Subject: Re: - merge-sys_clone-sys_unshare-nsproxy-and-namespace.patch removed from -mm tree
Posted by Oleg Nesterov on Sun, 17 Jun 2007 14:38:30 GMT
View Forum Message <> Reply to Message

On 06/16, Herbert Poetzl wrote:
>
> On Tue, May 08, 2007 at 07:45:35PM -0700, akpm@linux-foundation.org wrote:
> >
> > The patch titled
> > Merge sys_clone()/sys_unshare() nsproxy and namespace handling
> > has been removed from the -mm tree. Its filename was
> > merge-sys_clone-sys_unshare-nsproxy-and-namespace.patch
> >
> > This patch was dropped because it was merged into mainline or a subsystem tree
> >
>
> .. [zapped] ...
>
> > + * Called from unshare. Unshare all the namespaces part of nsproxy.
> > + * On sucess, returns the new nsproxy and a reference to old nsproxy
> > + * to make sure it stays around.
> > + */
> > +int unshare_nsproxy_namespaces(unsigned long unshare_flags,
> > + struct nsproxy **new_nsp, struct fs_struct *new_fs)
> > +{
>
> this makes sys_unshare leak and nsproxy (reference)
>
> can be tested with the following command sequence:
> vcmd -nu ^17 -- vcmd -nu ^17 -- sleep 10

I know almost nothing about this stuff, could you please explain in brief
what this command does and how do you detect a leak?

> (and some nsproxy accounting/debugging as used in
> Linux-VServer)
>
> we probably want to drop the reference to the old
> nsproxy in sys_unshare() but I do not see a good reason
> to take the reference in the first place (at least not
> with the code in mainline 2.6.22-rc4)

At first glance, sys_unshare() drops the reference to the old nsproxy,

```
old_nsproxy = current->nsproxy;
current->nsproxy = new_nsproxy;
new_nsproxy = old_nsproxy;
```

```
...

if (new_nsproxy)
 put_nsproxy(new_nsproxy);
```

However, nsproxy's code is full of strange unneeded get/put calls, for example:

```
struct uts_namespace *copy_utsname(int flags, struct uts_namespace *old_ns)
{
 struct uts_namespace *new_ns;

 BUG_ON(!old_ns);
 get_uts_ns(old_ns);

 if (!(flags & CLONE_NEWUTS))
  return old_ns;

 new_ns = clone_uts_ns(old_ns);

 put_uts_ns(old_ns);
 return new_ns;
}
```

I think it would be better to do

```
struct uts_namespace *copy_utsname(int flags, struct uts_namespace *old_ns)
{
 struct uts_namespace *new_ns;

 BUG_ON(!old_ns);

 if (!(flags & CLONE_NEWUTS)) {
  get_uts_ns(old_ns);
  return old_ns;
 }

 new_ns = clone_uts_ns(old_ns);
 return new_ns;
}
```

Not only this looks better (imho), this is more robust.

Let's look at copy_namespaces(), it does the same "get_xxx() in advance", but -EPERM forgets to do put_nsproxy(), so we definitely have a leak in copy_process().

So, if the command above does clone() which fails, perhaps this can explain the problem.

Oleg.

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