Subject: Re: [PATCH 0/9] namespaces: Introduction
Posted by Herbert Poetzl on Fri, 19 May 2006 12:42:35 GMT
View Forum Message <> Reply to Message

On Thu, May 18, 2006 at 10:34:30AM -0700, Andrew Morton wrote:
> "Serge E. Hallyn" <serue@us.ibm.com> wrote:
> >
> > This patchset introduces a per-process utsname namespace. These can
> > be used by openvz, vserver, and application migration to virtualize and
> > isolate utsname info (i.e. hostname). More resources will follow, until
> > hopefully most or all vserver and openvz functionality can be implemented
> > by controlling resource namespaces from userspace.
>
> Generally, I think that the whole approach of virtualising the OS
> so it can run multiple independent instances of userspace is a good
> one. It's an extension and a strengthening of things which Linux is
> already doing and it pushes further along a path we've been taking
> for many years.

yes, I too think that Linux has been moving in that
direction for a long time now, maybe even too long, so
that other OSes (like BSD or Solaris) already managed
to get a virtualization layer up and running (although
maybe at a much simpler level than we plan to do)

> If done right, it's even possible that each of these featurettes could
> improve the kernel in its own right - better layering, separation,
> etc.

agreed, most 'features' will require a cleanup of some
otherwise completely untouched areas, which (hopefully)
will improve those areas ...

> The approach which you appear to be taking is to separate the bits
> of functionality apart and to present them as separate works each of
> which is reviewed-by, acceptable-to and will-be-used-by all of the
> interested projects. That's ideal, and is very much appreciated.

IMHO many things will make perfect sense on their own
even without the 'other' virtualizations or isolations.
With Linux-VServer it's an every day occurance, that
folks just 'cherry pick' the isolation features and build
their own level of virtual/isolated environment.

at this point, many thanks to Sam, Eric and Serge
who do a really good job in massaging patches :)

> All of which begs the question "now what?".

>
> What we do _not_ want to do is to merge up a pile of infrastructural
> stuff which never gets used. On the other hand, we don't want to be in
> a position where nothing is merged into mainline until the entirety of
> vserver &&/|| openvs is ready to be merged.

yes, I agree here, and I'm pretty sure that we are
still missing many 'stakeholders' here just because
we do not see all possible areas of use ... let me
give a simple example here:

"pid virtualization"

- Linux-VServer doesn't really need that right now.
  we are perfectly fine with "pid isolation" here, we
  only "virtualize" the init pid to make pstree happy

- Snapshot/Restart and Migration will require "full"
  pid virtualization (that's where Eric and OpenVZ
  are heading towards)

- OpenSSI and *Mosix require system wide pid spaces
  which probably could be implemented with virtual
  pid spaces as well

- many security addons provide something called pid
  randomization, and I think they could probably
  benefit from a virtual pid space, too

now does that mean that e.g. Linux-VServer is against
"pid virtualization"? well, we are mainly against all
unnecessary overhead and strictly against losing the
ability to keep it simple for the user, i.e. somebody
who does not require all that stuff should be able to
pick the features (or spaces) she really needs ...

> I see two ways of justifying a mainline merge of things such as this
>
> a) We make an up-front decision that Linux _will_ have
>   OS-virtualisation capability in the future and just start putting
>   in place the pieces for that, even if some of them are not
>   immediately useful.

as long as this doesn't automatically mean bloat, I'm
more than happy with such a decision ...

>   I suspect that'd be acceptable, although I worry that we'd get
>   partway through and some issues would come up which are

> irreconcilable amongst the various groups.

I'm pretty sure that we _will_ hit some issues, but
I'm also sure that we will be able to work out those
issues, after all the 'end user' has to decide what
should be in mainline and what not ...

> It would help set minds at ease if someone could produce a
> bullet-point list of what features the kernel will need to get
> it to the stage where "most or all vserver and openvz functionality
> can be implemented by controlling resource namespaces from
> userspace." Then we can discuss that list, make sure that
> everyone's pretty much in agreement.

excellent idea, will start preparing such a list
from our PoV, so that we can merge that with the
other lists ...

> It would be good if that list were to identify which features are
> useful to Linux in their own right, and which ones only make
> sense within a whole virtualise-the-OS setup.

that's probably the hardest part ...

> b) Only merge into mainline those feature which make sense in a
> standalone fashion. eg, we don't merge this patchset unless the
> "per-process utsname namespace" feature is useful to and usable
> by a sufficiently broad group of existing Linux users.

the question here is, who are the users and _how_
will they get the feature? I see the following cases
here, which might overlap ...

- the feature makes perfectly sense in mainline as
  standalone feature (maybe even adds no overhead
  and/or simplifies/generalizes design) but has no
  direct 'user' per se (think private namespaces
  or linux capabilities)

- the feature is used by a number of projects in
  very different ways to improve or even realize
  certain 'other' features (think ext2/3 xattrs)

- the feature (although it adds pretty much overhead
  and/or complicates the design) is really useful
  for everyday use, and most folks who discover it
  do not understand how they could live without it
  (think various attributes on vfs mounts :)

> I suspect this will be a difficult approach.

yes, but should 'we' decide to take this approach, we
can at least guarantee that we (Linux-VServer) will
try to make use of those new features as soon as they
appear in mainline (as we did until now)

> The third way would be to buffer it all up in -mm until everything
> is sufficiently in place and then slam it all in.

for me, that sounds like a pretty bad idea, at least
for the first steps -- though we might consider this
approach for the last 10 or 20 percent, when we just
have to put the pieces together ...

> That might not be feasible for various reasons - please advise..
>
> A fourth way would be for someone over there to run a git tree - you
> all happily work away, I redistribute it in -mm for testing and one
> day it's all ready to merge. I don't really like this approach. It
> ends up meaning that nobody else reviews the new code, nobody else
> understands what it's doing, etc. It's generally subversive of the way
> we do things.

let me say that I'm strictly against such an approach
as it would be very similar to merging any of the
existing projects without further mainline consideration

> Eric, Kirill, Herbert: let us know your thoughts, please.

thanks for your work and time, we appreciate it

best,
Herbert