



 jmspeex • [Log Out](#)

Ubuntu "linux-source-2.6.15" package

[Overview](#) [Code](#) **[Bugs](#)** [Blueprints](#) [Translations](#) [Answers](#) Bug #52600:Reported by  jmspeex on 2006-07-11 ([Activity log](#))

Unprivileged users can monopolize CPU using SCHED_FIFO and max priority

This report is **private**

Security vulnerability

This bug has 2 duplicates

[Mark as duplicate](#)[Convert to a question](#)

Affects	Status	Importance	Assigned to	Milestone
linux-source-2.6.15 (Ubuntu)	New	High	 Ben Collins	

[Also affects project](#)[Also affects distribution](#)[Nominate for release](#)This is the same as [Bug #52224](#)

except that I just realized it's actually a security issue because not only admin users, but any users are allowed to set SCHED_FIFO with no restriction. All that's needed is a program like this one:

```
#include <sched.h>
int main() {
    struct sched_param param;
    param.sched_priority = sched_get_priority_max(SCHED_FIFO);
    sched_setscheduler(0, SCHED_FIFO, &param);
    while(1);
}
```

It will crash the system with no hope of recovering, even if ran by a normal user with no special rights (not even desktop-type rights). I know that it's hard to provide security against a user crashing the machine (e.g. swapping like hell or fork bomb), but this is a bit extreme.

I assume the best way to fix the problem is to

1) Limit the use of SCHED_FIFO to members of the audio group; and especially







2) Make sure that all non-root RT tasks are limited to e.g. 70% CPU over a time window and have a lower maximum RT priority (e.g. 40).

AFAIK, all these are already supported by the kernel and PAM, which is why I'm surprised Dapper got it so wrong.

Duplicates of this bug

- [Bug #52224](#)
- [Bug #67711](#)

[Unsubscribe](#)[Subscribe someone else](#)**Subscribers**

-  Ben Collins
-  Colin Watson
-  Kyle McMartin
-  Ubuntu Kernel Team
-  Ubuntu Security Team
-  jmspeex

[Update description / tags](#)[Link a related branch](#)[Link to CVE](#)

 **Martin Pitt** wrote on 2006-07-11: **Re: [Bug 52600] Any user can easily crash Dapper** (permalink)

Hi jmspeex,

jmspeex [2006-07-11 3:18 -0000]:
[\[...\]](#)

I think it's no different to the classic `:(){ :|:& };;`
bash fork bomb;
after executing it for some seconds, the computer is
basically dead.
PAM is no help there either, since you cannot limit the
total amount
of resources per user right now.


Unfortunately I did not yet see a sane resource
limitation
configuration that would be appropriate to install by
default. If you
have some news about that, I'd be happy to learn about
them.

Thank you!

Martin

[\[...\]](#)

In a world without walls and fences, who needs Windows
and Gates?

 **jmspeex** wrote on 2006-07-11: **Re: Any user can easily crash Dapper** (permalink)

> I think it's no different to the classic `:(){ :|:& };;`
bash fork bomb;
> after executing it for some seconds, the computer is
basically dead.

This is not the same as a fork bomb. With a fork bomb,
you have a change (even if small) to either react or have
a sort of high-priority daemon do something about it. With
an infinite loop at the highest priority, the crash is
instantaneous, guaranteed and impossible to counter, even
if you had an infinite amount of RAM and CPU. This is why
SCHED_FIFO is only available to root by default. Now

Search

Enter bug ID or keywords:

Search

[Show all open bugs](#)

Dapper changes that default to allow all users to use SCHED_FIFO without restriction. If you allow this, I think you might as well make /sbin/halt (and the like) suid root, as it has *exactly* the same effect.

As to the solution, it's already there. Kernels 2.6.12 and above incorporated Ingo Molnar's rt-limits patches which allow users to use SCHED_FIFO (wasn't even possible before AFAIK). It's also possible to limit:

- 1) Who can use it
- 2) The maximum priority allowed for non-root users
- 3) How much CPU a non-root task can eat before the OS automatically sets it to non-RT priority.

I don't remember all the details and the website I got the info from is down, but it involved using pam and having something like the following in /etc/security/limits.conf:


```
* - rt_priority 0
* - nice 0
@audio - rt_priority 50
@audio - nice 10
@audio - memlock 10000
```

This would mean only members of the audio group are allowed RT priority at a max priority of 50. IIRC, the default CPU time allowed to non-root users is 70%.


