> > + nsproxy = task_nsproxy(msg_ctx->task);
> > + if (nsproxy == NULL) {
> > +  rc = -EBADMSG;
> > +  printk(KERN_ERR "%s: Receiving process is a zombie. Dropping "
> > +         "message.\n", __func__);
> > +  rcu_read_unlock();
> > +  mutex_unlock(&ecryptfs_daemon_hash_mux);
> > +  goto wake_up;
> > + }
> > + rc = ecryptfs_find_daemon_by_euid(&daemon, msg_ctx->task->euid,
> > +       nsproxy->user_ns);
> > + rcu_read_unlock();
> >   mutex_unlock(&ecryptfs_daemon_hash_mux);
> >   if (rc) {
> >    rc = -EBADMSG;
> >    printk(KERN_WARNING "%s: User [%d] received a "
> > -        "message response from process [%d] but does "
> > +        "message response from process [0x%p] but does "
> >          "not have a registered daemon\n", __func__,
> >          msg_ctx->task->euid, pid);
> >    goto wake_up;
> > @@ -389,10 +412,17 @@ int ecryptfs_process_response(struct ecryptfs_message *msg,
uid_t euid,
> >          euid, msg_ctx->task->euid);
> >    goto unlock;
> >   }
> > + if (nsproxy->user_ns != user_ns) {
>
> Since you didn't grab a ref to the nsproxy, it's possible that it
> will have been freed before this, right?  So you probably just want
> to grab a copy of nsproxy->user_ns while under the rcu_read_lock,
> where you can be sure it's still around.

Have eCryptfs properly reference the pid and user_ns objects. Copy
user_ns out of nsproxy in case nsproxy goes away after we drop the
lock.

Signed-off-by: Michael Halcrow <mhalcrow@us.ibm.com>
---
 fs/ecryptfs/messaging.c |   16 ++++++++++++----
 1 files changed, 12 insertions(+), 4 deletions(-)

diff --git a/fs/ecryptfs/messaging.c b/fs/ecryptfs/messaging.c
index f0d74b8..61506e5 100644
--- a/fs/ecryptfs/messaging.c
+++ b/fs/ecryptfs/messaging.c
@@ -20,6 +20,8 @@
  * 02111-1307, USA.
  */
 #include <linux/sched.h>
+#include <linux/user_namespace.h>
+#include <linux/nsproxy.h>
 #include "ecryptfs_kernel.h"

 static LIST_HEAD(ecryptfs_msg_ctx_free_list);
@@ -208,8 +210,8 @@ ecryptfs_spawn_daemon(struct ecryptfs_daemon **daemon, uid_t euid,
   goto out;
  }
  (*daemon)->euid = euid;
- (*daemon)->user_ns = user_ns;
- (*daemon)->pid = pid;
+ (*daemon)->user_ns = get_user_ns(user_ns);
+ (*daemon)->pid = get_pid(pid);
  (*daemon)->task = current;
  mutex_init(&(*daemon)->mux);
  INIT_LIST_HEAD(&(*daemon)->msg_ctx_out_queue);
@@ -298,6 +300,10 @@ int ecryptfs_exorcise_daemon(struct ecryptfs_daemon *daemon)
  hlist_del(&daemon->euid_chain);
  if (daemon->task)
   wake_up_process(daemon->task);
+ if (daemon->pid)
+  put_pid(daemon->pid);
+ if (daemon->user_ns)
+  put_user_ns(daemon->user_ns);
  mutex_unlock(&daemon->mux);
  memset(daemon, 0, sizeof(*daemon));
  kfree(daemon);
@@ -368,6 +374,7 @@ int ecryptfs_process_response(struct ecryptfs_message *msg, uid_t
euid,
  struct ecryptfs_msg_ctx *msg_ctx;
  size_t msg_size;
  struct nsproxy *nsproxy;
+ struct user_namespace *current_user_ns;
  int rc;

  if (msg->index >= ecryptfs_message_buf_len) {
@@ -391,8 +398,9 @@ int ecryptfs_process_response(struct ecryptfs_message *msg, uid_t
euid,
   mutex_unlock(&ecryptfs_daemon_hash_mux);
   goto wake_up;
  }
+ current_user_ns = nsproxy->user_ns;
  rc = ecryptfs_find_daemon_by_euid(&daemon, msg_ctx->task->euid,
-      nsproxy->user_ns);
+       current_user_ns);
  rcu_read_unlock();
  mutex_unlock(&ecryptfs_daemon_hash_mux);
  if (rc) {
@@ -410,7 +418,7 @@ int ecryptfs_process_response(struct ecryptfs_message *msg, uid_t
euid,
        euid, msg_ctx->task->euid);
   goto unlock;
  }
- if (nsproxy->user_ns != user_ns) {
+ if (current_user_ns != user_ns) {
   rc = -EBADMSG;
   printk(KERN_WARNING "%s: Received message from user_ns "
          "[0x%p]; expected message from user_ns [0x%p]\n",
--
1.5.1.6
```

____
Containers mailing list

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