Subject: [RFC] [PATCH -mm] oom_kill: remove uid==0 checks
Posted by serue on Wed, 12 Dec 2007 21:18:35 GMT
View Forum Message <> Reply to Message

>From a5fd2d7c75168076dc6b4b94ea8cda529fc506b1 Mon Sep 17 00:00:00 2001
From: serue@us.ibm.com <serue@us.ibm.com>
Date: Wed, 5 Dec 2007 14:07:40 -0800
Subject: [RFC] [PATCH -mm] oom_kill: remove uid==0 checks

Root processes are considered more important when out of memory
and killing proceses. The check for CAP_SYS_ADMIN was augmented
with a check for uid==0 or euid==0.

There are several possible ways to look at this:

1. uid comparisons are unnecessary, trust CAP_SYS_ADMIN
   alone. However CAP_SYS_RESOURCE is the one that really
   means "give me extra resources" so allow for that as
   well.
2. Any privileged code should be protected, but uid is not
   an indication of privilege. So we should check whether
   any capabilities are raised.
3. uid==0 makes processes on the host as well as in containers
   more important, so we should keep the existing checks.
4. uid==0 makes processes only on the host more important,
   even without any capabilities. So we should be keeping
   the (uid==0||euid==0) check but only when
   userns==&init_user_ns.

I'm following number 1 here.

Andrew, I've cc:d you here bc in doing this patch I noticed that your
64-bit capabilities patch switched this code from an explicit check
of cap_t(p->cap_effective) to using __capable(). That means that
now being glossed over by the oom killer means PF_SUPERPRIV will
be set. Is that intentional?

Signed-off-by: Serge Hallyn <serue@us.ibm.com>
---
 mm/oom_kill.c | 2 +-
 1 files changed, 1 insertions(+), 1 deletions(-)

diff --git a/mm/oom_kill.c b/mm/oom_kill.c
index 016127e..9fd8d5d 100644
--- a/mm/oom_kill.c
+++ b/mm/oom_kill.c
@@ -128,7 +128,7 @@ unsigned long badness(struct task_struct *p, unsigned long uptime,
  * Superuser processes are usually more important, so we make it

```
   * less likely that we kill those.
   */
- if (__capable(p, CAP_SYS_ADMIN) || p->uid == 0 || p->euid == 0)
+ if (__capable(p, CAP_SYS_ADMIN) || __capable(p, CAP_SYS_RESOURCE))
   points /= 4;

  /*
--
1.5.1
```

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

---

Subject: Re: [RFC] [PATCH -mm] oom_kill: remove uid==0 checks
Posted by Andrew Morgan on Wed, 12 Dec 2007 23:06:17 GMT
View Forum Message <> Reply to Message

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

Serge E. Hallyn wrote:
> Andrew, I've cc:d you here bc in doing this patch I noticed that your
> 64-bit capabilities patch switched this code from an explicit check
> of cap_t(p->cap_effective) to using __capable(). That means that
> now being glossed over by the oom killer means PF_SUPERPRIV will
> be set. Is that intentional?

Yes, I switched the check because the old one didn't work with the new
capability representation.

However, I had not thought this aspect of this replacement through. At
the time, it seemed obvious but in this case it actually depends on
whether you think using privilege (PF_SUPERPRIV) means "benefited from
privilege", or "successfully completed a privileged operation".

I suspect, in this case, the correct thing to do is add the equivalent of:

#define CAPABLE_PROBE_ONLY(a,b)  (!security_capable(a,b))

and use that in the code in question. That is, return to the old
behavior in a way that will not break if we ever need to add more bits.

Thanks for finding this.