In any case, the current behaviour is AFAIK totally unacceptable. I'm doing a lot of work with audio and it's nice to see that users are now allowed to set rt priority, but this is going *way* too far. Even the realtime-lsm module at least restricted RT priority to members of a certain group. Given that the solution is already in all the standard tools, I suppose the fix would only involve changing a few scripts and config files.

 jmspeex wrote on 2006-07-11: ([permalink](#))


Actually, some info is available at http://ubuntustudio.com/wiki/index.php/Dapper:Studio_Preparation specifically for Dapper

 **Ben Collins** wrote on 2006-07-12: ([permalink](#))

Does this have anything to do with the realcap kernel module? If so, I know that it is not loaded automatically, and also has module params that enable it to be locked to certain guid's and other restrictions.


 **Martin Pitt** wrote on 2006-07-12: ([permalink](#))

In any way I would expect that SCHED_FIFO is restricted to CAP_SYS_NICE processes; capabilities(7) and sched_setscheduler(2) seem to indicate that this is the intended behaviour.

 **Ben Collins** wrote on 2006-07-12: ([permalink](#))

Putting this in limits.conf fixed things for me:

```
* - rtprior 0
* - nice 0
@audio - rtprior 50
@audio - nice 10
@audio - memlock 10000
```

 **Ben Collins** wrote on 2006-07-12: ([permalink](#))

Sorry, that should be rtprio, not rtprior

 **Matt Zimmerman** wrote on 2006-07-12: **Re: [Bug 52600] Re: Unprivileged users can monopolize CPU using SCHED_FIFO and max priority** ([permalink](#))

On Tue, Jul 11, 2006 at 06:09:58PM -0000, Ben Collins wrote:

> Putting this in limits.conf fixed things for me:

>

> * - rtprior 0

> * - nice 0

> @audio - rtprior 50

> @audio - nice 10

> @audio - memlock 10000

Surely unprivileged users shouldn't be able to use a realtime scheduler by

default, though, regardless of resource limits? Did the kernel policy on this change?

[\[...\]](#)

 jmspeex wrote on 2006-07-12: ([permalink](#))

> Surely unprivileged users shouldn't be able to use a realtime scheduler by
> default, though, regardless of resource limits? Did the kernel policy on
> this change?


Well, I would say the user logged on the console should be able to use a realtime scheduler under certain restrictions, like not taking 100% of the CPU. The only problem with the current setup is that **all** users can use a realtime scheduler and with no restrictions. If the limits.conf thing works, that would probably be the best solution (I'm going to try that...).

 jmspeex wrote on 2006-07-12: ([permalink](#))

OK, I tested the change to limits.conf. It solved 2 of the 3 problems I noted earlier. The third is still not addressed: there should be a limit in the amount of CPU an unprivileged task is allowed to eat. I remember the default in 2.6.12 being 50%-70% and I don't know why it's 100% (i.e. no limit) in Dapper.

 jmspeex wrote on 2006-07-12: ([permalink](#))

Looks like the name of the thing that must be set is RLIMIT_RT_CPU (see <http://lwn.net/Articles/120797/>), but I can't find how to set that. It would actually help me if someone could point out what script/config file is responsible for allowing rt priority in the first place (the kernel restricts that to privileged processes by default). BTW, I'm sure this is not a kernel bug, but a distro config bug because other distros aren't affected.

 Martin Pitt wrote on 2006-07-13: ([permalink](#))

I subscribed Colin since he applied the RTPRIO patch to

PAM and might know about this issue.

adding the mentioned snippet in limits.conf would be possible in a security update; however, it's a conf file which the user might have modified manually, and in this case we cannot automatically upgrade it.

Therefore I would prefer doing it the other way round: the implicit default (provided by the kernel, I assume?) should be to not allow SCHED_FIFO to normal users (just as suggested by sched_setscheduler(2)). Then limits.conf can be used to increase the privileges for certain users, and I'm fine with shipping a PAM which does that in dapper-updates. Would that be possible?


If not, we have to use the first approach and write a stanza about it in the USN.

Thanks!

 **jmspeex** wrote on 2006-07-14: ([permalink](#))

I enquired about RLIMIT_RT_CPU on lkml and here's the response I got: <http://lkml.org/lkml/2006/07/13/224>

The interesting part is: "This problem should be addressed by a userspace RT watchdog. Ubuntu should not have shipped their system with unlimited non-root realtime enabled and no watchdog.". That's very similar to the other opinions I got (e.g. from Con Kolivas). Anyone can write (or knows where to find) such a watchdog?

