Subject: Re: [PATCH 1/2] namespaces: introduce sys_hijack (v10)
Posted by Casey Schaufler on Tue, 27 Nov 2007 22:54:26 GMT
View Forum Message <> Reply to Message

--- "Serge E. Hallyn" <serue@us.ibm.com> wrote:

> Quoting Stephen Smalley (sds@tycho.nsa.gov):
> > On Tue, 2007-11-27 at 10:11 -0600, Serge E. Hallyn wrote:
> > > Quoting Crispin Cowan (crispin@crispincowan.com):
> > > > Just the name "sys_hijack" makes me concerned.
> > > >
> > > > This post describes a bunch of "what", but doesn't tell us about "why"
> > > > we would want this. What is it for?
> > >
> > > Please see my response to Casey's email.
> > >
> > > > And I second Casey's concern about careful management of the privilege
> > > > required to "hijack" a process.
> > >
> > > Absolutely. We're definately still in RFC territory.
> > >
> > > Note that there are currently several proposed (but no upstream) ways to
> > > accomplish entering a namespace:
> > >
> > > 1. bind_ns() is a new pair of syscalls proposed by Cedric. An
> > > nsproxy is given an integer id. The id can be used to enter
> > > an nsproxy, basically a straight current->nsproxy = target_nsproxy;
> > >
> > > 2. I had previously posted a patchset on top of the nsproxy
> > > cgroup which allowed entering a nsproxy through the ns cgroup
> > > interface.
> > >
> > > There are objections to both those patchsets because simply switching a
> > > task's nsproxy using a syscall or file write in the middle of running a
> > > binary is quite unsafe. Eric Biederman had suggested using ptrace or
> > > something like it to accomplish the goal.
> > >
> > > Just using ptrace is however not safe either. You are inheriting *all*
> > > of the target's context, so it shouldn't be difficult for a nefarious
> > > container/vserver admin to trick the host admin into running something
> > > which gives the container/vserver admin full access to the host.
> >
> > I don't follow the above - with ptrace, you are controlling a process
> > already within the container (hence in theory already limited to its
> > container), and it continues to execute within that container. What's
> > the issue there?
>
> Hmm, yeah, I may have overspoken - I'm not good at making up exploits

> but while I see it possible to confuse the host admin by setting bogus
> environment, I guess there may not be an actual exploit.
>
> Still after the fork induced through ptrace, we'll have to execute a
> file out of the hijacked process' namespaces and path (unless we get
> *really* 'exotic'). With hijack, execution continues under the caller's
> control, which I do much prefer.
>
> The remaining advantages of hijack over ptrace (beside "using ptrace for
> that is crufty") are
>
> 1. not subject to pid wraparound (when doing hijack_cgroup
> or hijack_ns)
> 2. ability to enter a namespace which has no active processes
>
> These also highlight selinux issues. In the case of hijacking an
> empty cgroup, there is no security context (because there is no task) so
> the context of 'current' will be used. In the case of hijacking a
> populated cgroup, a task is chosen "at random" to be the hijack source.
>
> So there are two ways to look at deciding which context to use. Since
> control continues in the original acting process' context, we might
> want the child to continue in its context. However if the process
> creates any objects in the virtual server, we don't want them
> mislabeled, so we might want the task in the hijacked task's context.

I wouldn't be surprised if you've been over this a dozen times
already, but why hijack an existing process instead of injecting
a new one with completely specified attributes? That way you don't
distinguish between an empty cgroup and a propulated one and you're
not at the mercy of the available hijackees. I know that I would be
much less uncomfortable with that schenario.

> Sigh. So here's why I thought I'd punt on selinux at least until I had
> a working selinux-enforced container/vserver :)
>
> -serge
>
> PS: I'm certainly open to the suggestion that the kernel patch in the
> end us as crufty as using ptrace.
>
> > > That's where the hijack idea came from. Yes, I called it hijack to make
> > > sure alarm bells went off :) bc it's definately still worrisome. But at
> > > this point I believe it is the safest solution suggested so far.
> >
> > --
> > Stephen Smalley
> > National Security Agency

> >
> > -
> > To unsubscribe from this list: send the line "unsubscribe
> linux-security-module" in
> > the body of a message to majordomo@vger.kernel.org
> > More majordomo info at http://vger.kernel.org/majordomo-info.html
> -
> To unsubscribe from this list: send the line "unsubscribe
> linux-security-module" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at http://vger.kernel.org/majordomo-info.html
>
>
>

Casey Schaufler
casey@schaufler-ca.com

_____

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