Cheers

Andrew

>
> Signed-off-by: Serge Hallyn <serue@us.ibm.com>
> ---
>  mm/oom_kill.c |    2 +-
>  1 files changed, 1 insertions(+), 1 deletions(-)
>
> diff --git a/mm/oom_kill.c b/mm/oom_kill.c
> index 016127e..9fd8d5d 100644
> --- a/mm/oom_kill.c
> +++ b/mm/oom_kill.c
> @@ -128,7 +128,7 @@ unsigned long badness(struct task_struct *p, unsigned long uptime,
>     * Superuser processes are usually more important, so we make it
>     * less likely that we kill those.
>     */
> - if (__capable(p, CAP_SYS_ADMIN) || p->uid == 0 || p->euid == 0)
> + if (__capable(p, CAP_SYS_ADMIN) || __capable(p, CAP_SYS_RESOURCE))
>     points /= 4;
>
>   /*

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.7 (Darwin)
Comment: Using GnuPG with Mozilla - http://enigmail.mozdev.org

iD8DBQFHYGln+bHCR3gb8jsRAgNwAKDQED4YNy479LKfDL1fhVGWMK22eACgjPMh
JcFgzPsvIQkoatjvJ1vtHQ8=
=50l1
-----END PGP SIGNATURE-----

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

---

Subject: Re: [RFC] [PATCH -mm] oom_kill: remove uid==0 checks
Posted by akpm on Fri, 21 Dec 2007 00:34:42 GMT
View Forum Message <> Reply to Message

On Wed, 12 Dec 2007 15:06:17 -0800
Andrew Morgan <morgan@kernel.org> wrote:

> Serge E. Hallyn wrote:
> > Andrew, I've cc:d you here bc in doing this patch I noticed that your
> > 64-bit capabilities patch switched this code from an explicit check
> > of cap_t(p->cap_effective) to using __capable().  That means that

> > now being glossed over by the oom killer means PF_SUPERPRIV will
> > be set.  Is that intentional?
>
> Yes, I switched the check because the old one didn't work with the new
> capability representation.
>
> However, I had not thought this aspect of this replacement through. At
> the time, it seemed obvious but in this case it actually depends on
> whether you think using privilege (PF_SUPERPRIV) means "benefited from
> privilege", or "successfully completed a privileged operation".
>
> I suspect, in this case, the correct thing to do is add the equivalent of:
>
> #define CAPABLE_PROBE_ONLY(a,b)  (!security_capable(a,b))
>
> and use that in the code in question. That is, return to the old
> behavior in a way that will not break if we ever need to add more bits.

I'm struggling to understand whether the above was an ack, a nack or a
quack.

> Thanks for finding this.

>From that I'll assume ack ;)

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

---

Subject: Re: [RFC] [PATCH -mm] oom_kill: remove uid==0 checks
Posted by serue on Fri, 21 Dec 2007 14:46:06 GMT
View Forum Message <> Reply to Message

Quoting Andrew Morton (akpm@linux-foundation.org):
> On Wed, 12 Dec 2007 15:06:17 -0800
> Andrew Morgan <morgan@kernel.org> wrote:
>
> > Serge E. Hallyn wrote:
> > > Andrew, I've cc:d you here bc in doing this patch I noticed that your
> > > 64-bit capabilities patch switched this code from an explicit check
> > > of cap_t(p->cap_effective) to using __capable().  That means that
> > > now being glossed over by the oom killer means PF_SUPERPRIV will
> > > be set.  Is that intentional?
> >
> > Yes, I switched the check because the old one didn't work with the new
> > capability representation.
> >

> > However, I had not thought this aspect of this replacement through. At
> > the time, it seemed obvious but in this case it actually depends on
> > whether you think using privilege (PF_SUPERPRIV) means "benefited from
> > privilege", or "successfully completed a privileged operation".
> >
> > I suspect, in this case, the correct thing to do is add the equivalent of:
> >
> > #define CAPABLE_PROBE_ONLY(a,b)  (!security_capable(a,b))
> >
> > and use that in the code in question. That is, return to the old
> > behavior in a way that will not break if we ever need to add more bits.

Oh, I'm sorry - Andrew Morgan, I somehow read that email to say you were
going to post such a patch, and let it fall off my todo list.  Should I
go ahead and post a patch or do you have one ready?

> I'm struggling to understand whether the above was an ack, a nack or a
> quack.
>
> > Thanks for finding this.
>
> >From that I'll assume ack ;)

It actually wasn't an ack of my patch.  But I'm not sure where to look
for that.

thanks,
-serge

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