 **Matt Zimmerman** wrote on 2006-07-14: ([permalink](#))

On Thu, Jul 13, 2006 at 10:54:20PM -0000, jmspeex wrote:

> I enquired about RLIMIT_RT_CPU on lkml and here's the response I got:

> <http://lkml.org/lkml/2006/07/13/224>

>

> The interesting part is: "This problem should be addressed by a

> userspace RT watchdog. Ubuntu should not have shipped their system with

> unlimited non-root realtime enabled and no watchdog.". That's very

> similar to the other opinions I got (e.g. from Con Kolivas). Anyone can

> write (or knows where to find) such a watchdog?

We don't want a watchdog; we should revert to the old


semantics where a
realtime capability was required in order to use a
realtime scheduler.
That's the simplest way to get back to a secure state.
The security
mechanisms we have in place assumed that this was the
case, and they require
a rethink if we change those semantics.

[\[...\]](#)

 jmspeex wrote on 2006-07-14: ([permalink](#))

```
> We don't want a watchdog; we should revert to the old
> semantics where a
> realtime capability was required in order to use a
> realtime scheduler.
> That's the simplest way to get back to a secure state.
> The security
> mechanisms we have in place assumed that this was the
> case, and they require
> a rethink if we change those semantics.
```

There's two issues here. First, yes we want to restrict the users who have access to rt scheduling. This can be achieved simply by making rtprio off by default and having only members of (e.g.) the audio group specified through /etc/security/limits.conf. Now, the second issue is what I raised in the mailing list. Even for unprivileged users that can run rt tasks (e.g. in the audio group), they shouldn't be able to lock up the system. For this part, a watchdog *is* necessary and the right way to do things. With all that, you can pretty much get the best of both worlds, i.e. low-latency audio and no security (lockup) issue.

 Matt Zimmerman wrote on 2006-07-14: ([permalink](#))

```
On Fri, Jul 14, 2006 at 12:11:06AM -0000, jmspeex wrote:
> > We don't want a watchdog; we should revert to the old
> semantics where a
> > realtime capability was required in order to use a
> > realtime scheduler.
> > That's the simplest way to get back to a secure
> > state. The security
> > mechanisms we have in place assumed that this was the
> > case, and they require
> > a rethink if we change those semantics.
> >
> > There's two issues here. First, yes we want to restrict
```

the users who
> have access to rt scheduling. This can be achieved
simply by making
> rtprio off by default and having only members of (e.g.)
the audio group
> specified through /etc/security/limits.conf.

It isn't that simple. The audio group is defined as
allowing access to
audio devices, not having privileges to DoS the system
using the scheduler,
and changing that should be an explicit decision, not a
workaround for a
bug.

> Now, the second issue is what I raised in the mailing
list. Even for
> unprivileged users that can run rt tasks (e.g. in the
audio group), the
> shouldn't be able to lock up the system. For this part,
a watchdog *is*
> necessary and the right way to do things. With all
that, you can pretty
> much get the best of both worlds, i.e. low-latency
audio and no security
> (lockup) issue.

We're not going to add non-trivial functionality like
this in order to fix a
security vulnerability; we'll fix it in the simplest and
safest way that
won't disrupt our users.

[\[...\]](#)


 jmspeex wrote on 2006-07-14: ([permalink](#))

> It isn't that simple. The audio group is defined as
allowing access to
> audio devices, not having privileges to DoS the system
using the scheduler,
> and changing that should be an explicit decision, not a
workaround for a
> bug.

Sure, but the real solution is simply limiting the amount
of CPU available with RT priority.

> We're not going to add non-trivial functionality like
this in order to fix a
> security vulnerability; we'll fix it in the simplest
and safest way that
> won't disrupt our users.

It *will* disrupt users if you change it now. At the moment, anyone using jack or other real-time audio software have their stuff working fine. If you decide to just revert to not allowing RT priority for the audio group (or the user of the console or whatever), then you just broke things that were previously working and that's no more acceptable I think. I assume whoever made the change to allow unprivileged users to use RT priority (who was it) was doing that to improve audio (and other stuff) performance in Ubuntu.

 **Matt Zimmerman** wrote on 2006-07-15: ([permalink](#))

On Fri, Jul 14, 2006 at 02:24:33AM -0000, jmspeex wrote:

> > It isn't that simple. The audio group is defined as allowing access to
> > audio devices, not having privileges to DoS the system using the scheduler,
> > and changing that should be an explicit decision, not a workaround for a
> > bug.

>

> Sure, but the real solution is simply limiting the amount of CPU

> available with RT priority.

In the long term, perhaps. For a security update to our stable release, we need to be more conservative.

