Subject: Re: [PATCH] cgroups: implement device whitelist lsm (v3)
Posted by Stephen Smalley on Mon, 17 Mar 2008 16:48:10 GMT
View Forum Message <> Reply to Message

On Mon, 2008-03-17 at 09:16 -0700, Casey Schaufler wrote:
> --- "Serge E. Hallyn" <serue@us.ibm.com> wrote:
>
> > Quoting Casey Schaufler (casey@schaufler-ca.com):
> > ...
> > > In particular, capabilities are not an access control mechanism,
> > > they are a privilege mechanism. A lot of discussion about LSM has
> > > centered around the appropriate charactoristics of an LSM, and
> > > these discussions always assume that the LSM in question is
> > > exactly an access control mechanism. If we split the LSM into
> > > a LACM for access control and an LPM for privilege management
> > > maybe we can eliminate the most contentious issues.
> > >
> > > Does anyone know why that would be stoopid before I whack out
> > > patches?
> >
> > No I'd like to see those patches. It would ideally allow LSM to become
> > *purely* restrictive and LPM to be purely empowering, presumably making
> > the resulting hook sets easier to review and maintain. The LPM wouldn't
> > (I assume) gain any *new* hook points so we wouldn't be adding any new
> > places for hooks to be overriden by a rootkit.
>
> I don't expect to put in any additional hooks points, although
> it's safe to bet that someone will want to pretty quickly. What
> I see as the big concern is our old friend the granularity question.
> I can pretty well predict that we'll have quite a bruhaha over
> whether each hook point should have it's own hook or if they should
> be shared based on the privilege supported. For example, in namei.c
> the function generic_permission() currently calls
> capable(CAP_DAC_OVERRIDE). The privilege supported approach would
> be to create a hook that gets used in many places that is a drop-in
> replacement for that,
>
> 	if (capable(CAP_DAC_OVERRIDE))
> becomes
> 	if (lpm_dac_override())

nit: I'd use priv_ rather than lpm_, just as we use security_ rather
than lsm_.

Do you plan to pass other arguments to the privilege hook call, like the
object? If not, then there is no point in changing the capable call
sites at all - just change its implementation to invoke a priv_capable()
hook instead of a security_capable() hook.

> The alternative is to go the same route as the LSM, where it
> becomes
>
> if (lpm_generic_permission_may_exec())
>
> The former scheme is much easier to implement. It also would
> mean that if would wanted to implement a finer granularity on
> DAC overrides (e.g. CAP_DAC_READ, CAP_DAC_WRITE, CAP_DAC_EXECUTE)
> you would have to introduce new hooks. That wouldn't be any worse
> than today's situation where you would have to change the argument
> passed to capable as far as the calling (e.g. generic_permission)
> code is concerned, but it would mean updating all the LPMs. I
> currently count 1084 calls to capable (sloppy grep method) and that's
> way too many hooks in my mind. But, if there's anyone who thinks
> that the way to go is for each existing capable call to be a hook,
> feel free to make a convincing argument.
>
> This should be fun.

Changing all of the call sites seems a bit prohibitive for an initial implementation; rewiring the internals of capable() to use a new privilege hook interface would be a lot simpler.

You also have to migrate the other security hooks presently used to support capabilities to your privilege framework.

--
Stephen Smalley
National Security Agency

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