> > We're not going to add non-trivial functionality like this in order to fix a
> > security vulnerability; we'll fix it in the simplest and safest way that
> > won't disrupt our users.

>

> It *will* disrupt users if you change it now. At the moment, anyone

> using jack or other real-time audio software have their stuff working

> fine. If you decide to just revert to not allowing RT priority for the

> audio group (or the user of the console or whatever), then you just

> broke things that were previously working and that's no more acceptable

> I think. I assume whoever made the change to allow unprivileged users to

> use RT priority (who was it) was doing that to improve audio (and other

> stuff) performance in Ubuntu.

Better to break it for those few users now, the ones who started with Dapper and are actively using the real-time capability, than to silently introduce security vulnerabilities or complicated resource monitoring infrastructure for everyone.

[\[...\]](#)


 jmspeex wrote on 2006-07-15: ([permalink](#))

```
> > Sure, but the real solution is simply limiting the
amount of CPU
> > available with RT priority.
>
> In the long term, perhaps. For a security update to our
stable release, we
> need to be more conservative.
```

Breaking stuff that used to work is not something I consider conservative. Especially if done silently in a security update.

```
> Better to break it for those few users now, the ones
who started with Dapper
> and are actively using the real-time capability, than
to silently introduce
> security vulnerabilities or complicated resource
monitoring infrastructure
> for everyone.
```

Or maybe just introduce the minimal amount of monitoring to just keep everyone happy. I've just found the `das_watchdog` real-time watchdog. It doesn't have much features/control, but at least it can prevent a lockup.

 Matt Zimmerman wrote on 2006-07-15: ([permalink](#))

On Fri, Jul 14, 2006 at 11:36:27PM -0000, jmspeex wrote:

```
> > > Sure, but the real solution is simply limiting the
amount of CPU
> > > available with RT priority.
> >
> > In the long term, perhaps. For a security update to
our stable release, we
> > need to be more conservative.
>
> Breaking stuff that used to work is not something I
consider
```

> conservative. Especially if done silently in a security update.


The fact that this works is a bug, a security vulnerability, not a feature.
Fixing it is like breaking a "working" root exploit.

[\[...\]](#)

 jmspeex wrote on 2006-07-15: ([permalink](#))

> The fact that this works is a bug, a security vulnerability, not a feature.
> Fixing it is like breaking a "working" root exploit.

No, the feature has been wrongly implemented, but is not a bug in itself. It *is* desirable for an unprivileged user to be able to run tasks at rt priority when the task doesn't eat too much CPU. Say you found that it's possible to abuse ping, which is suid root. Would the solution be to remove the suid flag on ping or just to make sure people can't abuse it. Unprivileged real-time scheduling *can* be made safe. No need to throw the baby out with the bath water.

 Matt Zimmerman wrote on 2006-07-16: ([permalink](#))

On Sat, Jul 15, 2006 at 09:54:34AM -0000, jmspeex wrote:

> > The fact that this works is a bug, a security vulnerability, not a feature.
> > Fixing it is like breaking a "working" root exploit.
>
> No, the feature has been wrongly implemented, but is not a bug in
> itself. It *is* desirable for an unprivileged user to be able to run
> tasks at rt priority when the task doesn't eat too much CPU. Say you
> found that it's possible to abuse ping, which is suid root. Would the
> solution be to remove the suid flag on ping or just to make sure people
> can't abuse it. Unprivileged real-time scheduling *can* be made safe. No
> need to throw the baby out with the bath water.

It would be desirable if it were intentional, but it was not, not for this release. As I have already told you, we would like to implement this

properly for a future release, but backporting a new feature is not the right way to approach this dangerous regression.

[\[...\]](#)

 jmspeex wrote on 2006-07-16: ([permalink](#))


Sigh I give up...

(note to self: think twice before reportung Ubuntu bugs)

 jmspeex wrote on 2006-07-17: ([permalink](#))

Oh, and any intention on addressing the fact that typing "limit" on my system gives me (among other things):
memorylocked unlimited
maxproc unlimited

No user would ever "while(1)fork();", right?

 Matt Zimmerman wrote on 2006-07-17: ([permalink](#))


On Sun, Jul 16, 2006 at 02:11:52AM -0000, jmspeex wrote:
> *Sigh* I give up...
>
> (note to self: think twice before reportung Ubuntu bugs)

I'm afraid we won't always choose to resolve an issue in the way that you think we should, but I've tried to explain our rationale as best I can.

I wrote a wiki page some time ago to explain about our release process, and some of the information there applies here as well.

<https://wiki.ubuntu.com/TimeBasedReleases>

[\[...\]](#)


 Matt Zimmerman wrote on 2006-07-17: ([permalink](#))

On Mon, Jul 17, 2006 at 07:46:19AM -0000, jmspeex wrote:
> Oh, and any intention on addressing the fact that typing "limit" on my system gives me (among other

```
things):  
> memorylocked unlimited  
> maxproc unlimited  
>  
> No user would ever "while(1)fork();", right?
```

Please limit discussion in this bug report to the issue at hand; if you'd like to (re-)raise the topic of default resource limits, please read up on the previous discussion and then start a thread on ubuntu-devel@lists.ubuntu.com.

[\[...\]](#)

 **Martin Pitt** wrote on 2006-07-21: **Re: [Bug 52600] Re: [Bug 52600] Re: Unprivileged users can monopolize CPU using SCHED_FIFO and max priority** ([permalink](#))

Hi Matt, hi Ben,

Matt Zimmerman [2006-07-13 23:40 -0000]:

```
> We don't want a watchdog; we should revert to the old  
semantics where a  
> realtime capability was required in order to use a  
realtime scheduler.  
> That's the simplest way to get back to a secure state.  
The security  
> mechanisms we have in place assumed that this was the  
case, and they require  
> a rethink if we change those semantics.
```

I fully agree here. Changing PAM defaults etc. to allow limited realtime privileges to normal users sounds fine for edgy, but not retroactively for Dapper.

Ben, adding a capability check doesn't sound that hard, is it? Do you agree to this solution?

Thank you,

Martin

--

Martin Pitt <http://www.piware.de>
Ubuntu Developer <http://www.ubuntu.com>
Debian Developer <http://www.debian.org>

In a world without walls and fences, who needs Windows and Gates?


 jmspeex wrote on 2006-07-21: ([permalink](#))

> Ben, adding a capability check doesn't sound that hard,
is it? Do you
> agree to this solution?

How about simply adding the following to /etc/security/
limits.conf as suggested by Ben above:


```
* - rtprio 0  
* - nice 0
```

At least it's easy to revert to the right behaviour for
those who want rt to work.


 Ben Collins wrote on 2006-07-22: **Re: [Bug 52600]
Re: [Bug 52600] Re: Unprivileged users can
monopolize CPU using SCHED_FIFO and max priority**
([permalink](#))

On Fri, 2006-07-21 at 12:48 +0200, Martin Pitt wrote:
[\[...\]](#)

Yeah, I'm checking the source code now.

 Martin Pitt wrote on 2006-09-25: ([permalink](#))


Ben, what's the status on this? Can we fix this in the
next update?

 Ben Collins wrote on 2006-09-26: ([permalink](#))

Sure thing, I'll have a patch attached here for review
before upload.

 jmspeex wrote on 2006-11-16: ([permalink](#))

Just noticed that Edgy behaves exactly the same as Dapper
with respect to SCHED_FIFO rights. Is that now considered
a feature? In any case, I think after more than 4 months,
it's now safe to remove the private flag from this bug.
Oh, and I don't think it should be considered as a kernel
bug either.

 Ben Collins wrote on 2007-01-23: ([permalink](#))

I've tested this script on our feisty kernel, and it's affected. Thing is, there are no scheduler patches to that kernel. It's all stock in that area.

I rechecked the dapper kernel, and there as well, are no scheduler patches. This isn't Ubuntu specific. I've yet to verify this on stock kernels, but feisty's kernel is about as close to stock as you can get.

After discussion with Kyle, I think we can chalk this up to upstream policy on real time capabilities.

Very likely, sane defaults in userspace need to be set by us (via the suggested pam.d config).

 jmspeex wrote on 2007-01-23: ([permalink](#))

Of course it does it with a stock kernel. The problem isn't with the kernel, but with the userspace configuration. The fix is to change the /etc/security/limits.conf as detailed above or (even better) include a realtime watchdog. The best solution would be to have the kernel doing the policy just like what the original implementation by Ingo Molnar. Don't know why it got dropped when merging in 2.6.12.

 jmspeex wrote on 2007-01-23: ([permalink](#))

Oh, and *please* remove the private flag from this bug. Given how old the bug is, it's not like making it public can do any more damage and it would allow other people to comment.

[Add a comment/attachment](#)

What next?

- » [Report another bug](#) about linux-source-2.6.15 in ubuntu
- » [List open bugs](#) for linux-source-2.6.15 in ubuntu

[Help improve Launchpad](#)

© 2004-2008 [Canonical Ltd.](#) | [Terms of use](#)